Article

Improving Re-Identification by Estimating and Utilizing Diverse Uncertainty Types for Embeddings

Markus Eisenbach *, Andreas Gebhardt, Dustin Aganian and Horst-Michael Gross

Neuroinformatics and Cognitive Robotics Lab, Ilmenau University of Technology, 98693 Ilmenau, Germany; dustin.aganian@tu-ilmenau.de (D.A.); horst-michael.gross@tu-ilmenau.de (H.-M.G.)
* Correspondence: markus.eisenbach@tu-ilmenau.de

Abstract: In most re-identification approaches, embedding vectors are compared to identify the best match for a given query. However, this comparison does not take into account whether the encoded information in the embedding vectors was extracted reliably from the input images. We propose the first attempt that illustrates how all three types of uncertainty, namely model uncertainty (also known as epistemic uncertainty), data uncertainty (also known as aleatoric uncertainty), and distributional uncertainty, can be estimated for embedding vectors. We provide evidence that we do indeed estimate these types of uncertainty, and that each type has its own value for improving re-identification performance. In particular, while the few state-of-the-art approaches that employ uncertainty for re-identification during inference utilize only data uncertainty to improve single-shot re-identification performance, we demonstrate that the estimated model uncertainty vector can be utilized to modify the feature vector. We explore the best method for utilizing the estimated model uncertainty based on the Market-1501 dataset and demonstrate that we are able to further enhance the performance above the already strong baseline UAL. Additionally, we show that the estimated distributional uncertainty resembles the degree to which the current sample is out-of-distribution. To illustrate this, we divide the distractor set of the Market-1501 dataset into four classes, each representing a different degree of out-of-distribution. By computing a score based on the estimated distributional uncertainty vector, we are able to correctly order the four distractor classes and to differentiate them from an in-distribution set to a significant extent.

Keywords: uncertainty estimation for embedding vectors; model uncertainty; data uncertainty; distributional uncertainty; re-identification; out-of-distribution

1. Introduction

The development of self-driving cars [1,2] and robots for collaborative assembly [3,4] represents a significant advancement in autonomous systems. However, these technologies rely on intelligent components to perceive their environment, especially for perception of humans and interaction with them. In the current state of the art, the majority of perception systems are based on neural networks, which, like many machine learning methods, compute outputs for any input, regardless of whether they can make safe decisions or not. Therefore, estimating the uncertainty in the decision-making process is crucial for the development of a reliable autonomous system that can assist humans without causing harm. As a result, uncertainty quantification or estimation for neural networks has emerged as a growing research trend in recent years.

Starting from basic work on uncertainty quantification [5,6], work on uncertainty estimation for neural networks has been mainly focused on classification and regression tasks [7–10]. This includes perception tasks such as object detection [11–14], semantic and medical image segmentation [15–18], as well as action recognition [19], in which actions and objects are classified, and localization is achieved through bounding box regression. The determined uncertainty in these cases relates to the final classification or regression layer.

However, there is currently a trend to use neural networks as encoders to describe images or language as feature vectors, which is known as learning a feature embedding. The estimation of uncertainties for embedding vectors is much less explored [8].

To quantify uncertainties for feature vectors, most state-of-the-art approaches use probabilistic embeddings [20,21], which are able to estimate data uncertainty. Only few approaches [22–25] estimate model uncertainty for feature vectors by using a Bayesian module on top of the deterministic backbone. These are the two most relevant types of uncertainty in literature [26,27]. However, for autonomous agents it is also important to recognize when the data are out-of-distribution, which means that the current input is very dissimilar from data seen during training. Then, it is likely that the agent cannot make safe decisions even if model uncertainty and data uncertainty are low. If such situations are relevant, according to literature, uncertainty can be categorized into three types of uncertainty [7]: (1) model uncertainty (also known as epistemic uncertainty), (2) data uncertainty (also known as aleatoric uncertainty), and (3) distributional uncertainty. Neither probabilistic embeddings nor Bayesian modules are able to quantify distributional uncertainty. Indeed, to the best of our knowledge, the estimation of distributional uncertainty for feature vectors has not been explored at all.

We show that it is beneficial to estimate all three types of uncertainty and how each of the uncertainty types has its individual benefits for predicting different types of error sources for the feature vector. Notably, this information can be used to improve the embedding during training and inference.

In particular, we demonstrate how a component-wise *model uncertainty* estimate can be leveraged to enhance the feature vector during inference. Furthermore, we show that during training estimated *data uncertainty* can be employed to scale the embedding vector elements in accordance with the degree of uncertainty, as proposed by Dou et al. [25]. Additionally, we illustrate how estimated *distributional uncertainty* can be utilized to determine the extent to which an input is out-of-distribution.

While our approach can be used in general to estimate and utilize the three types of uncertainty, for the purposes of benchmarking, we decided in favor of person re-identification as the application. Person re-identification represents an exemplary human perception task that is relevant for autonomous agents that require the ability to track individuals or interact with humans. The selection of person re-identification for benchmarking is consistent with related work on uncertainty estimation for feature vectors [22,23,28], which made the same decision since person re-identification provides well-defined benchmarks while the task is very challenging. Furthermore, it is one of the few domains, where uncertainties for embedding vectors have been employed to enhance decision-making, thereby enabling a comparison of our results with multiple state-of-the-art approaches.

While the state-of-the-art approaches that make use of uncertainties for person re-identification during inference only use data uncertainty to improve the single-shot person re-identification performance, in our experiments, we show that we can use the other two types of uncertainty to further improve the performance over the strong baseline and to quantify the extent to which an input image is out-of-distribution.

Our contributions are as follows:

1. In contrast to related work, we estimate three different types of uncertainty for feature vectors and, based on our experiments, we emphasize the individual value of diverse uncertainty types for improving the embedding and thereby the re-identification.
2. We show that our *distributional uncertainty* estimate enables us to measure the degree to which an input is out-of-distribution, which, based on literature, is a strong indicator that the extracted values actually represent distributional uncertainty.
3. We utilize the estimated *model uncertainty* vector to perform element-wise modifications to the feature vector, and thereby improve the person re-identification performance notably over an already strong baseline.

This paper is organized as follows: In Section 2, we describe the state of the art on uncertainty quantification for feature vectors and the utilization of this uncertainty

information. In Section 3, we describe the mathematics of uncertainty quantification and explain the algorithm for estimating three different types of uncertainty. In Section 4, we evaluate our experiments regarding the usefulness of the estimation and utilization of diverse types of uncertainty. Finally, in Section 5, we draw a conclusion and provide an outlook on future work.

2. Related Work

The majority of related work on uncertainty quantification focuses on classification and regression tasks [7–10]. In contrast, the estimation of uncertainties for embedding vectors is much less explored [8].

In the following, we describe, how uncertainty is quantified for feature vectors (Section 2.1) and how this uncertainty information can be utilized to improve the feature embedding (Section 2.2).

2.1. Uncertainty Quantification for Feature Vectors

In literature, mainly the quantification of data uncertainty (also known as aleatoric or statistical uncertainty) and model uncertainty (also known as epistemic or systemic uncertainty) is considered. Additionally, the quantification of distributional uncertainty has been proposed by Malinin and Gales [26] and Nandy et al. [27].

2.1.1. Quantification of Data Uncertainty

To estimate uncertainties for feature vectors, in related work, most often probabilistic embeddings [21] are used, which are sometimes also referred to as stochastic embeddings [29].

Probabilistic Embeddings for Data Uncertainty Estimation

Probabilistic embeddings were first introduced for natural language processing [20,30] and later, based on the example of face recognition [21,31], also for computer vision tasks. The key idea of probabilistic embeddings is to estimate a mean and a variance vector instead of a point estimate for the feature vector. The variance vector is interpreted as a diagonal covariance matrix. Therefore, each output is represented as a multivariate normal distribution in the embedding space. The mean vector represents the feature vector and the variance vector represents the data uncertainty of the feature vector elements. Probabilistic embeddings have been widely used for computer vision tasks, e.g., for face recognition [21,31–33], few-shot recognition [29], emotion recognition [34], human pose embeddings [35], image retrieval [36–38], 3D shape retrieval [39], point cloud instance segmentation [40], self-supervised and representation learning [41–46], multi-source domain adaptation [47], vision-language embedding alignment [48–50], and person re-identification [25,28,51–54]. In the majority of these approaches, a Gaussian distribution has been used to model the uncertainty. Only few approaches [28,33] present methods for deviating from a Gaussian distribution.

To get a definite-valued output for further processing, most of these works sample from this distribution, which is also the way variational autoencoders (VAEs) [55] handle the distribution in the latent space. Some approaches [21,52,56] use the distributions directly. Dou et al. [25] use the variance information to weight the mean vector.

The estimated uncertainty in probabilistic embeddings represents only data uncertainty. Therefore, these probabilistic embeddings cannot quantify model uncertainty or distributional uncertainty by themselves.

Alternative to Probabilistic Embeddings

To date, only one alternative to probabilistic embeddings has been proposed for the estimation of data uncertainty. Taha et al. [23] propose a method for estimating a kind of data uncertainty, which they then use to compute uncertainty-based weighting factors in the triplet loss. However, the uncertainty they estimate does not correspond to the

mathematical model of data uncertainty. Therefore, probabilistic embeddings represent the only mathematically grounded approach for estimating data uncertainty for feature vectors that has been proposed in the state of the art.

2.1.2. Quantification of Model Uncertainty

Quantifying model uncertainty for feature vectors is addressed rarely. Approaches have only been proposed for image captioning [24] and person re-identification [22,23,25].

Bayesian Neural Networks for Model Uncertainty Estimation

All these approaches [22–25] use Bayesian Neural Networks (BNNs) to estimate the model uncertainty. BNNs replace scalar weights with a distribution over them and then use approaches such as Markov Chain Monte Carlo or Variational Inference to update the distribution such that a good set of weights becomes more and more likely. They are not very popular in deep neural networks, since they suffer from performance problems and are hard to train.

As a compromise to approximate model uncertainty with limited extra computation, applying a Bayesian component on top of a deterministic backbone is a popular choice. Indeed, the re-identification approaches [22,23,25] all use a Bayesian module on top of a deterministic backbone. In this Bayesian module, Monte Carlo Dropout is used to sample K models to estimate the model uncertainty.

Joint Estimation of Data and Model Uncertainty

Only few approaches [23,25] attempt to jointly estimate both data uncertainty and model uncertainty for feature vectors. We use the Uncertainty Aware Learning (UAL) approach [25] as our baseline since it uses a reasonable choice of a Bayesian module to estimate model uncertainty and a mathematically grounded approach to estimate data uncertainty by using a probabilistic embedding.

2.1.3. Quantification of Distributional Uncertainty

To the best of our knowledge, there is no related work that has estimated distributional uncertainty for feature vectors, so far. Basic studies, like the ones of Malinin and Gales [26] and Nandy et al. [27], propose methods for estimating distributional uncertainty for a classification task.

