

Leaky Distilled Monte Carlo for Efficient Uncertainty Quantification

Alireza Maleki¹[0009-0000-9505-624X] (✉),
Diego Perez-Palacin^{1,2}[0000-0002-9646-1156], Vincenzo
Scotti¹[0000-0002-8765-604X], and Raffaella Mirandola¹[0000-0003-3154-2438]

¹ Karlsruhe Institute of Technology, Karlsruhe, Germany

`alireza.maleki@kit.edu`

`vincenzo.scotti@kit.edu` `raffaella.mirandola@kit.edu`

² Linnaeus University, Växjö, Sweden

`diego.perez@lnu.se`

Abstract. Deep Neural Networks (DNNs) often produce overconfident predictions even when their inferences are incorrect. This kind of unexpected behavior is especially likely and possibly harmful when dealing with Out-of-Distribution (OOD) samples. The occurrence of these situations underscores the need to quantify the uncertainty of the model’s output to mitigate its overconfidence. In this context, Monte Carlo (MC) sampling is a widely used approach for quantifying uncertainty in estimators like DNNs. However, this methodology requires processing a data point with the DNN multiple times, resulting in a computationally expensive and time-consuming process that is unsuitable for time-sensitive applications. The teacher–student framework is designed to deal with these situation by distilling a teacher model knowledge into a more efficient student model, offering a mitigation to the MC sampling cost. Yet, this framework typically fails to capture the teacher’s sampling behavior in the student and, thus, to generalize beyond the training data. In this work, we propose a revised teacher–student framework, *Leaky Distilled Monte-Carlo* (LDMC), that distills both the MC sampling process and the teacher’s predictive knowledge. LDMC improves the accuracy of uncertainty estimation across the evaluated datasets, achieving gains of up to 30% relative to existing single-pass methods when compared to the full MC estimate, while increasing the inference time by only about 7–20% of a single forward pass.

Keywords: Monte-Carlo · uncertainty · distillation · teacher-student

1 Introduction

As reliance on Deep Neural Networks (DNNs) grows, so does the risk associated with their failures [6]. In domains like healthcare, autonomous driving, and robotics, reliable model inference is essential to prevent harmful or costly errors [18]. Similarly, the increasing deployment of large-scale chatbots and decision-support systems highlights the need for reliable fact-checking and confidence-aware predictions. In this context, uncertainty estimation of the inference provides a quantitative

measure of predictive reliability and has become a crucial component of trustworthy machine learning systems [12,27].

Uncertainty can be characterized along multiple dimensions [11]. A commonly adopted taxonomy distinguishes between *aleatoric uncertainty*, which arises from inherent noise in the data, and *epistemic uncertainty*, which reflects uncertainty due to limited model knowledge. From a knowledge-based perspective, model inputs can be broadly divided into *knowns* and *unknowns*. *Knowns* correspond to inputs that are well represented in the training distribution and for which the model can make reliable inferences. In contrast, *unknowns* encompass inputs that fall outside the model’s reliable knowledge [11]. When we do not have high observational noise, *known* inputs are typically associated with low uncertainty, whereas *unknown* inputs should yield high uncertainty estimates.

In turn, DNNs are well known to produce confident but incorrect predictions when faced with out-of-distribution (OOD) inputs, insufficient training data, or corrupted samples. This is an overconfident behavior that motivates the need for principled uncertainty estimation mechanisms.

Since the distributions of model parameters and outputs are generally intractable [7], uncertainty must be estimated rather than computed exactly. From a Bayesian perspective, Monte Carlo (MC)–based methods [3] approximate the predictive posterior distribution by performing multiple stochastic forward passes using mechanisms such as model ensembles [21] or MC Dropout (MCD), which randomizes dropout masks to simulate sampling from the posterior [7]. Although MC–based methods provide strong performance, their computational cost limits practical deployment. In particular, repeated stochastic forward passes at inference time make them unsuitable for latency-sensitive applications. Recent work has explored distillation-based approaches to compress MC-based uncertainty into a single forward pass [15,24]. However, these approaches often suffer from degraded uncertainty quality, especially on OOD or previously unseen data.

To address this limitation, we introduce *Leaky Distilled Monte Carlo (LDMC)* method, which is a teacher–student distillation framework where a student DNN is trained to approximate the behavior of a teacher DNN [10]. To enable this information leak, we propose *Quantized Monte-Carlo Dropout (QMCD)*, which quantizes the dropout mask distribution. With a single forward pass, the student approximates the teacher’s uncertainty estimates, despite having only partial access to the teacher’s stochastic sampling process. By exposing the student to incomplete information about the teacher’s MC sampling process during training, the student learns to infer the teacher’s uncertainty behavior, enabling efficient single-pass uncertainty estimation at inference time.

The main contributions of this paper are: (1) A teacher–student distillation framework that compresses MC-based uncertainty estimation into a single forward pass. The student is trained using partial information about the teacher’s dropout masks, encouraging robust approximation of the sampling behavior. (2) A controlled variant of MCD that closely approximates standard MCD while reducing complexity by quantizing the distribution of dropout masks. (3) An evaluation of

the framework under a structured data split, demonstrating improved uncertainty estimation compared to state-of-the-art (SotA) baselines.

The remainder of this paper is organized as follows. Section 2 reviews related work on uncertainty estimation. Section 3 presents background on MCD and teacher–student uncertainty estimation. Section 4 introduces LDMC. Section 5 reports the experimental evaluation. Section 6 concludes the paper.

2 Related Work

We organize the related work into two categories: uncertainty estimation in deep learning and efficient uncertainty estimation via distillation.

2.1 Uncertainty Estimation in Deep Learning

A wide range of methods has been proposed for uncertainty estimation in deep learning [8], including Bayesian approaches [13] and evidence-based methods [26]. Bayesian neural networks (BNNs) model uncertainty by placing probability distributions over network parameters and performing inference via posterior sampling [13]. While exact Bayesian inference provides a principled framework for uncertainty quantification, its computational cost renders it impractical for modern deep neural networks, thereby motivating approximate methods.