2.2. Utilization of Uncertainties

Estimated uncertainties can be utilized during training to improve the learning process and during inference to filter out unsuitable inputs, to fuse multi-model or multi-shot data, as well as to improve the feature vector.

2.2.1. Utilization of Uncertainty during Training

One thing in common with most works that attempt to estimate uncertainty in some way is that they let their uncertainty estimates flow into the loss function in order to improve learning, hoping to enable the model to learn a better embedding.

In the event that some training data are unsuitable and hinder optimal training progress, it is beneficial to restrict their influence. This can be achieved by incorporating an estimate of data uncertainty. In accordance with this principle, the estimated data uncertainty from a probabilistic embedding is often employed to implicitly enhance the embedded features during training, thereby facilitating an improvement in feature matching [21,25,28,31–33,36–38,51,53,54].

The estimation of model uncertainty with a Bayesian module has also been shown to improve the embedding during training [22,23,25].

2.2.2. Utilization of Uncertainty during Inference

During inference, estimated uncertainty for feature vectors can be used to filter out unsuitable inputs, to fuse multi-model or multi-shot data, and to improve the feature vector.

Utilization of Uncertainty to Filter Out Unsuitable Inputs

Many approaches restrict their utilization of uncertainty to the calculation of a scalar uncertainty quantifier, also known as a quality or reliability score. This scalar uncertainty quantifier is subsequently employed to identify low-quality or out-of-distribution inputs [21,25,26,28,57] in order to filter them out.

One common use case is the visualization of the estimated uncertainty for monitoring purposes. However, the awareness of unsuitable inputs is also a beneficial piece of information for an autonomous agent to recognize that the input is likely different from the training data. This may occur when an object is cropped based on an imprecise bounding box. In such a case, the autonomous agent may attempt to obtain a more accurate representation by capturing the scene from a different perspective, rather than relying on the current input to make a decision. Similarly, in the context of person re-identification, if an autonomous agent tracks its user based on a feature vector and other cues, as in the framework of Müller et al. [58], the uncertainty estimate is a valuable indicator to determine when a feature vector should be excluded from the tracking process, as it may be out-of-distribution.

Utilization of Distributional Uncertainty to Detect Out-of-Distribution Data

For a classification task, Malinin et al. [26] show the value of distributional uncertainty to recognize cases, where the input is far away from the distribution of examples in the training data. This field of out-of-distribution detection has been the subject of much research in recent years [59,60], but with the sole focus on classification tasks.

Utilization of Uncertainty for Feature Vectors to Detect Out-of-Distribution Data

Schwaiger et al. [57] examined whether estimated model uncertainty or data uncertainty for feature vectors can be used to identify out-of-distribution samples. These types of estimated uncertainty were not good enough to reliably detect out-of-distribution data. In our experiments, we confirm that these two types of estimated uncertainty are only somewhat suitable for out-of-distribution detection and that estimated distributional uncertainty is a much better fit.

Utilization of Uncertainty for Multi-Modality, Multi-View, and Multi-Shot

If the goal is to compute a joint embedding for vision and language inputs [48–50], estimated data uncertainty helps to better align the embeddings from different modalities.

To weight different samples for the same instance, data uncertainty and model uncertainty have been used to fuse multiple shots in re-identification [21,25,61], multiple views [62], or multiple modalities [63]. Please note that multi-shot approaches are not part of our examinations. In this paper, we discuss the value of estimated uncertainty for single-shot re-identification.

Utilization of Uncertainty to Improve the Feature Vector

In order to compute the feature vector, Dou et al. [25] divide the estimated mean tensor of the probabilistic embedding by the estimated variance tensor of the probabilistic embedding. Therefore, they include data uncertainty in the computation of the feature vector during inference, which represents an improvement over the baseline.

Other than for multi-shot purposes and filtering of unsuitable data, estimated model uncertainty for feature vectors has not been utilized during inference so far.

3. Uncertainty Estimation in Neural Networks — Mathematics and Algorithm

In deep-learning-based re-identification (ReID), we use a deep neural network (DNN) to map some input $x \in \mathcal{X}$ to an abstract representation $z \in \mathcal{Z}$, known as the embedding, that we can efficiently compare to other embeddings in order to find inputs that are similar to each other. The DNN is thus a function $f_\theta : \mathcal{X} \to \mathcal{Z}$, parameterized by its weights θ. Generally, $\mathcal{Z} = \mathbb{R}^d$ and z is therefore called the d-dimensional feature vector or embedding vector.

In Section 3.1, we introduce the mathematical theory underlying the estimation of uncertainty for feature vectors. In Section 3.2, we present the motivation for extracting different types of uncertainty. In Section 3.3, we propose an architecture for the extraction of three types of uncertainty. In Section 3.4, we discuss the alignment between the estimated uncertainty in our architecture and the mathematical theory.

3.1. Mathematical Theory on Modeling Uncertainty

As detailed in the analysis conducted by Gawlikowski et al. [7], there are numerous points throughout the preparation, design, and training of a DNN where errors may occur and uncertainty arises. This uncertainty is subsequently reflected in the predictive uncertainty, which is modeled as a probability distribution $p(z)$ over the model outputs, specifically the embedding vector in this case. In order to gain further insight into this somewhat vague concept, it is helpful to decompose this general uncertainty into sub-types, based on its source.

3.1.1. Distinction of Model, Data, and Distributional Uncertainty

Model uncertainty is inherent to the model and is caused by problems with the model structure or training process, such as insufficient model complexity, training duration, or dataset composition. Data uncertainty is inherent to the data and is caused by information loss in the measuring process, such as limited image resolution, motion blur, or labeling errors. Distributional uncertainty is intended to cover the concept of out-of-distribution (OOD) inputs.

Even when a model is perfectly tuned for its task and the data is flawless (low model and data uncertainty), the predictions may still be unreliable if the model is applied to inputs it was not designed to handle (high distributional uncertainty). For example, if a person ReID model is provided with images of dogs, bikes, or text, unexpected results may be produced. Figure 1 illustrates the role of *distributional uncertainty* in contrast to the other uncertainty types.
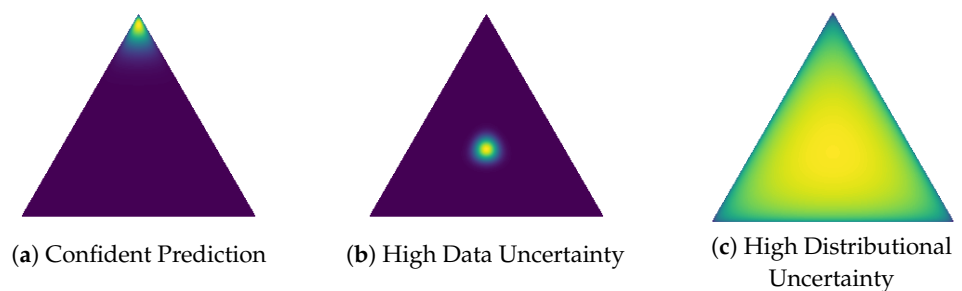

(a) Confident Prediction **(b)** High Data Uncertainty **(c)** High Distributional Uncertainty

Figure 1. Example for a Dirichlet distribution with three classes in three scenarios. Points on the simplex correspond to a parameter configuration (p_1, p_2, p_3) of the categorical distribution ($p(i) = p_i$). Yellow (blue) corresponds to a high (low) probability for the configuration. (**a**) The configuration $p_1 \approx 1$, $p_2, p_3 \approx 0$ is most likely. (**b**) The configuration $p_1 \approx p_2 \approx p_3$ is most likely. (**c**) All configurations are equally likely. Image source: Malinin and Gales [26], similar visualization used by Gawlikowski et al. [7].

In the case of high data uncertainty and low distributional uncertainty (Figure 1b), we know that we do not know which class is presented, whereas in the case of high distributional uncertainty (Figure 1c), we do not know whether we know and thus should not trust any seemingly certain prediction of category or level of data uncertainty we might

get. One could say that distributional uncertainty arises due to the interaction between data and model, specifically due to a mismatch between them. In addition to model uncertainty and data uncertainty, we will also consider distributional uncertainty in this work.

3.1.2. Mathematical Model

A universally accepted mathematical model for uncertainty in neural networks has yet to be established, and it is possible that one may never emerge. This is because, as Kiureghian et al. [64] argue, the specific model may be context-dependent. Malinin and Gales [26] adapt the more common model proposed by Gal and Ghahramani [65] to include distributional uncertainty, and arrive at Equation (1). Their approach is classification-based, and thus the predictive uncertainty is modeled as a distribution over the class prediction $\hat{y}$ instead of the embedding vector.

$$\underbrace{p(\hat{y}\,|\,x,\mathcal{D})}_{\text{Uncertainty}} = \int \int \underbrace{p(\hat{y}\,|\,\xi)}_{\substack{\text{Data}\\\text{Uncertainty}}} \cdot \underbrace{p(\xi\,|\,x,\theta)}_{\substack{\text{Distributional}\\\text{Uncertainty}}} \cdot \underbrace{p(\theta\,|\,\mathcal{D})}_{\substack{\text{Model}\\\text{Uncertainty}}} \,\mathrm{d}\xi\mathrm{d}\theta \tag{1}$$

Here, x is the input, $\mathcal{D}$ is the training dataset, and θ represents the model parameters. The Dirichlet distribution $p(\xi\,|\,x,\theta)$ is a distribution over the categorical distribution $p(\hat{y}\,|\,\xi)$, where ξ acts as a stand-in for the distribution $p(\hat{y}\,|\,\xi)$ and can in practice be thought of as its parameters. This formulation is not ideally suited to embedding-based tasks, as it models uncertainty based on the class prediction $\hat{y}$. Consequently, we adapt Equation (1) to be based in the embedding space, resulting in Equation (2), where z is the embedding vector as defined above.

$$\underbrace{p(z\,|\,x,\mathcal{D})}_{\text{Uncertainty}} = \int \int \underbrace{p(z\,|\,\xi)}_{\substack{\text{Data}\\\text{Uncertainty}}} \cdot \underbrace{p(\xi\,|\,x,\theta)}_{\substack{\text{Distributional}\\\text{Uncertainty}}} \cdot \underbrace{p(\theta\,|\,\mathcal{D})}_{\substack{\text{Model}\\\text{Uncertainty}}} \,\mathrm{d}\xi\mathrm{d}\theta \tag{2}$$

Now, data uncertainty is modeled as a distribution in the embedding space, distributional uncertainty as a distribution over those distributions (i.e., their parameters), and model uncertainty as a distribution over the model parameters. This is consistent with the modelling approach employed in previous works that utilize probabilistic embeddings, i.e., distributions in the embedding space, to capture data uncertainty [20,21,25,28–54] (see Section 2.1.1), as well as Bayesian Neural Networks, in which distributions are placed over the model parameters, to capture model uncertainty [22–25] (see Section 2.1.2).

Note that the color scheme for data , distributional , and model uncertainty, as well as the feature vector is used throughout this work to further indicate what is being displayed.

3.2. Value of Quantified Uncertainty

The mathematical theory offers insight into the distinction between three types of uncertainty. This subsection aims to provide readers with a comprehensive understanding of the individual value of these three types of uncertainty. Readers who are already familiar with this topic may choose to skip this subsection.

The most straightforward value of quantified uncertainty is the measure of the extent to which the output of the model can be trusted. Such information can be provided to decision-makers, users, subsequent processing modules in autonomous agents, or to the model itself, with the objective of improving the learned representation during training. The deployment of uncertainty as a metric for trustworthiness is of particular significance in scenarios where critical decisions must be made, as misjudgments can result in substantial consequences, including damage or injury. One such example is the interaction between autonomous agents and humans. When high uncertainty is observed, it indicates *that* the system's output should not be trusted. However, the reason *why*, such predictions

should not be trusted is not immediately apparent. To uncover the underlying reason, it is necessary to decompose the uncertainty into different types, which we will discuss below.

3.2.1. Individual Value of Model Uncertainty

The value of quantified model uncertainty lies in the acquisition of information regarding the competence of the model with respect to the task at hand. In particular, it indicates how likely it is that the model makes errors when operating in laboratory conditions.

If the model uncertainty indicates that the model is deficient in its ability to perform the task at hand, we may attempt to improve the model. This may be achieved, for instance, by providing additional training data for scenarios in which the model is uncertain. Moreover, model uncertainty may serve as a valuable indicator of the model's convergence state during training. If the model has not converged on some of the training data, an estimate of model uncertainty can be utilized to regulate the extent to which the model should learn from a particular set of training examples.

Model uncertainty is also the foundation for distributional uncertainty. If the model is not adequately tuned to its task, we should not rely on its capacity to accurately predict when a given situation aligns with this task. This is also evident in the mathematical model (Equation (2)) where the distributional uncertainty term is conditioned upon the model uncertainty term.

3.2.2. Individual Value of Distributional Uncertainty

The value of quantified distributional uncertainty lies in the acquisition of information regarding the expected competence of the model in a given situation. Therefore, distributional uncertainty measures the extent to which the model's expertise aligns with the problem it is attempting to solve in the current situation.

If distributional uncertainty indicates a limitation in the model's ability to address the current situation, the input can be delegated to an alternative model or a human operator. Alternatively, this information can be provided to any subsequent processing modules of an autonomous agent that depend on the accuracy of the model output.

Distributional uncertainty also serves as a foundation for estimating data uncertainty. If the model's expertise does not match the current situation, it is not reasonable to rely on it to assess the quality of the data. This is also reflected in the mathematical model (Equation (2)) where the data uncertainty term is conditioned upon the distributional uncertainty term.

3.2.3. Individual Value of Data Uncertainty

The value of quantified data uncertainty lies in the acquisition of information regarding the theoretical limits of prediction accuracy. Even if the model were perfectly constructed for its task and if the current situation perfectly matched that task, it might still be impossible to make certain predictions if the data on which they are based is flawed.

In such cases, data uncertainty indicates that the data does not contain sufficient information to make a precise prediction. In these instances, it is possible to gather better data or otherwise make a nuanced prediction, specifying multiple possibilities that could be true. Furthermore, if data uncertainty is modeled as a probabilistic embedding, the inherent uncertainty in the data, such as missing details due to resolution or motion blur, can be transferred into the embedding space.

In practice, the boundary between data uncertainty and distributional uncertainty can be quite fuzzy. An image of a person that has been wrongly cropped, displaying only an arm, could be perceived as exhibiting high data uncertainty due to the erroneous setting of the crop. However, it could also be regarded as exhibiting high distributional uncertainty, given that images of arms belong to a distinct domain from images of people. Similarly, the introduction of minor amounts of noise into an image would be regarded as data uncertainty. However, if the noise is so extensive that the original image is no longer recognizable, this would represent an instance of the domain of images of noise, which would be indicated by high distributional uncertainty.

In summary, the information obtained allows us to identify the sources of uncertainty. We can distinguish whether the data are ambiguous (data uncertainty), the model is unable to cope with the current situation (distributional uncertainty), or the model is underperforming even under ideal conditions (model uncertainty).

3.3. Uncertainty Estimation for Embedding Vectors

The basis for our architecture to estimate the three types of uncertainty is the Uncertainty Aware Learning (UAL) approach [25]. The structure of our architecture is given in Figure 2 (detail) and Figure 3 (overview).

As shown in Figure 3, the basic feature extraction is performed by the backbone f, which processes the input $x \in \mathbb{R}^{3 \times 256 \times 128}$ and extracts a feature tensor $F \in \mathbb{R}^{d \times 16 \times 8}$ (Equation (3)). Therefore, as a result of the downsampling operations in the backbone, the height and width of the feature map are reduced to $\frac{1}{16}$ of the height and width of the input image (see Figure 3).

$$F = f(x) \tag{3}$$

The backbone is a ResNet-50 based on the BOT approach [66], which is pre-trained on ImageNet. In this case, the number of channels in the feature tensor F is $d = 2048$. The feature tensor F is then processed by dedicated modules—the Bayesian module and the embedding heads—to estimate the three types of uncertainty in addition to the feature vector.

The Bayesian module enables the estimation of *model uncertainty*. The embedding heads realize a probabilistic embedding, which enables the estimation of *data uncertainty*. The combination of the Bayesian module and the variance head enables the estimation of *distributional uncertainty*. All modules for uncertainty estimation are initialized randomly. Their structure is detailed in Figure 2, and details about their realization are presented below.
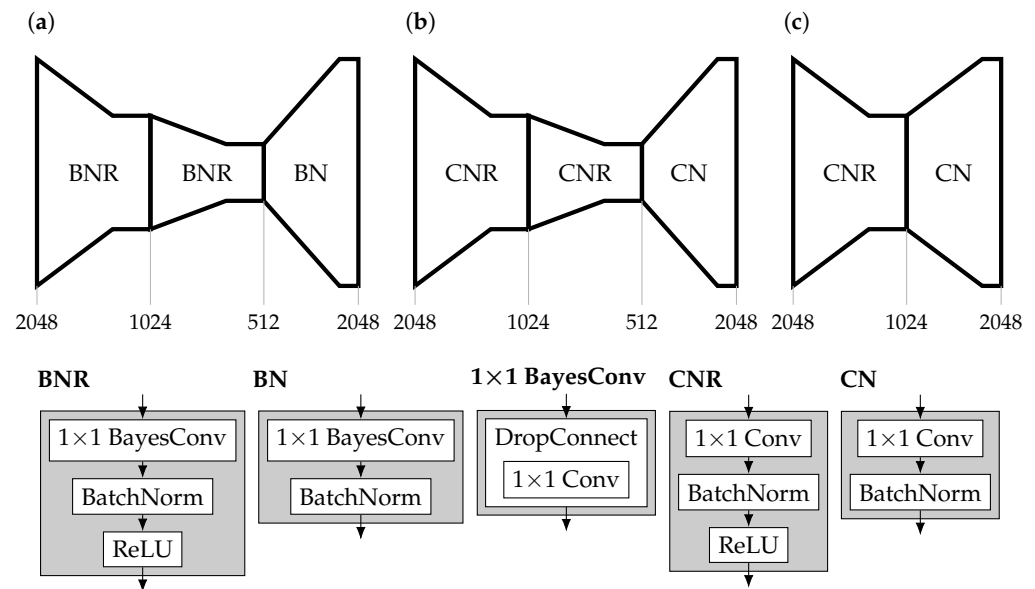

Figure 2. Structure of (**a**) the Bayesian module, (**b**) the mean head, and (**c**) the variance head (top row) and of their building blocks (bottom row). Underneath the diagrams in (**a**–**c**), the number of channels at each stage is indicated. Abbreviations: B—Bayes convolution, C—standard convolution, N—batch normalization, R—ReLU.
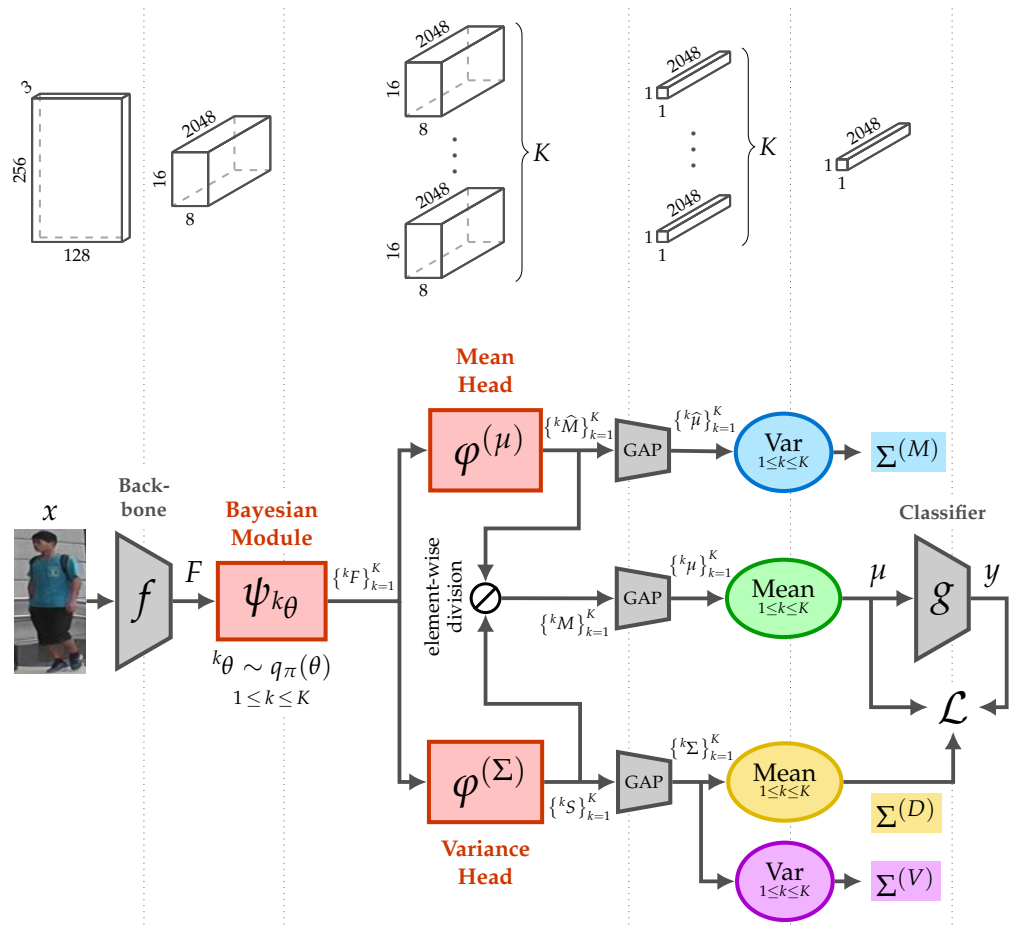
Figure 3. Overview of our architecture to estimate the three types of uncertainty for an embedding. First, the backbone f produces a feature tensor F from the input x. Next, the parameters $^{k}\theta$ of the Bayesian module are sampled K times from the weight distribution $q_{\pi}(\theta)$, thus producing K versions of the Bayesian module, each of which processes the feature tensor F, producing a total of K resulting feature tensors $\left\{^{k}F\right\}_{k=1}^{K}$. Each of these resulting feature tensors ^{k}F is processed separately by the variance head $\varphi^{(\Sigma)}$, yielding the variance features ^{k}S, and by the mean head $\varphi^{(\mu)}$, yielding the raw mean features $^{k}\widehat{M}$, which are divided element-wise by the variance features to obtain the mean features ^{k}M. The mean and variance operations shown as ellipses are applied element-wise across the K versions of the raw mean vector $^{k}\widehat{\mu}$, the mean vector $^{k}\mu$, and the variance vector $^{k}\Sigma$, all of which are obtained through global average pooling (GAP) from their precursors. Computing the variance of the raw mean vector in this way yields the estimate for model uncertainty $\Sigma^{(M)}$. The mean of the mean vector yields the feature vector μ. The mean of the variance vector yields the estimate for data uncertainty $\Sigma^{(D)}$. And finally, the *variance* of the *variance* vector yields the estimate for distributional uncertainty $\Sigma^{(V)}$. Together, the feature vector and the data uncertainty estimate define the embedding distribution $\mathcal{N}(\mu, \Sigma)$, where $\Sigma = \mathrm{diag}(\Sigma^{(D)})$. During training, we additionally need a classifier g to map the feature vector μ to a categorical output y, which encodes different individuals in the training dataset. The feature vector μ, the data uncertainty estimate $\Sigma^{(D)}$, and the logits y, all contribute to the Loss $\mathcal{L}$. The shape of all named Tensors is sketched above the architecture for the respective sections. Mathematical and algorithmic details are explained in Section 3.3.