To enable scalable uncertainty estimation in deep neural networks, several approximate Bayesian inference methods have been proposed. Among them, Monte Carlo Ensemble (MCE) [21] and MCD [7] are particularly popular due to their simplicity and effectiveness [27]. MCE involves training multiple independently initialized models, which are treated as samples from the posterior distribution. Each input is passed through all models, and the outputs are aggregated to quantify the predictive uncertainty across the ensemble. MCD [7] approximates Bayesian inference by applying stochastic dropout masks at inference time, enabling uncertainty estimation using a single trained model. As shown in [7], increasing the number of MCD samples leads the uncertainty estimate to converge toward the BNN approximation. To obtain more stable uncertainty estimates, a controlled variant of MCD (CMCD) was introduced, in which MC sampling is performed by randomly selecting from a set of pre-sampled masks rather than drawing new masks from the original distribution at each forward pass [9]. Due to their simplicity and effectiveness, MCE and MCD form the backbone of many SotA uncertainty estimation methods.

Another line of work approximates the Bayesian posterior locally around a trained network using the Laplace approximation. Laplace Redux [4] revisits this classical approach and demonstrates that a Gaussian approximation centered at the maximum a posteriori solution can provide competitive uncertainty estimates. Among these approximations, variational inference methods learn a tractable variational posterior to approximate the intractable Bayesian posterior, enabling scalable uncertainty estimation in deep models [2,19].

2.2 Efficient Uncertainty Estimation and Distillation

Despite their performance, MC-based approaches require multiple stochastic forward passes, resulting in significant computational overhead. This limitation has spurred research into efficient alternatives that preserve uncertainty quality while reducing inference cost. Distillation-based approaches, such as the teacher–student framework [10], aim to transfer the predictive distribution or uncertainty estimates of a teacher model—the model being distilled—into a single-forward-pass student model [15,24]. These approaches enable uncertainty estimation with a single forward pass, making them attractive for real-time applications. In Bayesian Dark Knowledge [15], the student is trained to directly predict the output distribution produced by the MC-sampled teacher, from which the total uncertainty is later derived. In Bayesian Bright Knowledge (BBK) [24] the student is augmented with an additional output head dedicated to estimating the epistemic uncertainty provided by the teacher. Hydra distillation method [25] preserves the predictive diversity of the original ensemble by utilizing a multi-headed architecture where separate lightweight heads mimic individual ensemble members through a shared body network.

However, existing distillation techniques often struggle to generalize under distribution shifts and tend to underestimate uncertainty for OOD inputs. In particular, distilled students may fail to capture the stochastic behavior of MC-based teachers, leading to degraded uncertainty estimation. Our work addresses this challenge by allowing the student to exploit partial information about the teacher’s sampling process, resulting in improved uncertainty fidelity without incurring MC inference costs.

3 Background

In this section we present MCD for uncertainty estimation in DNNs and the application of the BBK framework for efficient uncertainty estimation.

3.1 Monte-Carlo Dropout

MCD-based address the problem of uncertainty estimation for a given DNN model θ [7]. For a classification problem over K classes $(\mathcal{C}_1, \dots, \mathcal{C}_k, \dots, \mathcal{C}_K)$, given an input $\mathbf{x}$, the predictive (total) uncertainty is quantified using the entropy of the model’s output distribution $\mathbf{y} = p_\theta(k|\mathbf{x})$ [14]. A high entropy indicates a near-uniform predictive distribution, reflecting high uncertainty, whereas a low entropy implies a concentrated distribution, indicating high confidence. The total uncertainty, $u_t(\mathbf{x})$, is defined as

$$u_t(\mathbf{x}) = H[p_\theta(k|\mathbf{x})] = - \sum_{k=1}^K P_\theta(\mathcal{C}_k|\mathbf{x}) \log P_\theta(\mathcal{C}_k|\mathbf{x}). \quad (1)$$

Since the true predictive distribution $p_\theta(k|\mathbf{x})$ is generally intractable, we approximate it using MC sampling. The DNN is equipped with D dropout layers [7], each with dropout probability p_{drop} , which enables MCD. At each iteration $t \in \{1, \dots, T\}$

and for each dropout layer $d \in \{1, \dots, D\}$ of the DNN, dropout masks $\mathbf{m}_d^{(t)}$ are independently sampled from a Bernoulli distribution, producing outputs that are aggregated to form an empirical distribution

$$q(k|\mathbf{x}) \approx p(k|\mathbf{x}). \quad (2)$$

The i -th element in the dropout mask of the d -th layer, denoted by $m_{d,i}^{(t)}$, is sampled as

$$m_{d,i}^{(t)} = \begin{cases} \frac{1}{1-p_{drop}}, & \text{with probability } 1-p_{drop}, \\ 0, & \text{with probability } p_{drop}. \end{cases} \quad (3)$$

where the scaling factor $1/(1-p_{drop})$ ensures that the expected value of the mask remains equal to one.

Let $\mathbf{z}^{(t)}(\mathbf{x})$ denote the output logits of θ at MC iteration t , defined as

$$\mathbf{z}^{(t)}(\mathbf{x}) = f_\theta(\mathbf{x}; \mathcal{M}^{(t)}), \quad (4)$$

where $f_\theta(\cdot)$ is the function computed by the DNN and $\mathcal{M}_t$ denotes the set of dropout masks applied at iteration t :

$$\mathcal{M}^{(t)} = \{\mathbf{m}_1^{(t)}, \dots, \mathbf{m}_D^{(t)}\}. \quad (5)$$

For a fixed mask realization $\mathcal{M}^{(t)}$, we obtain a local predictive distribution by applying the softmax($\cdot$) function to the logits, yielding $q_\theta(k|\mathbf{x}; \mathcal{M}^{(t)})$. By aggregating predictions across multiple (T) MC samples, we can then estimate the empirical predictive distribution as follows:

$$\begin{aligned} q_\theta(k|\mathbf{x}; \mathcal{M}^{(t)}) &= \text{softmax}(\mathbf{z}^{(t)}(\mathbf{x})) \\ q_\theta(k|\mathbf{x}) &= \frac{1}{T} \sum_{t=1}^T q_\theta(k|\mathbf{x}; \mathcal{M}^{(t)}) \end{aligned} \quad (6)$$