3.3.1. Bayesian Module

The first of these modules is the Bayesian module $\psi_{k\theta}$, which acts as a pre-processing unit for the two embedding heads and enables the estimation of model uncertainty in the original approach as well as the estimation of distributional uncertainty with our

modification. It is essentially a three-layered bottleneck block (Figure 2a), whose weights $^k\theta$ are sampled K times for each input, as described in Algorithm 1. The resulting K versions of the module each process the output of the backbone F to compute a feature tensor $^kF \in \mathbb{R}^{d \times 16 \times 8}$ of the same size (Equation (4)).

$$^kF = \psi_{k\theta}(F), \quad k \in \{1, \ldots, K\} \tag{4}$$

The weight sampling can be more easily understood as dropping each entry of the full weight tensors $\widehat{\theta}$, which are learned by ordinary backpropagation, to 0 with probability $p = 0.3$, which is known as DropConnect. The number of weight samples K is 1 during training but is increased to 30 during inference which then allows the uncertainty estimation. This is identical to the hyperparameters used by Dou et al. [25].

Algorithm 1: Bayesian Module

   Input: Extracted feature tensor F according to Equation (3)
               Weight tensors $\widehat{\theta}$
               Number of weight samples K
               Drop-probability p for DropConnect

Output: Output features of the Bayesian module $\left\{^kF\right\}_{k=1}^K$

1 **for** $k \in \{1, \ldots, K\}$ **do**
2     $^k\theta \leftarrow \widehat{\theta}$ `// Initialize `$^k\theta$` with weights of Bayesian module`
       `// `$^k\theta$` contains the weights of several tensors`
       `// but is viewed as vector here for simplicity`
3   **for** $^k\theta_i \in {}^k\theta$ **do**
4   Sample $\beta \sim$ Bernoulli$(1 - p)$
5         $^k\theta_i \leftarrow \beta \cdot {}^k\theta_i$ `// Apply DropConnect`
6     $^kF \leftarrow \psi_{k\theta}(F)$ `// Apply Bayesian module using sampled weights`
7 **return** $\left\{^kF\right\}_{k=1}^K$

Discussion of the Number of Weight Samples during Training

In preliminary experiments, we investigated the effect of using more than one weight sample during both training and inference. In this case, we consistently observed a small decrease in the re-identification quality.

Why is a single weight sample beneficial during training? In the case of $K = 1$ during training, the model is required to produce an accurate embedding distribution based on the single weight configuration it has access to. The model must learn to produce these, for any given weight perturbation it may encounter. However, in the case of $K > 1$ during training, the prediction of the embedding distribution results from the average over the output for all K weight configurations. There is no requirement that any individual output is perfect or fit any particular weight configuration perfectly. It is sufficient for the resulting average to be a good prediction. Consequently, the quality of the output for each single weight configuration can decline, or become less well-adapted.

3.3.2. Embedding Heads

Like the Bayesian module, the mean and variance heads $\varphi^{(\mu)}, \varphi^{(\Sigma)}$ are also bottleneck-blocks but without any weight-sampling (see Figure 2b,c). They are both applied to each of the K versions of the resulting feature tensor ^kF separately which then consequently results

in K versions of the raw mean features ${}^{k}\widehat{M} \in \mathbb{R}^{d \times 16 \times 8}$ (Equation (5)) and the variance features ${}^{k}S \in \mathbb{R}^{d \times 16 \times 8}$ (Equation (6)).

$$
{}^{k}\widehat{M} = \varphi^{(\mu)}\left({}^{k}F\right) \tag{5}
$$

$$
{}^{k}S = \varphi^{(\Sigma)}\left({}^{k}F\right) \tag{6}
$$

The raw mean features ${}^{k}\widehat{M}$ and the variance features ${}^{k}S$ are then utilized according to Equation (7) by applying element-wise division ($\oslash$) to obtain the mean features ${}^{k}M \in \mathbb{R}^{d \times 16 \times 8}$. This represents the first manner in which our approach employs the uncertainty values to enhance the embedding. In contrast to most approaches, this application of the data uncertainty estimate is also employed during inference.

$$
{}^{k}M = {}^{k}\widehat{M} \oslash {}^{k}S \tag{7}
$$

The subsequent Global Average Pooling (GAP) is applied to each of the K versions of the raw mean features ${}^{k}\widehat{M}$, the mean features ${}^{k}M$, and the variance features ${}^{k}S$ separately, collapsing the width and height dimensions. This results in K versions of each of the raw mean vector ${}^{k}\widehat{\mu} \in \mathbb{R}^{d}$ (Equation (8)), the mean vector ${}^{k}\mu \in \mathbb{R}^{d}$ (Equation (9)), and the variance vector ${}^{k}\Sigma \in \mathbb{R}^{d}$ (Equation (10)).

$$
{}^{k}\widehat{\mu} = \text{GAP}\left({}^{k}\widehat{M}\right) \tag{8}
$$

$$
{}^{k}\mu = \text{GAP}\left({}^{k}M\right) \tag{9}
$$

$$
{}^{k}\Sigma = \text{GAP}\left({}^{k}S\right) \tag{10}
$$

In order to obtain the final mean vector $\mu \in \mathbb{R}^{d}$, the element-wise mean is computed across the K versions of the mean vector ${}^{k}\mu$ (Equation (11)). Here, $\mathbb{E}$ refers to the expected value of the indicated values.

$$
\mu_{i} = \mathbb{E}\left\{{}^{k}\mu_{i}\right\}_{k=1}^{K} \tag{11}
$$

By utilizing the Bayesian module and the embedding heads, the three types of uncertainty can be estimated, which will be explained in detail below.

3.3.3. Estimation of Model Uncertainty

The model uncertainty estimate $\Sigma^{(M)} \in \mathbb{R}^{d}$ is calculated as the element-wise variance over the K versions of the raw mean vector ${}^{k}\widehat{\mu}$ (Equation (12)). This is identical to the method proposed by Dou et al. [25].

$$
\Sigma_{i}^{(M)} = \mathbb{V}\text{ar}\left\{{}^{k}\widehat{\mu}_{i}\right\}_{k=1}^{K} \tag{12}
$$

Here, $\mathbb{V}\text{ar}$ refers to the variance of the indicated values. This formulation does not align particularly well with the mathematical theory in Equation (2). Most notably, this estimate is not independent of the input as the term $p(\theta \mid \mathcal{D})$ from Equation (2) would suggest the model uncertainty estimate should be. Instead, we propose that $\Sigma^{(M)}$ measures the effect of model uncertainty on the feature vector. A more detailed discussion is given in Section 3.4.

3.3.4. Estimation of Data Uncertainty

Data uncertainty is described in the mathematical model (Equation (2)) as a distribution $p(z \mid \xi)$ over the embedding space. As is common with probabilistic embeddings, our architecture approximates this distribution as a Gaussian $p(z \mid \xi) \approx \mathcal{N}(\mu, \Sigma)$ by estimating its parameters μ and Σ. The final data uncertainty estimate $\Sigma^{(D)} \in \mathbb{R}^{d}$ is obtained

by computing the element-wise mean across the K versions of the variance vector ${}^k\Sigma$ (Equation (13)).

$$\Sigma_i^{(D)} = \mathbb{E}\left\{{}^k\Sigma_i\right\}_{k=1}^{K} \tag{13}$$

$$\Sigma = \text{diag}\left(\Sigma^{(D)}\right) \tag{14}$$

The diagonal covariance matrix $\Sigma \in \mathbb{R}^{d \times d}$ of the multivariate normal distribution is obtained by using $\Sigma^{(D)}$ as the diagonal of the matrix (Equation (14)).

The reason for the model's ability to encode data uncertainty into the variance vector ${}^k\Sigma$ can be found in the loss function (Section 3.3.7). In essence, the model is encouraged to assign higher uncertainty values to instances where the classification is bad. During training, the model learns to distribute these higher values towards vector positions that are primarily responsible for the inferior classification outcome.

3.3.5. Estimation of Distributional Uncertainty

Our proposed distributional uncertainty estimate $\Sigma^{(V)} \in \mathbb{R}^d$ is calculated as the element-wise <u>v</u>ariance over the K versions of the <u>v</u>ariance vector ${}^k\Sigma$ (Equation (15)).

$$\Sigma_i^{(V)} = \mathbb{V}\text{ar}\left\{{}^k\Sigma_i\right\}_{k=1}^{K} \tag{15}$$

In the mathematical model (Equation (2)), distributional uncertainty resides in the distribution $p(\xi \mid x, \theta)$ over the embedding distribution $p(z \mid \xi)$. Since we approximated $p(z \mid \xi)$ as a Gaussian $\mathcal{N}(\mu, \Sigma)$ which is solely defined by its parameters, we can say that $\xi = (\mu, \Sigma)$ in our case. Hence, $p(\xi \mid x, \theta)$ becomes $p((\mu, \Sigma) \mid x, \theta)$.

The reason for excluding μ here and focusing only on $p(\Sigma \mid x, \theta)$ is that the variability in the raw mean vector $\widehat{\mu}$ is already utilized as model uncertainty (see Section 3.3.3). The mean vector μ is merely a result of the element-wise division of mean features by variance features (see Equation (7)), so considering it does not offer any new information. Furthermore, this approach ensures consistency in the shape of our uncertainty estimates as vectors of the same length as the embedding vector. A detailed investigation that supports the hypothesis that the vector $\Sigma^{(V)}$ actually measures distributional uncertainty is provided in Section 4.4.

3.3.6. Obtaining a Scalar Uncertainty Quantifier

In certain situations, it is very helpful to have a detailed uncertainty estimate which matches some other component's shape whose uncertainty it describes. We have already seen this in Equation (7) where we use the variance feature tensor to modify the mean feature tensor and will explore a similar strategy in Section 4.3. However, in other situations, it would be more helpful to have a simple and straightforward scalar uncertainty quantifier. This can be the case if the task is to define a cutoff for unacceptably high uncertainty or if the uncertainty values should be provided for human interpretation. Thus we need to choose a function to convert our uncertainty estimates $\Sigma^{(D)}, \Sigma^{(M)}, \Sigma^{(V)} \in \mathbb{R}^d$ to a scalar uncertainty score $s \in \mathbb{R}$.

Given that the data uncertainty estimate is employed as the diagonal of a covariance matrix in our approach and that the other two uncertainty vectors are also calculated as variances, it appears reasonable to consider the distributions to which these variances pertain and to utilize tools of probability theory in order to quantify the uncertainty inherent in these distributions. In particular, entropy is commonly used for this purpose. This is because the entropy H of some random variable A is defined as the expected surprisal. Naturally, we consider the entropy of the probabilistic embedding $\mathcal{N}(\mu, \text{diag}(\Sigma^{(D)}))$ for data uncertainty. Analogously, we consider the entropy of $\mathcal{N}(\widehat{\mu}, \text{diag}(\Sigma^{(M)}))$ and $\mathcal{N}(\Sigma^{(D)}, \text{diag}(\Sigma^{(V)}))$ for model and distributional uncertainty. The entropy of a multivariate normal distribution is given in Equation (16).

$$H(A) = \frac{d}{2} \ln (2\pi e) + \frac{1}{2} \ln \det \Sigma, \qquad\qquad A \sim \mathcal{N}(\mu, \Sigma) \qquad (16)$$

Because this formula only depends on Σ, and because we are working with diagonal covariance matrices whose diagonals are the uncertainty vectors, this simplifies to Equation (17) which describes the function $\mathcal{H} : \mathbb{R}^d \to \mathbb{R}$ that maps the uncertainty vector to the entropy of the respective distribution.

$$\mathcal{H}\left(\Sigma^{(\omega)}\right) = \frac{d}{2} \ln (2\pi e) + \frac{1}{2} \sum_{i=1}^{d} \ln\left(\Sigma_i^{(\omega)}\right), \qquad\qquad \omega \in \{M, D, V\} \qquad (17)$$

We will use this measure in our experiments where a scalar uncertainty value is needed, i.e., for correlation analyses (Sections 4.2 and 4.3.2) and for the utilization of uncertainty to predict the extent to which an input is out-of-distribution (Section 4.4).

3.3.7. Loss

As detailed in Equation (18), the loss function consists of the classification and data uncertainty loss $\mathcal{L}_{\text{C\&DU}}$, Triplet Hard Loss $\mathcal{L}_{\text{TriH}}$ [67], and a weight decay term $\lambda \cdot \mathcal{L}_2^{\text{reg}}$.

$$\mathcal{L} = \mathcal{L}_{\text{C\&DU}} + \mathcal{L}_{\text{TriH}} + \lambda \cdot \mathcal{L}_2^{\text{reg}} \qquad (18)$$

$$\mathcal{L}_{\text{C\&DU}} = \frac{1}{\sigma} \cdot \mathcal{L}_{\text{CE}} + \ln \sigma, \qquad\qquad \text{with } \sigma = \frac{1}{d}\left\|\Sigma^{(D)}\right\|_1 \qquad (19)$$

The classification and data uncertainty loss $\mathcal{L}_{\text{C\&DU}}$, as defined in Equation (19), is essentially a weighted cross entropy loss ($\mathcal{L}_{\text{CE}}$). The size of the logit vector $y = g(\mu)$, $y \in \mathbb{R}^T$, which is needed to compute the cross entropy loss $\mathcal{L}_{\text{CE}}$, depends on the number T of individuals in the training dataset.

The cross-entropy loss is multiplied by the inverse of the data uncertainty term ($\frac{1}{\sigma} \cdot \mathcal{L}_{\text{CE}}$). As very small uncertainty values would result in an increased loss, the model is forced to learn appropriate uncertainty values greater than zero. Furthermore, large uncertainty values are encouraged in cases where the classification is poor. As a consequence, a trivial solution to minimize this part of the loss would be reached by assigning very large uncertainty values regardless of the input. Thus, we employ the logarithmic addition of the uncertainty term σ as a penalizing term.

The application of weight decay and batch normalization layers supports this limiting influence by ensuring that weights and activations remain relatively small, including the activations utilized as uncertainty values.

In addition to these effects, which only regulate the magnitude of the uncertainty values, the information propagated back from the cross entropy and triplet loss terms ensure that the uncertainty values are relevant to the feature vector entries of the same position and work as correction terms utilized via element-wise division as intended. This is because the embedding vector that serves as a basis for these loss terms is itself a result of said element-wise division.

The classification and data uncertainty loss $\mathcal{L}_{\text{C\&DU}}$ represents the second way in which our approach utilizes the uncertainty values to improve the embedding. The third and final application of the estimated uncertainty values will be presented in Section 4.3.4.

3.3.8. Training

The objective of person re-identification (ReID) is to train a model that enables the comparison of images depicting a single person each, with the goal of identifying images that depict the same subject. Each image is cropped around the subject, such that the subject is centered in the image. The comparison is conducted based on a learned feature embedding, such that each input image x can be described by a d-dimensional feature vector $z \in \mathbb{R}^d$.

Learning a Feature Embedding

As described in Figure 3 above, in our architecture, we compute the feature vector $z = \mu$. In the training phase, batch normalization followed by a fully connected layer and a Softmax function are applied to the feature vector z. These serve as a classification layer, and thus map the feature vector z to a classification output y. For the purpose of the classification task, every person in the training dataset is treated as a single category. Cross-entropy serves as classification loss. During the training phase, batches are sampled such that they contain several images per individual. Therefore, each training batch contains both matching and non-matching image pairs. Given that the classification layer comprises a limited number of parameters, the model is compelled to learn a robust feature embedding to facilitate accurate classification. In addition to the classification task, triplets comprising an anchor image, a matching image, and a non-matching image present within the training batch are employed as input for the triplet loss function. All other loss terms in Equation (18) serve the purpose of forcing the model to learn adequate uncertainties. After training, the classification layer is no longer required and thus discarded.

Benchmarking

For benchmarking, a set of Q query images $\mathcal{X}^{\mathcal{Q}}$ (Equation (20)) is compared to a set of G gallery images $\mathcal{X}^{\mathcal{G}}$ (Equation (21)).

$$\mathcal{X}^{\mathcal{Q}} = \left\{ x_1^{\mathcal{Q}}, x_2^{\mathcal{Q}}, \ldots, x_Q^{\mathcal{Q}} \right\} \tag{20}$$

$$\mathcal{X}^{\mathcal{G}} = \left\{ x_1^{\mathcal{G}}, x_2^{\mathcal{G}}, \ldots, x_G^{\mathcal{G}} \right\} \tag{21}$$

The objective of ReID is to identify those gallery images that depict the same individual as each of the query images. Therefore, each image $x_i^{\mathcal{Q}}$ of the query set $\mathcal{X}^{\mathcal{Q}}$ is described by a feature vector $z_i^{\mathcal{Q}} \in \mathbb{R}^d$. All of the query feature vectors are then combined to form a matrix $Z^{\mathcal{Q}} \in \mathbb{R}^{Q \times d}$ (Equation (22)). Similarly, the gallery images $\mathcal{X}^{\mathcal{G}}$ are each described by a feature vector $z_i^{\mathcal{G}} \in \mathbb{R}^d$. All of the gallery feature vectors are then combined to form a matrix $Z^{\mathcal{G}} \in \mathbb{R}^{G \times d}$ (Equation (23)).

$$Z^{\mathcal{Q}} = \begin{bmatrix} z_1^{\mathcal{Q}} & z_2^{\mathcal{Q}} & \ldots & z_Q^{\mathcal{Q}} \end{bmatrix}^T \tag{22}$$

$$Z^{\mathcal{G}} = \begin{bmatrix} z_1^{\mathcal{G}} & z_2^{\mathcal{G}} & \ldots & z_G^{\mathcal{G}} \end{bmatrix}^T \tag{23}$$

Based on the query features $Z^{\mathcal{Q}}$ and gallery features $Z^{\mathcal{G}}$, a distance matrix $D \in \mathbb{R}^{Q \times G}$ is computed. This is accomplished by applying a given distance function $\delta : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, which can be used to compare two vectors (Equation (24)). In our case, we use the cosine distance δ_{cos} (Equation (25)).

$$D_{i,j}(Z^{\mathcal{Q}}, Z^{\mathcal{G}}) = \delta(z_i^{\mathcal{Q}}, z_j^{\mathcal{G}}) \tag{24}$$

$$\delta_{cos}(z_i^{\mathcal{Q}}, z_j^{\mathcal{G}}) = 1 - \frac{z_i^{\mathcal{Q}} \cdot z_j^{\mathcal{G}}}{\left\| z_i^{\mathcal{Q}} \right\|_2 \cdot \left\| z_j^{\mathcal{G}} \right\|_2} \tag{25}$$

For the sake of simplicity, we define the operator $\mathfrak{d}_{cos} : \mathbb{R}^{Q \times d} \times \mathbb{R}^{G \times d} \to \mathbb{R}^{Q \times G}$ such that it compares all query feature vectors $Z^{\mathcal{Q}}$ and gallery feature vectors $Z^{\mathcal{G}}$ to compute the distance matrix D using the cosine distance δ_{cos} for comparing the feature vectors.

The distance matrix D is then used to compute rankings for each query. These rankings serve as the basis for evaluating the quality of the re-identification approach.

3.4. Alignment of the Estimation of the Three Uncertainty Types with the Mathematical Theory

Above, we described the mathematical theory for three types of uncertainty and their estimation in our architecture. Below, we discuss how well the estimated uncertainty vectors align with the mathematical theory.

3.4.1. Model Uncertainty

Whether or not the estimated uncertainty vector $\Sigma^{(M)}$ is an appropriate measure of model uncertainty is debatable. According to the model uncertainty term in Equation (2), model uncertainty is only conditioned upon the training dataset. Thus, it should be data-independent.

On the one hand, in our architecture, we follow the interpretation of [25], who refer to the estimated vector $\Sigma^{(M)}$ as the model uncertainty. The estimated uncertainty vector $\Sigma^{(M)}$ is a measure of the variability introduced by the Bayesian Module and is mostly independent of the data uncertainty branch of the network. Therefore, it captures variability information that is independent of the other two uncertainty estimates. This is somewhat in accordance with the mathematical model.