Given the local predictive distributions $q_\theta(k|\mathbf{x}; \mathcal{M}^{(t)})$ obtained from individual MC samples, we can further decompose the predictive uncertainty into aleatoric and epistemic components. The aleatoric uncertainty, denoted by $u_a(\mathbf{x})$, is quantified as the expected entropy of the local predictive distributions [14]. The epistemic uncertainty, denoted by $u_e(\mathbf{x})$, is captured by the mutual information between the model parameters and the predictions [14], which can be computed as the difference between the total uncertainty and the aleatoric uncertainty. As we do not have access to p_θ , we use q_θ to estimate $\hat{u}_t$, $\hat{u}_a$, and $\hat{u}_e$. Formally, these quantities are defined as follows:

$$\hat{u}_t(\mathbf{x}) = H[q_\theta(k|\mathbf{x})], \quad (7) \quad \hat{u}_a(\mathbf{x}) = \frac{1}{T} \sum_{t=1}^T H[q_\theta(k|\mathbf{x}; \mathcal{M}^{(t)})], \quad (8)$$

$$\hat{u}_e(\mathbf{x}) = \hat{u}_t(\mathbf{x}) - \hat{u}_a(\mathbf{x}). \quad (9)$$

3.2 Teacher-Student for Uncertainty Estimation

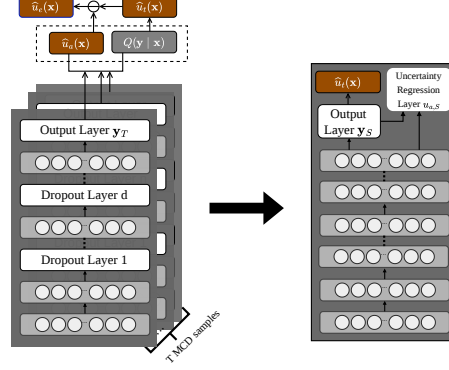


Fig. 1. The system model of a simple teacher–student method. The total uncertainty is computed from the student output layer, while epistemic uncertainty is predicted by the uncertainty quantification layer.

The teacher-student framework was first introduced to distill knowledge from a teacher model $\mathcal{T}$ into a, typically more efficient, student $\mathcal{S}$. The student is trained by minimizing the reconstruction error between its output $\mathbf{y}_\mathcal{S}$ and that of the teacher $\mathbf{y}_\mathcal{T}$.

In the context of MCD for uncertainty estimation, the use of the student DNN is twofold [24]: (i) predict the teacher output distribution (from which we can estimate $\hat{u}_{t,\mathcal{S}}(\mathbf{x}) \approx \hat{u}_{t,\mathcal{T}}(\mathbf{x})$), and (ii) predict an aleatoric uncertainty estimate $\hat{u}_{a,\mathcal{S}} \approx \hat{u}_{a,\mathcal{T}}$ (extending the architecture with a second output head).

The epistemic uncertainty estimate $\hat{u}_{e,\mathcal{S}} \approx \hat{u}_{e,\mathcal{T}}$ is predicted by removing the aleatoric component from the total uncertainty. The overall process and architectures are reported Fig. 1: $\mathcal{T}$ is used in the MC sampling process, $\mathcal{S}$ predicts both the estimate output class distribution (computing the logits from the hidden vector embedding the input) and the estimate aleatoric uncertainty estimate (using both the hidden vector and the logits).

We train the student to minimize the Kullback–Leibler (KL) divergence between its predictive distribution $\mathbf{y}_\mathcal{S} = q_\mathcal{S}(k|\mathbf{x})$ and the teacher’s MC empirical distribution $\mathbf{y}_\mathcal{T} = q_\mathcal{T}(k|\mathbf{x};\mathcal{M})$, encouraging the student to match the teacher’s predictive behavior [15]. We introduce an additional term to the loss for uncertainty supervision: the Mean Squared Error (MSE) loss between the teacher’s aleatoric uncertainty $\hat{u}_{a,\mathcal{T}}(\mathbf{x})$ and the student’s regression head output, as used in BBK [24]. The overall student’s training loss $\mathcal{L}_\mathcal{S}$ is therefore defined as follows:

$$\mathcal{L}_\mathcal{S} = \mathbb{E}_{\mathbf{x},\mathcal{M}}[\text{D}_{KL}(q_\mathcal{S}(k|\mathbf{x})||q_\mathcal{T}(k|\mathbf{x};\mathcal{M})) + \text{MSE}(\hat{u}_{a,\mathcal{S}}(\mathbf{x}), \hat{u}_{a,\mathcal{T}}(\mathbf{x};\mathcal{M}))], \quad (10)$$

where $\mathcal{M} = \{\mathcal{M}^{(1)}, \dots, \mathcal{M}^{(T)}\}$ is the set of all the sets of dropout masks generated by the MC sampling.

4 Proposed approach

In this section, we present our LDMC approach, where a student leverages partial information from the teacher’s MC sampling process and, then, describe how the teacher leaks this information to the student.

4.1 Leaky Distilled Monte-Carlo

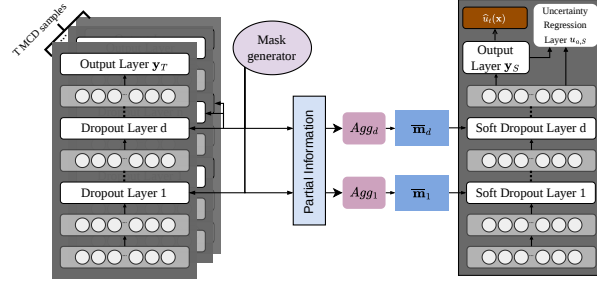


Fig. 2. System model of the LDMC method. The mask generator produces decomposable dropout masks used by the teacher for Monte Carlo sampling, while the common knowledge masks are shared with the student on the right. The aggregator module consolidates the information and forwards it to the soft dropout layers.

To provide the student with partial information about the MC sampling process in our proposed teacher-student framework, we expose $\mathcal{S}$ to information derived from the dropout masks following the process depicted in Figure 2. We employ a student model that shares the same backbone architecture as the teacher, but differs in the additional regression head (to predict u_a) and the use of *soft dropout layers* to make use of the leaked information. This enables a more general formulation of the loss in Equation (10). We model each dropout mask as a composition of two independent random tensors, $\mathbf{m} = f(\mathbf{m}_p, \mathbf{m}_s)$, where $\mathbf{m}_p$ represents mask components private to the teacher and $\mathbf{m}_s$ denotes components shared with the student as common knowledge. Under the assumption that $\mathbf{m}_p$ and $\mathbf{m}_s$ are independent, the entropy of the full mask decomposes additively as

$$H[p(\mathbf{m})] = H[p(\mathbf{m}_p)] + H[p(\mathbf{m}_s)].$$

Using this decomposition, we can rewrite Equation (10) as follows:

$$\mathcal{L}_{\mathcal{S}} = \mathbb{E}_{\mathbf{x}, \mathcal{M}_s, \mathcal{M}} [\text{D}_{KL}(q_{\mathcal{S}}(k|\mathbf{x}; \mathcal{M}_s) || q_{\mathcal{T}}(k|\mathbf{x}; \mathcal{M})) + \text{MSE}(\hat{u}_{a, \mathcal{S}}(\mathbf{x}; \mathcal{M}_s), \hat{u}_{a, \mathcal{T}}(\mathbf{x}; \mathcal{M}))], \quad (11)$$

where $\mathcal{M}_s$ is the set of masks with shared information used by $\mathcal{S}$.

Output prediction during training follows the procedure is illustrated in Figure 2. At each MC sampling iteration, a mask generator produces the dropout masks $\mathcal{M}^{(t)}$, the shared component are aggregated layerwise throughout the iterations and fed to the student in the form of soft masks $\bar{\mathcal{M}} = \{\bar{\mathbf{m}}_1, \dots, \bar{\mathbf{m}}_D\}$. The aggregation function can be any, possibly learnable, function combining the masks and preserving the values in the range $[0, 1] \subseteq \mathbb{R}$. The soft masks are applied via the hadamard product (element-wise multiplication) to the hidden vector $\mathbf{h}_d \leftarrow \mathbf{h}_d \odot \bar{\mathbf{m}}_d$. At inference time, the mask generator keeps generating masks for the student.

4.2 Quantized Monte-Carlo Dropout

As mentioned earlier, student models may exhibit reduced generalization under unseen or distribution-shifted data [20]. To convey information about the teacher model’s MC sampling process to the student and, thus, improve generalization, we need to leak data regarding the dropout masks. However, sharing all T masks sampled by the teacher is impractical because the masks are uniformly sampled (see Figure 3(a)): as T grows large, aggregating information from all masks yields no additional insight beyond their mean. Therefore, to establish an information channel between the teacher and the student for the MC process, we compress the mask distribution.

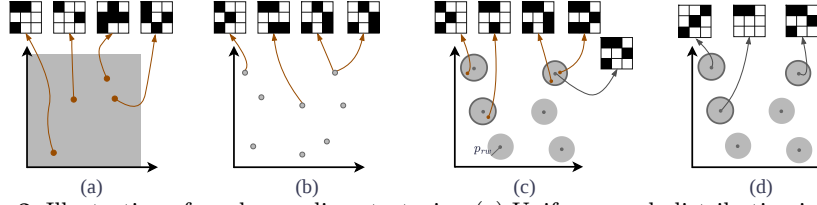


Fig. 3. Illustration of mask sampling strategies: (a) Uniform mask distribution in MCD. (b) CMCD, where M masks are sampled initially and resampled during training. (c) QMCD, where M central masks are first sampled, R masks selected during training, and T dropout samples drawn. (d) The R masks are provided to the student.

Our work generalizes over CMCD [9], which proposes to initially samples a set of M masks and then resamples from these during the MC process, as illustrated in Figure 3(b). Applying this method directly to provide MC-sampled masks to the student would result in a number of masks close to either M or T . Choosing a small M makes the process tractable, but leads to unstable uncertainty estimates and poor convergence. To address these issues, we propose a method that allows the student to receive informative mask data while maintaining stable uncertainty estimation.

We first sample M initial masks for each layer from a Bernoulli distribution with dropout rate p_{drop} to form the set

$$\mathcal{M}_I = \{\mathbf{m}_d^{(m)} \sim \text{Bernoulli}(\mathbf{p}_{drop}) \forall m \in [1, M] \subseteq \mathbb{N} \forall d \in [1, D] \subseteq \mathbb{N}\}. \quad (12)$$

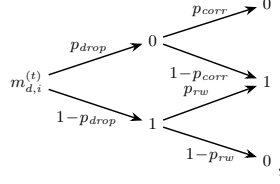


Fig. 4. Two-step Bernoulli sampling process that preserves the marginal probability distribution of each element in the mask tensor.

At each training iteration, we sample randomly without replacement $R < M$ masks from $\mathcal{M}_I$ for each dropout layer to compose $\mathcal{M}_s$. Subsequently, for each dropout layer, we sample randomly with replacement T masks using the previously R selected masks, as illustrated in Figure 3(c). Each of these $T \times D$ samples undergoes a random walk process with rate p_{rw} to finally compose $\mathcal{M}$. The overall sampling procedure is illustrated in Figure 4, with $p_{corr} = (1 - p_{drop}) / p_{drop} \cdot p_{rw}$ and $p_{rw} < p_{drop} / (1 - p_{drop})$. p_{corr} is a correction factor ensuring that the mean of the mask remains equal to 1, in fact

$$P(m_{d,i}^{(t)} = 0) = p_{drop} \cdot \left(1 - \frac{1 - p_{drop}}{p_{drop}} p_{rw} \right) + (1 - p_{drop}) \cdot p_{rw} = p_{drop}. \quad (13)$$

If the masks in $\mathcal{M}_s$ are sufficiently separated, increasing p_{rw} causes the distribution to approach the uniform distribution of standard dropout. In particular, for $p_{drop} = 1/2$ and $p_{rw} = 1/2$, the resulting distribution matches the standard dropout distribution exactly.