On the other hand, the estimated uncertainty vector $\Sigma^{(M)}$ changes with each input during inference, although the model stays the same. Certainly, this does not align with the mathematical term in Equation (2).

Therefore, we should rather interpret the estimated uncertainty vector $\Sigma^{(M)}$ as the influence of model uncertainty on the current feature vector, which depends on the current input data. In our experiments, we show, that this estimate $\Sigma^{(M)}$ is an important piece of information that can be utilized to improve the feature vector.

3.4.2. Data Uncertainty

The estimation of data uncertainty in our architecture closely aligns with the mathematical theory. The loss function forces the model to learn to estimate when predictions are uncertain based on the current input data.

3.4.3. Distributional Uncertainty

The estimation of distributional uncertainty in our architecture also closely aligns with the mathematical theory.

We compute the distributional uncertainty estimate as the variance of the data uncertainty estimate. However, we could also interpret the distributional uncertainty term in Equation (2) as the distribution over the embedding distribution. Since we have a probabilistic embedding, it could be interpreted as the distribution over both the feature vector μ and the variance vector $\Sigma^{(D)}$. Following this interpretation, both $\Sigma^{(M)}$ and $\Sigma^{(V)}$ should be considered part of the distributional uncertainty estimate.

In our experiments, we show, that $\Sigma^{(V)}$ computed as the <u>v</u>ariance of the <u>v</u>ariance vector can be considered a good estimate of distributional uncertainty. This hypothesis is supported by the observation that the estimated distributional uncertainty $\Sigma^{(V)}$ is a very good fit to predict which data are out-of-distribution, which is the described property of distributional uncertainty in literature [7,26,27]. In contrast, the estimated uncertainty vector $\Sigma^{(M)}$, which we consider the model uncertainty estimate, is much less suitable to predict which data are out-of-distribution.

4. Experiments

To explore the utilization of estimated uncertainties, we conducted several experiments. In Section 4.1, we describe the experimental setup. In Section 4.2, we examine the correlation between uncertainty estimates of the three different uncertainty types. In Section 4.3, we explore the utilization of estimated model uncertainty to improve the feature vector. And finally, in Section 4.4, we explore the utilization of estimated distributional uncertainty to predict the extent to which an input image is out-of-distribution.

4.1. Experimental Setup

In order to benchmark our model and the estimated uncertainties, we decided in favor of person re-identification as the application. This application is a common choice in the state of the art on uncertainty estimation for feature vectors [22,23,28]. Person re-identification provides well-defined benchmarks while the task is very challenging. To enable a comparison of our results with multiple state-of-the-art approaches, we utilize the Market-1501 dataset [68] (see Section 4.1.1), which is the most frequently used within this context.

We report the mean average precision (mAP) and the rank-1 statistic of the cumulative match characteristic (CMC) curve, which are the most commonly utilized measures of re-identification quality [68]. To get meaningful results, we repeated all experiments 20 times. Therefore, we report the mean and standard deviation for each quality measure. It is important to note that the rank-1 statistic is primarily designed to assess the re-identification ability to identify the best matches to each query in the gallery, which are easiest to find. In contrast, the mAP assesses the quality of the entire ranking. Given that uncertainties contribute to enhanced performance in challenging cases, we anticipate that the mAP will prove a more effective measure. Consequently, the mAP results will be the primary focus of discussion.

In some cases, we need to select a specific model for visualization purposes. In these cases, out of the 20 trained models, we decided in favor of the model whose performance closely matches the mAP reported by Dou et al. [25], which is the basis for our architecture. The mAP reported by Dou et al. [25] is 87.0, while our selected model achieves an mAP of 87.03 when the re-identification is conducted identically to the UAL approach [25].

4.1.1. Dataset

We utilize the Market-1501 dataset [68] for the training and evaluation of our models. It is commonly employed in the state of the art as a benchmark for the person re-identification task. The Market-1501 dataset contains 12,936 training images of $T = 751$ individuals. Its test partition contains 750 individuals and is split into $Q = 3368$ query images and $G = 15{,}913$ gallery images. The training set and the query set consist of images with a width-to-height ratio of 1:2, which have been cropped tightly around the depicted individual. Each image depicts a person centered within the frame. Most gallery images also follow this configuration. However, the gallery set also contains 2798 distractor images that deviate from this configuration. The distractors are further analyzed in Section 4.4.

4.1.2. Implementation Details

The basis for our implementation is the Bag of Tricks (BOT) approach [66] implemented in the framework FastReID [69]. BOT uses a slightly modified ResNet-50 [70] and several data augmentation techniques to establish a strong baseline for person re-identification. The modifications include the removal of reduction of resolution at one place in the ResNet and the addition of batch normalization between the extracted feature vector and the classification layer during training. The data augmentation techniques include scaling of the input images to a resolution of 256×128 pixels, Random Erasing [71] with probability $p = 0.5$, random horizontal flipping with probability $p = 0.5$, and padding by 10 pixels followed by random cropping of size 256×128. As in the UAL approach [25], which serves as the basis for our architecture, we employ the modified ResNet model as the backbone of our architecture and also include the batch normalization before the classification layer. As a result of this choice of backbone, the dimensionality of the feature vectors, as well as the uncertainty vectors, is $d = 2048$.

The training pipeline follows the one of the UAL approach [25]. We train our model for 120 epochs by utilizing the Adam optimizer [72] with a basic learning rate of $\eta = 0.00035$. The learning rate scheduling includes a warm-up phase for 2000 iterations, which are approximately 10 epochs, with a linear increase from 0.1η to η. Furthermore, the learning rate is reduced by a factor of 0.1 at epochs 40 and 90. The training batches are constructed by

first selecting eight individuals at random and then four random images for each individual. Therefore, the batch size is 64. Our loss consists of the classification and data uncertainty loss $\mathcal{L}_{\text{C\&DU}}$ and the Triplet Hard Loss $\mathcal{L}_{\text{TriH}}$ with a margin of 0.3. As regularization, we use weight decay with a strength of $\lambda = 0.0005$ and label smoothing with a smoothing factor $\alpha = 0.1$.

For further implementation details, we refer to the source code that is publicly available Source code available at https://www.tu-ilmenau.de/neurob/data-sets-code/uncertainties-for-embeddings (accessed on 22 August 2024).

4.2. Analysis of Correlation between Uncertainty Types

In this section, we examine the degree of independence between the three types of estimated uncertainty. This should determine whether the uncertainty types are correlated, if they are conditioned upon each other, or whether they are independent due to the extraction at different positions in the neural network. If the latter were to be the case, this would be in contradiction with the assumption that the estimated uncertainties follow mathematical theory. In accordance with mathematical theory, we expect that two types of uncertainty estimates will be correlated to some degree if the term of one type of uncertainty in Equation (2) is conditioned upon the term of another uncertainty type. From Equation (2), we know that the distributional uncertainty term is conditioned upon the model uncertainty term, and the data uncertainty term is conditioned upon the distributional uncertainty term. Therefore, we expect that a correlation will be observed between the model uncertainty estimate and the distributional uncertainty estimate. Furthermore, we also expect that the data uncertainty estimate and the distributional uncertainty estimate will be correlated to some degree. Conversely, we expect only a low correlation between the model uncertainty estimate and the data uncertainty estimate.

In order to investigate these relationships, it is necessary to compare the three types of uncertainties for identical inputs. Accordingly, for the three types of uncertainties, we extracted estimated uncertainty vectors for the query and gallery partitions of the Market-1501 dataset [68]. Then, we derived scalar uncertainty values from the estimated uncertainty vectors by computing the entropy, as discussed in Section 3.3.6. Figure 4a–c shows pairwise joint distributions for each pair of uncertainty types. Furthermore, a linear fit was applied to each plot to generate a trend line, and the respective correlation is reported in the plots. The visualized contour lines of the joint distribution are generated using Kernel Density Estimation (KDE).

Additionally, Figure 4d shows the contour lines for different subsets of the Market-1501 dataset. The separability of these in-distribution and out-of-distribution sets by utilizing either data uncertainty or distributional uncertainty is investigated in Section 4.4.

As expected, Figure 4a illustrates that the model uncertainty estimate and the data uncertainty estimate only have a very low correlation. Furthermore, the outcome of the comparison between the distributional uncertainty estimate and the data uncertainty estimate, as illustrated in Figure 4b, aligns with our expectations. The same is true for the results of the comparison of the distributional uncertainty estimate and the data uncertainty estimate, as illustrated in Figure 4c. The correlation between the distributional uncertainty estimate and the data uncertainty estimate is 0.7485, while the correlation between the distributional uncertainty estimate and the model uncertainty estimate is 0.6938. Given that both correlation coefficients are around 0.7 or more, we can consider these pairs of uncertainty estimates to be highly correlated. Therefore, the results of the pairwise correlation analyses for the three types of estimated uncertainty provide support for the hypothesis that the estimated uncertainties are in accordance with the mathematical theory.
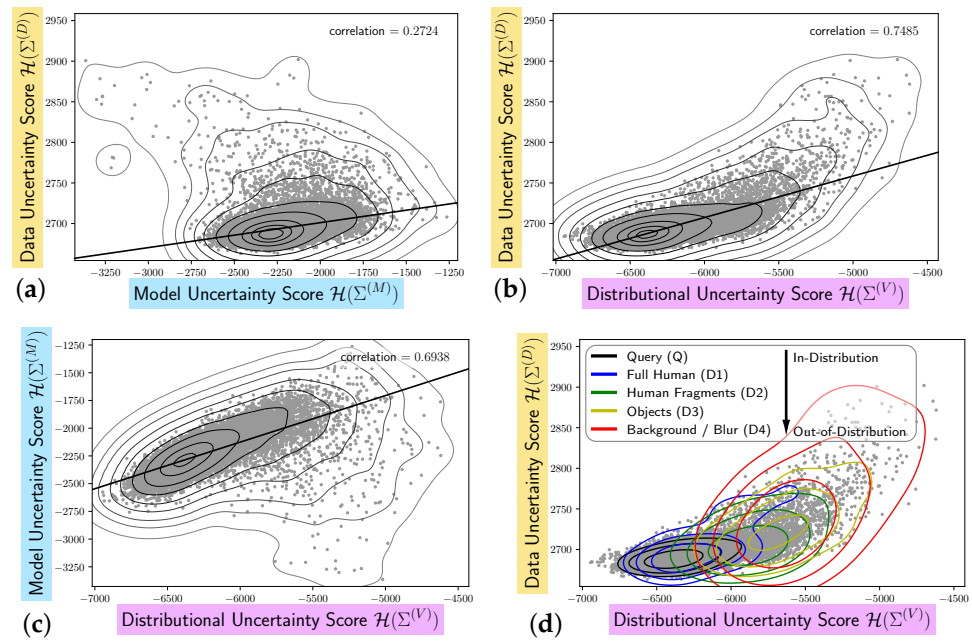
Figure 4. Correlation between the data, model, and distributional uncertainty estimates (**a**–**c**). Each point represents a pair of uncertainty estimates for a sample of the set of query and gallery images of the Market-1501 dataset [68]. Additionally, in (**d**), the contour lines of the distributions of the query and distractor sets are displayed. The distractor sets (D1)–(D4) are explained in Section 4.4.2. The colors are identical to those used in that section.

4.3. Utilization of Model Uncertainty during Inference

In this section, we examine how the model uncertainty estimate can be utilized during inference to refine the feature vector such that the re-identification quality improves. First, in Section 4.3.1, we report the results of related work and then rank our best result in comparison to them. Then, we derive step-by-step how model uncertainty should be utilized to refine the feature vector. Therefore, in Section 4.3.2, we derive why the model uncertainty estimate is suitable at all for refining the feature vector. In Section 4.3.3, we show that some simple solutions for incorporating model uncertainty into the process are not effective. In Section 4.3.4, we derive an appropriate technique for utilizing estimated model uncertainty in order to refine the feature vector, which consequently leads to an improved re-identification performance.

4.3.1. Comparison with the State of the Art

In Table 1, we list re-identification results of approaches that employ uncertainty estimation. The training and evaluation dataset was Market-1501 [68]. The results were either reported in related work [22,28,52] or are the results of our experiments.

It should be noted that the baseline (displayed in gray) in each of the three aforementioned publications differed. The results reported by Taha et al. [22] are based on the DenseNet baseline AWTL [73]. The results reported by An et al. [28] are based on a ResNet-50 baseline and the reported mean and standard deviation were calculated over 5 runs. The results reported by Zhang et al. [52] are based on the ResNet-50 baseline IDE [74]. It is therefore evident that attaining an improvement is considerably easier when one starts from a relatively weak baseline.

We follow the attempt of Dou et al. [25] and establish the strong BOT baseline [66] as a starting point. We report the average result and the standard deviation over 20 runs for all approaches with publicly available source code by the authors, namely Distribution Net (DistNet) [51], Probabilistic Face Embeddings (PFE) [21], and Uncertainty Aware Learning (UAL) [25]. For each approach, we list which types of uncertainty are estimated

during training and which types of uncertainties are utilized during inference for refining the feature vector. In our approach, we utilize the estimated model uncertainty $\Sigma^{(M)}$ and the estimated data uncertainty $\Sigma^{(D)}$ to refine the feature embedding vector μ. Therefore, we name our approach uncertainty-based embedding refinement (UBER). Furthermore, the incorporation of distributional uncertainty into the refinement of the feature embedding vector yields additional minor improvements (Section 4.3.4) and can be used for identifying input images that are out-of-distribution (Section 4.4).

Table 1. Comparison with state-of-the-art re-identification approaches that employ uncertainty estimation. The benchmarking dataset was Market-1501 [68]. For each approach, the types of uncertainties employed are listed.

| | | | | Uncertainty | | | | | |
| | Results | | | Training | | | Inference | | |
Approach	Reported In	mAP [%]	rank-1 [%]	M	D	V	M	D	V
AWTL baseline [73]	[22]	71.45	84.89	✗	✗	✗	✗	✗	✗
Bayesian Retrieval [22]	[22]	72.19	85.87	✓	✗	✗	✗	✗	✗
ResNet-50 baseline	[28]	75.65 ± 0.21	90.48 ± 0.45	✗	✗	✗	✗	✗	✗
DistNet [51]	[28]	74.61 ± 0.06	90.10 ± 0.29	✗	✓	✗	✗	✗	✗
PFE [21]	[28]	76.49 ± 0.23	90.48 ± 0.51	✗	✓	✗	✗	✗	✗
DUL [31]	[28]	77.12 ± 0.19	90.09 ± 0.17	✗	✓	✗	✗	✗	✗
MEIB [28]	[28]	79.67 ± 0.15	92.14 ± 0.16	✗	✓	✗	✗	✗	✗
IDE baseline [74]	[52]	78.4	89.7	✗	✗	✗	✗	✗	✗
LPP [52]	[52]	80.5	91.2	✗	✓	✗	✗	✗	✗
BOT baseline [66]		86.42 ± 0.09	94.53 ± 0.21	✗	✗	✗	✗	✗	✗
DistNet [51]		83.95 ± 0.20	93.44 ± 0.28	✗	✓	✗	✗	✗	✗
PFE [21]		86.42 ± 0.09	94.45 ± 0.20	✗	✓	✗	✗	✗	✗
UAL [25]		86.77 ± 0.14	94.64 ± 0.21	✓	✓	✗	✗	✓	✗
UBER (ours)		$\mathbf{87.21 \pm 0.17}$	$\mathbf{94.67 \pm 0.18}$	✓	✓	✓	✓	✓	✓

Additionally to the mean and standard deviation reported in Table 1, the results of all 20 runs of our experiments are shown in Figure 5 as box plots.
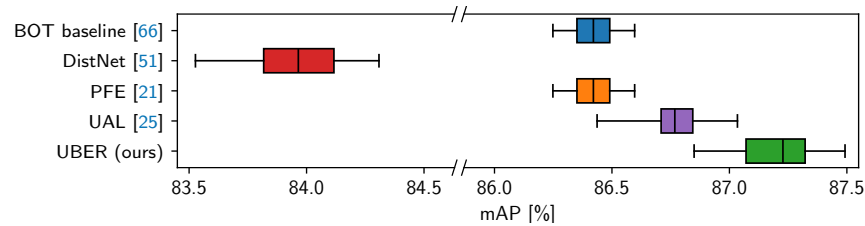

Figure 5. Comparison with state-of-the-art re-identification approaches that employ uncertainty estimation as box plots over the results of 20 runs based on the BOT baseline [66]. The benchmarking dataset was Market-1501 [68].

We can observe, that the performance of DistNet [51] declines significantly when it is applied to a strong baseline. This is consistent with the reported result by An et al. [28] (see Table 1). PFE [21] does not modify the feature vector during training. We observed that the learned uncertainty vector had very small values that barely affected the performance across various attempts to use them for embedding refinement. Therefore the result of PFE is basically identical to the baseline. UAL [25], which is the basis for our architecture, estimated uncertainties that helped to improve the performance over the strong baseline. The reported performance by Dou et al. [25] (mAP = 87.0, rank-1 = 95.2) could be reproduced in our experiments, although it is probably a cherry-picked result. Our approach UBER utilizes model uncertainty and data uncertainty. This helped to further improve the performance noticeably over the UAL result. 80% of our models trained in the

20 runs outperform the best of the 20 trained UAL models. The mean improvement of our approach UBER over UAL is higher than the improvement of UAL over the already strong baseline BOT. The highest mAP score achieved by the best of our trained UBER models across the 20 runs using an individually tailored constant *c* (see Section 4.3.4) was 87.52.

Below, a detailed explanation of the methodology employed to incorporate uncertainties into the refinement of the feature vector is provided, along with a justification for the selection of model uncertainty as the optimal type of uncertainty to be applied.

4.3.2. Relationship between the Feature Vector and Estimated Model Uncertainty

In order to explore the potential of incorporating estimated uncertainties during inference in order to improve the feature vectors, we investigated the relationship between the estimated uncertainty vectors and the feature vector. With regard to the model uncertainty estimate, we observed a correlation between the uncertainty values and the absolute values of the feature vector.

Figure 6a,c visualizes exemplary feature vectors and the associated model uncertainty values. Figure 6b,d shows the correlation.
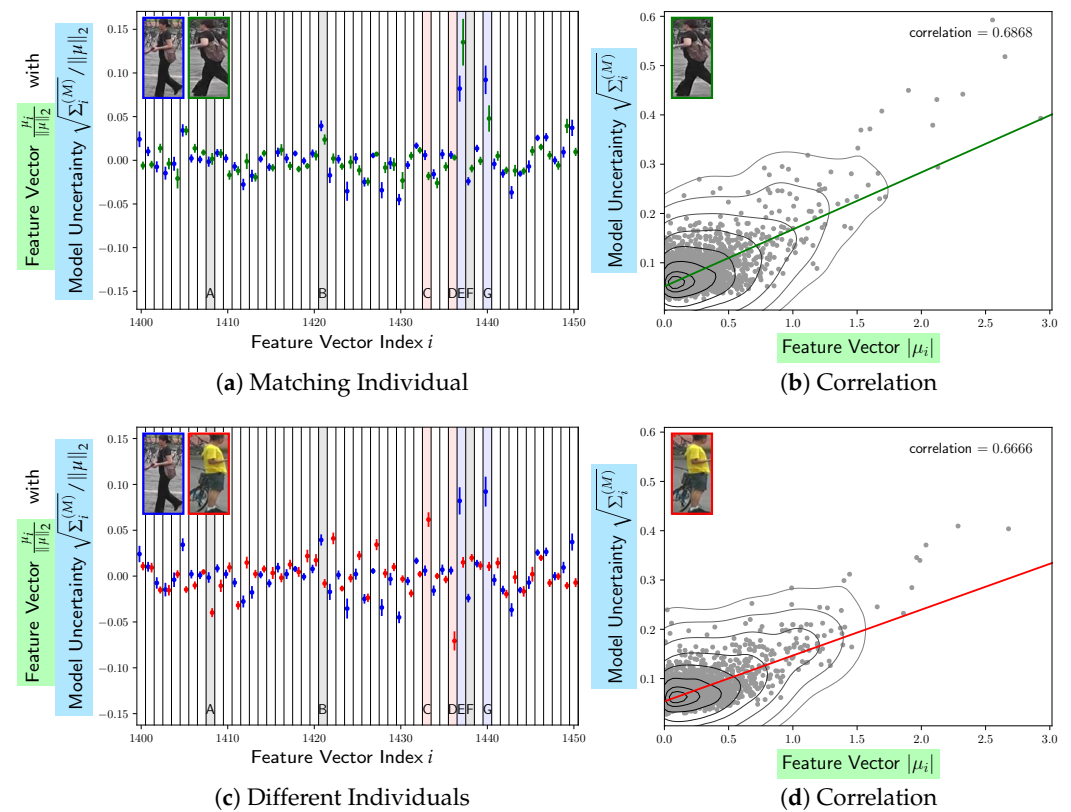

Figure 6. Excerpt of the normalized feature vector (elements 1400–1450, $d = 2048$) for (**a**) two images of the same identity and (**c**) two images of different identities. The corresponding model uncertainty is shown as error bars representing the standard deviation. The highlighted positions marked as A–G are analyzed in the text. The correlation of the absolute values of the feature vector and the model uncertainty is shown for two input images in (**b**,**d**). The images are part of the Market-1501 dataset [68].

In Figure 6a,c an excerpt of feature vectors for three input images is presented. The dots represent the values of the normalized feature vector $\frac{\mu}{\|\mu\|_2}$, which is the basis for calculating the cosine distance. The error bars illustrate the estimated model uncertainty values as standard deviation.