Algorithm 1 Quantized distribution MCD (QMCD)

Require: Input sample $\mathbf{x}$, teacher model $\mathcal{T}$, initial masks $\mathcal{M}_I$

- 1: $\mathcal{M}_s \leftarrow \{\mathbf{m}_{s,d}^{(r)} \sim \text{Unif}(\{\mathbf{m}_d^{(m)} \in \mathcal{M}_I\}) \mid \forall r \in [1, R] \subseteq \mathbb{N} \forall d \in [1, D] \subseteq \mathbb{N}\}$ ▷ W/o repl.
 - 2: **for** $t \in [1, T] \subseteq \mathbb{N}$ **do** ▷ For each iteration of MC sampling
 - 3: **for** $d \in [1, D] \subseteq \mathbb{N}$ **do** ▷ For each dropout layer
 - 4: $\hat{\mathbf{m}}_d^{(t)} \leftarrow \mathbf{m} \sim \text{Unif}(\{\mathbf{m}_{s,d}^{(r)} \in \mathcal{M}_s\})$ ▷ W/ repl.
 - 5: $\mathbf{p}_d^{(t)} \leftarrow (p_{rw} + (1 - 2p_{drop}) / p_{drop} p_{rw}) \cdot \hat{\mathbf{m}}_d^{(t)}$
 - 6: $\mathbf{n}_d^{(t)} \leftarrow \mathbf{n} \sim \text{Bernoulli}(\mathbf{p}_d^{(t)})$
 - 7: $\mathbf{m}_d^{(t)} \leftarrow \mathbf{m}_{s,d}^{(t)} \otimes \mathbf{n}_d^{(t)}$ ▷ Take random walk
 - 8: **end for**
 - 9: $\mathcal{M}^{(t)} \leftarrow \{\mathbf{m}_1^{(t)}, \dots, \mathbf{m}_D^{(t)}\}$
 - 10: $\mathbf{y}_{\mathcal{T}}^{(t)} \leftarrow q_{\mathcal{T}}(k | \mathbf{x}; \mathcal{M}^{(t)})$
 - 11: **end for**
 - 12: $\bar{\mathbf{y}}_{\mathcal{T}}^{(t)} \leftarrow 1/T \sum_{t=1}^T \mathbf{y}_{\mathcal{T}}^{(t)} \approx q_{\mathcal{T}}(k | \mathbf{x})$
 - 13: $\hat{u}_{a,\mathcal{T}}(\mathbf{x}) \leftarrow 1/T \sum_{t=1}^T H[\mathbf{y}_{\mathcal{T}}^{(t)}]$
 - 14: **return** $\bar{\mathbf{y}}_{\mathcal{T}}, \hat{u}_{a,\mathcal{T}}(\mathbf{x}), \mathcal{M}_s$
-

The QMCD uncertainty is computed as described in Algorithm 1. The algorithm outputs the average predicted distribution $\bar{\mathbf{y}}_{\mathcal{T}}$, the aleatoric uncertainty $\hat{u}_a(\mathbf{x})$, and the sub-sampled masks $\mathcal{M}_s$. These outputs are then used to train the student. It receives $\mathcal{M}_s$ as partial information from the teacher while the random tensors in $\mathcal{M}^{(t)}$ are used only by the teacher. Thus, the student is aware of the teacher’s sampling regions without accessing the full random variations.

5 Evaluation

In this section, we describe our experimental setup, we report about the evaluation of QMCD and that of the LDMC , and, finally, we discuss the results.

5.1 Experimental Setup

We assess the effectiveness of our method on two image classification datasets, namely *MNIST* [5] and *CIFAR-10* [16], and on one text classification dataset, namely *DBpedia-14* [28]. To assess the quality of uncertainty estimation, we partition the data into three groups: *known-known* (*kk*), *known-unknown* (*ku*), and *unknown-unknown* (*uu*), similar to the open-set recognition tasks [1]. The *kk* samples correspond to the in-distribution data, while *ku* and *uu* represent different types of *unknown* (OOD) data.

During training, the teacher model is trained exclusively on samples from the *kk* classes. The student model is then optimized to reproduce the teacher’s predictive distribution and uncertainty estimates on samples drawn from both the *kk* and *ku* sets. Importantly, the *ku* samples are not used as labeled OOD examples; rather, they provide additional inputs on which the student can observe the teacher’s uncertainty behaviour. At test time, both models are evaluated on *kk*, *ku*, and previously unseen *uu* classes. Since the *uu* classes are disjoint from those used during training, performance on *uu* assesses the extent to which the distilled uncertainty function generalizes beyond the specific training inputs.

Throughout the experiments, we apply a sensitivity analysis to the p_{drop} and p_{rw} to understand quantitatively the behavior and robustness of the proposed method (note that when $p_{rw} = 0$ QMCD behaves as CMCD). Furthermore, we compare our approach with two SotA baselines: we evaluate the proposed QMCD approach against CMCD [9], and we compare LDMC with BBK [24], a basic teacher–student method, as shown in Figure 1. For each configuration, we repeat the experiment seven times, and report the average result for robustness.

We conduct our experiments on a workstation equipped with an Intel Core i9-14900KF CPU, 64 GB of RAM, and a Nvidia RTX 4090 GPU.

5.2 Uncertainty Estimation with QMCD

We evaluate the QMCD approach on all the three datasets. Hereafter, we report the detailed results of MNIST and summary results of experiments on all three dataset. Results on CIFAR-10 and DBpedia-14 can be found in the Appendix.

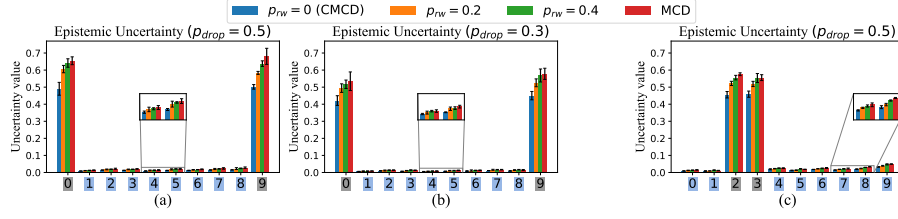


Fig. 5. Epistemic uncertainty per class is shown for the MNIST dataset under different experimental scenarios and dropout rates ($p_{drop} \in \{0.3, 0.5\}$). The colored bars correspond to different p_{rw} values ($p_{rw} \in \{0, 0.2, 0.4\}$) and the MCD baseline. Labels highlighted in gray and blue denote the *unknown* and *known* classes, respectively.

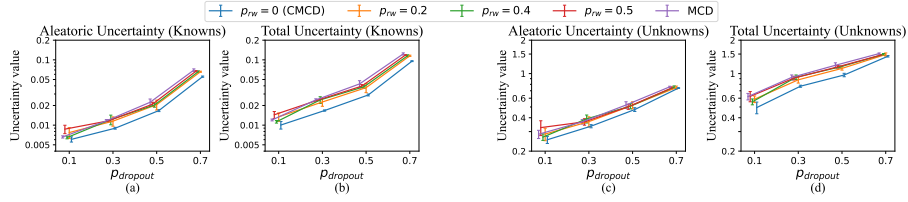


Fig. 6. (a) Average aleatoric uncertainty, (b) average total uncertainty of the *known* classes. (c) Average aleatoric uncertainty, (d) average total uncertainty of the *unknown* classes. The uncertainty value converges to that of MCD value increasing p_{rw} .

In MNIST experiments, we designate digits '0' and '9', respectively, as the *ku* and *uu* classes (the unknown classes), while all the remaining digits treated as *kk*. To assess sensitivity to the choice of unknown classes, we test a second scenario in which digits '2' and '3' are designated, respectively, as *ku* and *uu*. We train an *AlexNet* [17] with $D=2$ dropout layers, we set $M=10$ [9], $R=3$, and $T=128$, we vary $p_{rw} \in \{0, 0.2, 0.5\}$ and $p_{drop} \in \{0.1, 0.3, 0.5, 0.7\}$.

As we can see from Figure 5, for all classes, as we increase p_{rw} , the uncertainty values get closer to the base MCD value for both the epistemic and the total uncertainty. The value of the epistemic uncertainty increases by increasing of p_{rw} , because the model's entropy increases by p_{rw} , and depending on the behavior of the aleatoric uncertainty, which is intrinsic to the data, the total uncertainty can be ascending by p_{rw} . Looking at Figure 6, we can see how the uncertainty values of the *known* and *unknown* classes increase by increasing p_{drop} and p_{rw} . The sensitivity of the model increases with p_{drop} for all classes.

For CIFAR-10, we adopt a *VGG-16* model [23] with $D=3$ dropout layers. For DBpedia-14, we use a *DistilBERT* model [22] in which the dropout layers $D=7$ are replaced with the proposed dropout layers for different values of p and p_{rw} . All other experimental settings follow those used for MNIST.

We report the results for all three datasets at $p_{drop}=0.3$ in Table 1. For each dataset, the leftmost column presents the uncertainty estimated by the MCD baseline, while the remaining columns report the relative differences between the QMCD

Table 1. Epistemic, aleatoric, and total uncertainty values and their relative differences under varying p_{rw} values for MNIST, CIFAR-10, and DBpedia, reported for known (k) and unknown (u) samples.

		MNIST					CIFAR-10				DBpedia			
		Unc. Value	Diff. value %			Unc. Value	Diff. value %			Unc. Value	Diff. value %			
p_{rw} value		$p_{rw} =$					$p_{rw} =$				$p_{rw} =$			
		0	0.2	0.5		0	0.2	0.5		0	0.2	0.5		
Epistemic	k	0.012	38%	13%	2%	0.038	16%	15%	0%	0.025	3%	1%	1%	
	u	0.556	22%	8%	2%	0.116	15%	14%	0%	0.238	-11%	-3%	1%	
Aleatoric	k	0.011	22%	3%	2%	0.077	9%	9%	2%	0.126	-4%	0%	0%	
	u	0.368	9%	0%	0%	0.270	8%	7%	2%	1.243	-11%	-4%	-1%	
Total	k	0.024	31%	8%	2%	0.115	10%	10%	2%	0.152	4%	1%	1%	
	u	0.924	17%	5%	2%	0.386	10%	8%	2%	1.481	-12%	-4%	-1%	

estimates and the MCD values. Across all three uncertainty types, increasing p_{rw} leads, as expected, to closer agreement with the MCD estimates. As shown in Table 1 and detailed Figure 5, QMCD outperforms CMCD [9], corresponding to QMCD with $p_{rw}=0$, consistently on both *known* and *unknown* classes as p_{rw} increases.

5.3 Distillation and Efficient Inference with LDMC

In these experiments we follow the same configurations from the ones in Section 5.2. Moreover, as before, we do the experiments on the three datasets.

In MNIST experiments, we designate digits '0' and '9', respectively, as the *ku* and *uu* classes (the unknown classes), while all the remaining digits treated as *kk*. To assess sensitivity to the choice of unknown classes, we test a second scenario in which digits '2' and '3' are designated, respectively, as *ku* and *uu*. We train an *AlexNet* [17] with $D=2$ dropout layers, we set $M=10$ [9], $R=3$, and $T=128$, we vary $p_{rw} \in \{0, 0.2, 0.5\}$ and $p_{drop} \in \{0.1, 0.3, 0.5, 0.7\}$. As aggregator function to compress the ordered masks for the soft-dropout layer, we use a weighted sum (with learnable weights, one for each of the R masks) followed by a sigmoid action. The CIFAR-10 experiments and the DBpedia-14 experiments follow the structure of MNIST with VGG-16 and DistilBERT, respectively, as we did in the previous experiment. We train student models using LDMC and BBK, as baseline, using the loss function introduced in Equation (11).

Figure 7 shows how a student trained with LDMC achieves lower KL divergence and RMSE losses than BBK. Varying the value of p_{drop} , as shown in Figure 7(b), or modifying the *ku* and *uu* settings, as presented in Figure 7(c), does not alter this behaviour. Results in Figure 8 show how LDMC consistently yields lower loss values than the baseline method BBK across all configurations.

The full results for all three datasets at $p_{drop}=0.3$ and $p_{rw} \in \{0.2, 0.5\}$ are summarized in Table 2. In the majority of cases, LDMC yields lower loss values across all three input categories. For MNIST, LDMC achieves reductions in KL loss of 35%,

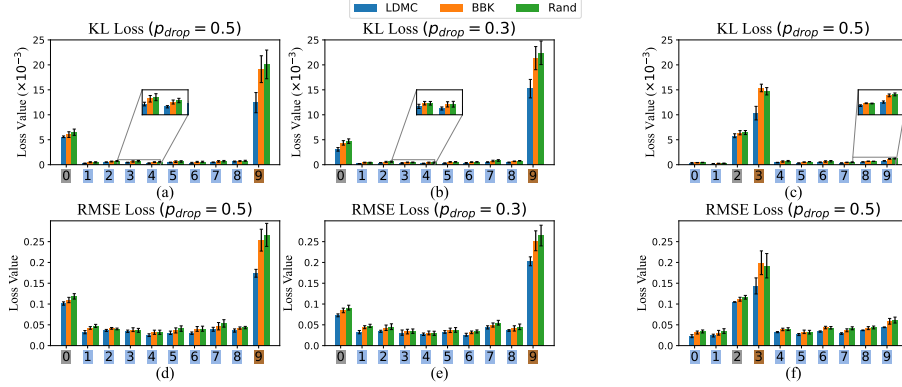


Fig. 7. Student training loss 207/25 different configurations on the MNIST dataset and $p_{rw}=0.2$. (a) KL loss for $p=0.3$ (b) KL loss for $p=0.5$ (c) KL loss for $p=0.5$, ku is '0' and uu is '9'. (d) RMSE loss for $p=0.3$ (e) RMSE loss for $p=0.5$ (f) RMSE loss for $p=0.5$, ku is '2' and uu is '3'.

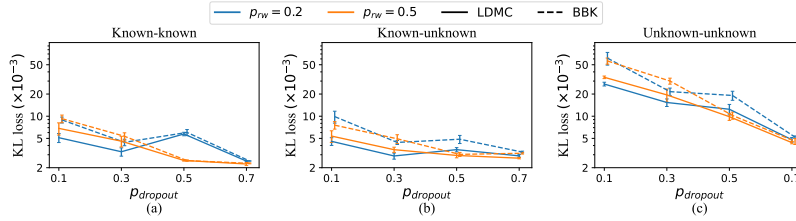


Fig. 8. KL loss comparison between LDMC and BBK method for varying values of p at $p_{rw}=0.2$ and $p_{rw}=0.5$. (a) shows the average loss for the kk classes, (b) the loss for the ku classes, and (c) the loss for the uu class.

27%, and 31% for kk , ku , and uu , respectively, compared to BBK. For DB-Pedia14, these values are -14% , 60% , and 24% , respectively. Overall, the LDMC student achieves nearly the same classification accuracy, for kk data, as the BBK student, and both match or slightly exceed the teacher's accuracy. Although LDMC has a 7–20% increase in inference time relative to BBK, as shown in Table 3, similarly to BBK, LDMC reduces inference cost of a single data by roughly a factor of $1/T$ (for $T=100$) compared to MC-based methods, making this approach effective for real time applications.

5.4 Discussion

With our experiments, we demonstrated that the dropout mask distribution can be compressed at different levels by adjusting the random walk parameter p_{rw} . This compression enables the construction of an information channel through which the student can access the teacher's MC sampling behavior. As a result, the student distills not only the teacher's predictive outputs, but also information

Table 2. Comparison of KL divergence, RMSE loss, and classification accuracy for LDMC and the BBK methods on MNIST, CIFAR-10, and DBpedia with $p_{drop} = 0.3$ under different p_{rw} values.

			MNIST		CIFAR-10		DBpedia	
p_{rw}			LDMC	BBK	LDMC	BBK	LDMC	BBK
KL-loss ($\times 10^{-3}$)	kk	0.2	0.34$\pm$0.05	0.53 $\pm$ 0.07	7.21$\pm$0.35	8.23 $\pm$ 0.54	0.16 $\pm$ 0.04	0.14$\pm$0.03
		0.5	0.41$\pm$0.08	0.59 $\pm$ 0.13	7.85$\pm$0.54	8.15 $\pm$ 1.38	2.13$\pm$0.21	2.52 $\pm$ 0.26
	ku	0.2	3.20$\pm$0.33	4.40 $\pm$ 0.45	22.9$\pm$1.0	23.9 $\pm$ 1.6	0.26$\pm$0.12	0.66 $\pm$ 0.14
		0.5	4.58$\pm$0.75	5.34 $\pm$ 0.65	21.8$\pm$0.8	23.3 $\pm$ 0.9	2.30$\pm$0.28	3.02 $\pm$ 0.31
	uu	0.2	15.3$\pm$1.8	22.0 $\pm$ 2.3	24.5$\pm$1.2	33.6 $\pm$ 3.6	1.09$\pm$0.11	1.44 $\pm$ 0.14
		0.5	19.4$\pm$0.4	30.0 $\pm$ 0.3	25.3$\pm$1.5	32.3 $\pm$ 0.6	3.07$\pm$0.29	4.20 $\pm$ 0.36
RMSE-loss	kk	0.2	0.03$\pm$0.00	0.04 $\pm$ 0.00	0.19$\pm$0.01	0.19 $\pm$ 0.01	0.11$\pm$0.03	0.15 $\pm$ 0.04
		0.5	0.04$\pm$0.00	0.04 $\pm$ 0.00	0.19$\pm$0.00	0.20 $\pm$ 0.01	0.13$\pm$0.02	0.14 $\pm$ 0.02
	ku	0.2	0.07$\pm$0.00	0.09 $\pm$ 0.01	0.24$\pm$0.01	0.33 $\pm$ 0.01	0.14$\pm$0.05	0.20 $\pm$ 0.05
		0.5	0.09$\pm$0.01	0.09 $\pm$ 0.00	0.33$\pm$0.01	0.34 $\pm$ 0.01	0.16$\pm$0.02	0.20 $\pm$ 0.03
	uu	0.2	0.26$\pm$0.01	0.29 $\pm$ 0.02	0.35$\pm$0.02	0.39 $\pm$ 0.02	0.56 $\pm$ 0.06	0.54$\pm$0.04
		0.5	0.27$\pm$0.04	0.30 $\pm$ 0.04	0.37$\pm$0.01	0.37 $\pm$ 0.02	0.33$\pm$0.03	0.42 $\pm$ 0.02
Accuracy(%)	kk	0.2/0.5	99.5/99.5	99.5/99.5	93/93.1	92.8/93	99.3/99.2	98.9/98.9

Table 3. Inference time (ms) of each method across the evaluated datasets.

MNIST			CIFAR-10			DBpedia-14		
MCD	LDMC	BBK	MCD	LDMC	BBK	MCD	LDMC	BBK
23.4 $\pm$ 0.1	0.250 $\pm$ 0.005	0.207 $\pm$ 0.004	75.8 $\pm$ 0.1	0.618 $\pm$ 0.010	0.527 $\pm$ 0.01	298 $\pm$ 3	2.29 $\pm$ 0.02	2.14 $\pm$ 0.05

about the MC sampling process itself, leading to improved uncertainty prediction performance. In our initial experiments, the teacher model was trained with different degrees of mask distribution compression. As compression increases, the uncertainty estimates increasingly deviate from those produced by standard MC sampling, revealing a trade-off between compression and fidelity to the original MC uncertainty. This divergence remains negligible for $p_{rw} > 0.2$.

Using the proposed LDMC framework, as shown in the second experiment, the student consistently outperforms the classical distillation approach. While requiring the student to receive the R base masks from the teacher introduces additional training overhead and slightly increases inference cost due to the inclusion of soft dropout layers, the overall computational burden remains significantly lower than that of full MC sampling. The extended training time limits exhaustive exploration of all combinations of classification settings and hyperparameters such as M and R . This substantial speedup with respect to full MCD combined with the fidelity in uncertainty estimation enables the practical deployment of uncertainty-aware models in time-critical applications.

6 Conclusion

Time-critical applications make uncertainty estimation more challenging by imposing additional constraints on latency while still expecting effective estimates. Although MC-based sampling methods provide this effective uncertainty estimation, their high inference cost limits practical deployment. In this work, we aim to distill the Monte Carlo sampling process into a student model to enable reliable one-shot uncertainty estimation. Our approach achieves improved uncertainty estimation compared to standard teacher-student distillation methods.

Several directions remain for future investigation. Exploring the hyperparameters space may provide insight into the trade-off between uncertainty fidelity and computational efficiency, and alternative mask sampling or distribution compression strategies could improve stability and approximation quality. Future work will also investigate extending the distillation framework to other uncertainty estimation methods, such as Bayesian approaches. Moreover, the proposed method can be integrated into real-time systems, where predictive uncertainty guides adaptive planning and execution. Within this framework, the student model can be updated online, enabling continual refinement and adaptation to distributional shifts in dynamic environments.

Data availability Link: <https://github.com/alirezamlkki/LDMC>

Acknowledgments. This work was supported by funding from the pilot program Core Informatics at KIT (KiKIT) of the Helmholtz Association (HGF) and KASTEL Security Research Labs, Karlsruhe.

Disclosure of Interests. The authors have no competing interests to declare.

References

1. Bendale, A., Boulton, T.E.: Towards open set deep networks. In: Proc. of the IEEE conference on computer vision and pattern recognition. pp. 1563–1572 (2016)
2. Blundell, C., Cornebise, J., Kavukcuoglu, K., Wierstra, D.: Weight uncertainty in neural networks. In: ICML (2015)
3. Chen, M.H., Shao, Q.M., Ibrahim, J.G.: Monte Carlo methods in Bayesian computation. Springer Science & Business Media (2012)
4. Daxberger, E., Kristiadi, A., Immer, A., Eschenhagen, R., Bauer, M., Hennig, P.: Laplace redux-effortless bayesian deep learning. *Advances in neural information processing systems* **34**, 20089–20103 (2021)
5. Deng, L.: The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine* **29**(6), 141–142 (2012)
6. Deuschel, J., Foltyn, A., Roscher, K., Scheele, S.: The role of uncertainty quantification for trustworthy ai. In: *Unlocking artificial intelligence: from theory to applications*, pp. 95–115. Springer (2024)
7. Gal, Y., Ghahramani, Z.: Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: *international conference on machine learning*. pp. 1050–1059 (2016)

8. Gal, Y., et al.: Uncertainty in deep learning, vol. 1. University of Cambridge (2017)
9. Hasan, M., Hossain, I., Rahman, A., Nahavandi, S.: Controlled dropout for uncertainty estimation. In: 2023 IEEE International Conf. on Systems, Man, and Cybernetics (SMC). pp. 973–980 (2023)
10. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: NIPS Deep Learning and Representation Learning Workshop (2015)
11. Hüllermeier, E., Waegeman, W.: Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning* **110**, 457–506 (2021)
12. Jospin, L.V., Laga, H., Boussaid, F., Buntine, W., Bennamoun, M.: Hands-on bayesian neural networks—a tutorial for deep learning users. *IEEE Computational Intelligence Magazine* **17**(2), 29–48 (2022)
13. Jospin, L.V., Laga, H., Boussaid, F., Buntine, W., Bennamoun, M.: Hands-on bayesian neural networks—a tutorial for deep learning users. *IEEE Computational Intelligence Magazine* **17**(2), 29–48 (2022)
14. Kendall, A., Gal, Y.: What uncertainties do we need in bayesian deep learning for computer vision? In: NeurIPS (2017)
15. Korattikara Balan, A., Rathod, V., Murphy, K.P., Welling, M.: Bayesian dark knowledge. *Advances in neural information processing systems* **28** (2015)
16. Krizhevsky, A.: Learning multiple layers of features from tiny images. Tech. rep. (2009)
17. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: *Advances in Neural Information Processing Systems*. vol. 25. Curran Associates, Inc. (2012)
18. Leveson, N.G.: *Engineering a safer world: Systems thinking applied to safety*. MIT press (2016)
19. Nakamura, H., Okada, M., Taniguchi, T.: Representation uncertainty in self-supervised learning as variational inference. In: *Proc. of the IEEE/CVF International Conf. on Computer Vision*. pp. 16484–16493 (2023)
20. Ovadia, Y., et al.: Can you trust your model’s uncertainty? evaluating predictive uncertainty under dataset shift. In: NeurIPS (2019)
21. Rahaman, R., et al.: Uncertainty quantification and deep ensembles. *Advances in neural information processing systems* **34**, 20063–20075 (2021)
22. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In: *Proc. of the 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing (EMC²) at NeurIPS* (2019)
23. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2015)
24. Tedeschini, B.C., Kwon, G., Nicoli, M., Win, M.Z.: Real-time bayesian neural networks for 6g cooperative positioning and tracking. *IEEE Journal on Selected Areas in Communications* **42**(9), 2322–2338 (2024)
25. Tran, L., Veeling, B.S., Roth, K., Swiatkowski, J., Dillon, J.V., Snoek, J., Mandt, S., Salimans, T., Nowozin, S., Jenatton, R.: Hydra: Preserving ensemble diversity for model distillation. *arXiv preprint arXiv:2001.04694* (2020)
26. Urbani, M., Gasparini, G., Brunelli, M.: A numerical comparative study of uncertainty measures in the dempster–shafer evidence theory. *Information Sciences* **639**, 119027 (2023)
27. Wang, K., Shen, C., Li, X., Lu, J.: Uncertainty quantification for safe and reliable autonomous vehicles: A review of methods and applications. *IEEE Transactions on Intelligent Transportation Systems* (2025)
28. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: *Advances in Neural Information Processing Systems* (2015)