In case of low estimated model uncertainty values for all three feature vectors, it can be observed that at specific positions, such as those highlighted as A, B, and F, we can see that the feature vectors exhibit relatively low differences for a matching individual (a)

and higher differences for different individuals (b). These differences are then aggregated, resulting in a low distance between the feature vectors of matching individuals and a higher distance between the feature vectors of different individuals.

However, the values of the feature vector at the highlighted positions E and G attain high absolute values for the images framed in blue and green of matching identity. Concurrently, the estimated model uncertainty is also relatively large at these locations. It is evident that there are large differences between the feature vectors of the two images depicting the same individual (a) at the highlighted positions. If the estimated model uncertainty is not taken into account, the differences would result in a greater distance between the feature vectors. Hence, the images would be evaluated as depicting individuals with notable dissimilarities.

For the feature vector corresponding to the non-matching image framed in red, we can also observe larger absolute values at the highlighted positions C and D. However, the estimated model uncertainty is much less pronounced than in the case described above.

Therefore, we hypothesize that the estimated model uncertainty is a valuable source of information to judge whether some outliers can be trusted or whether they are the result of uncertain model outputs.

Figure 6b,d illustrate the correlation between the elements of the estimated model uncertainty vector and the absolute values of the feature vector for two exemplary input images. Once more, the outliers can be primarily attributed to large model uncertainty. For the majority of feature vector elements, the estimated model uncertainty is relatively low.

4.3.3. Simple Solutions Prove to Be Ineffective

Before we describe the optimal method for integrating uncertainty information to enhance the re-identification performance, we first demonstrate that simple incorporation strategies are ineffective.

Replacement of the Feature Vector by the Estimated Uncertainty Vector

In light of the observed correlation in Figure 6b,d, one could hypothesize that the estimated model uncertainty may contain the same information as the feature vector. Therefore, we conducted an initial experiment to determine the extent to which re-identification results would be affected if the feature vector were replaced with the estimated model uncertainty vector. Furthermore, given that the three types of estimated uncertainty are also correlated (see Section 4.2), we also investigated the impact of replacing the feature vector with the estimated data uncertainty vector and the estimated distributional uncertainty vector.

For given query feature vectors $\mu^{\mathcal{Q}} \in \mathbb{R}^{Q \times d}$ and gallery feature vectors $\mu^{\mathcal{G}} \in \mathbb{R}^{G \times d}$, the conventional computation of the distance matrix D^{μ}, which serves as the basis for calculating the rankings and subsequently the re-identification performance, is described in Equation (26).

$$D^{\mu} = \mathfrak{d}_{cos}(\mu^{\mathcal{Q}}, \mu^{\mathcal{G}}) \tag{26}$$

This computation of the distance matrix can also be applied when the feature vector is replaced by an uncertainty vector. For given query uncertainty vector estimates $\Sigma^{(\omega),\mathcal{Q}} \in \mathbb{R}^{Q \times d}$, $\omega \in \{M, D, V\}$ and gallery uncertainty vector estimates $\Sigma^{(\omega),\mathcal{G}} \in \mathbb{R}^{G \times d}$, the distance matrix $D^{\Sigma^{(\omega)}}$ can be calculated as in Equation (27) with the variance used as the basis for the calculation. Similarly, the distance matrix $D^{\sqrt{\Sigma^{(\omega)}}}$, which employs the standard deviation as the basis, can be calculated as in Equation (28). In the case of the standard deviation, the square root is applied element-wise to the estimated uncertainty vectors.

$$D^{\Sigma^{(\omega)}} = \mathfrak{d}_{cos}(\Sigma^{(\omega),\mathcal{Q}}, \Sigma^{(\omega),\mathcal{G}}), \qquad \omega \in \{M, D, V\} \tag{27}$$

$$D^{\sqrt{\Sigma^{(\omega)}}} = \mathfrak{d}_{cos}\left(\sqrt{\Sigma^{(\omega),\mathcal{Q}}}, \sqrt{\Sigma^{(\omega),\mathcal{G}}}\right) \tag{28}$$

The results for replacing the feature vectors with the estimated uncertainty vectors are reported in Table 2.

Table 2. Validation: Can uncertainty vectors be utilized as a replacement for the feature vectors?

Distance Matrix		mAP [%]	rank-1 [%]
BOT baseline [66]		86.41 ± 0.09	94.42 ± 0.23
D^{μ} (identical to UAL [25])		86.77 ± 0.14	94.64 ± 0.21
$D^{\sqrt{\Sigma^{(M)}}}$		73.53 ± 0.52	91.05 ± 0.35
$D^{\Sigma^{(M)}}$	Uncertainty	53.50 ± 1.42	78.55 ± 1.31
$D^{\sqrt{\Sigma^{(D)}}}$	vector replaces	41.25 ± 1.15	70.49 ± 1.05
$D^{\Sigma^{(D)}}$	feature	41.17 ± 1.14	70.39 ± 1.08
$D^{\sqrt{\Sigma^{(V)}}}$	vector	2.45 ± 0.45	2.79 ± 0.86
$D^{\Sigma^{(V)}}$		2.09 ± 0.36	2.21 ± 0.68

We observe that when the standard deviation of the model uncertainty estimate was utilized to replace the feature vector ($D^{\sqrt{\Sigma^{(M)}}}$), we unexpectedly obtained a relatively good result. Similarly, when the feature vector was replaced with the variance of the model uncertainty estimate ($D^{\Sigma^{(M)}}$) or the standard deviation or variance of the data uncertainty estimate ($D^{\sqrt{\Sigma^{(D)}}}$, $D^{\Sigma^{(D)}}$), the resulting outcome deviated significantly from the expected random outcome. In contrast, the distributional uncertainty estimates ($D^{\sqrt{\Sigma^{(V)}}}$, $D^{\Sigma^{(V)}}$) are unsuitable as a replacement. Thus, we can conclude, that the model uncertainty estimate and the data uncertainty estimate contain some information that is also present in the feature vector. However, even the best result achieved with the standard deviation of the model uncertainty estimate ($D^{\sqrt{\Sigma^{(M)}}}$) is significantly inferior to the results achieved with the feature vector (D^{μ}).

Simple Score-Level Fusion

In light of the unexpectedly good re-identification performance of the standard deviation of the model uncertainty estimate ($D^{\sqrt{\Sigma^{(M)}}}$) as a replacement for the feature vector, one could hypothesize that combining the re-identification results could yield further improvements. To validate this hypothesis, we conducted a second experiment to determine whether simple score-level fusion yields improvements. The distance scores for each pair of query and gallery images are stored in the distance matrices. In this case, the matrices are D^{μ} for the feature-vector-based scores and $D^{\sqrt{\Sigma^{(M)}}}$ for the uncertainty-based scores. Therefore, the distance matrices can be combined element-wise.

For score-level fusion, we considered an additive combination (Equation (29)) and a multiplicative combination (Equation (30)) of the distance scores, where $\odot$ symbolizes an element-wise multiplication.

$$D^{+M} = D^{\mu} + D^{\sqrt{\Sigma^{(M)}}} \tag{29}$$

$$D^{\odot M} = 1 - (1 - D^{\mu}) \odot \left(1 - D^{\sqrt{\Sigma^{(M)}}}\right) \tag{30}$$

Table 3 shows the score-level fusion results. The additive combination (D^{+M}) does not result in an improved re-identification performance. Furthermore, the multiplicative combination ($D^{\odot M}$) offers only a marginal improvement over the sole feature-vector-based re-identification (D^{μ}). Therefore, simple score-level fusion cannot achieve notable improvements. To this end, it is essential to identify a more effective approach to incorporate the model uncertainty estimate.

Table 3. Validation: Can simple score-level fusion of the model-uncertainty-based re-identification and feature-vector-based re-identification yield improvements?

Distance Matrix	mAP [%]	rank-1 [%]
BOT baseline [66]	86.41 ± 0.09	94.42 ± 0.23
D^μ (identical to UAL [25])	86.77 ± 0.14	94.64 ± 0.21
$D^{\odot M}$	86.80 ± 0.14	94.72 ± 0.18
D^{+M}	86.74 ± 0.14	94.73 ± 0.18

4.3.4. Utilizing the Estimated Model Uncertainty to Adjust Feature Vectors

Besides the combination of distance matrices, we can also consider adjusting the feature vector by utilizing the model uncertainty in order to achieve an improvement.

Adjusting the Feature Vector

Similarly to the application of data uncertainty in UAL [25], the feature vector can be divided element-wise by the estimated model uncertainty vector. If necessary, a constant c can be added to the uncertainty vector before it is applied to the feature vector. This is to circumvent the issue of division by small numbers. The adjusted feature vector $\bar{\mu}^{\mathcal{Q}/\mathcal{G}}$ with the model uncertainty employed as standard deviation $\sqrt{\Sigma^{(M),\mathcal{Q}/\mathcal{G}}}$ is computed as in Equation (31), where $\odot$ represents the element-wise multiplication. The distance matrix $D^{\mu,\sqrt{\Sigma^{(M)}}}$ is then computed using the adjusted query feature vectors $\bar{\mu}^{\mathcal{Q}}$ and gallery feature vectors $\bar{\mu}^{\mathcal{G}}$ as in Equation (32).

$$\bar{\mu}^{\mathcal{Q}/\mathcal{G}} = \mu^{\mathcal{Q}/\mathcal{G}} \odot \frac{1}{\sqrt{\Sigma^{(M),\mathcal{Q}/\mathcal{G}}} + c} \tag{31}$$

$$D^{\mu,\sqrt{\Sigma^{(M)}}} = \mathfrak{d}_{cos}(\bar{\mu}^{\mathcal{Q}}, \bar{\mu}^{\mathcal{G}}) \tag{32}$$

In order to find an appropriate value for the constant c, we performed a logarithmic hyperparameter search and selected the value of c that yielded the best average mAP across all 20 runs. The result achieved with the best identified hyperparameter $c = 0.2435$, is presented in Table 4. We also show the result when the estimated model uncertainty is employed as variance $D^{\mu,\Sigma^{(M)}}$, which was achieved with the best identified hyperparameter $c = 0.1225$.

Table 4. Validation: Can the estimated model uncertainty vector be used to adjust the feature vector?

Distance Matrix	mAP [%]	rank-1 [%]
BOT baseline [66]	86.41 ± 0.09	94.42 ± 0.23
D^μ (identical to UAL [25])	86.77 ± 0.14	94.64 ± 0.21
$D^{\mu,\Sigma^{(M)}}$	87.21 ± 0.17	94.70 ± 0.17
$D^{\mu,\sqrt{\Sigma^{(M)}}}$	87.20 ± 0.17	94.70 ± 0.21

By employing an appropriate c, identified through hyperparameter search, we observe a notable improvement over UAL [25] for both versions of the estimated model uncertainty used to adjust the feature vector. The performance gain measured by the mAP is 0.44 percentage points on average, which is a greater improvement than UAL [25] achieved over the BOT baseline [66] (0.36 percentage points).

Estimating the Hyperparameter c Based on the Uncertainty Vector

With the adjustment of the feature vector, we can get a notable improvement in re-identification performance. However, we must perform a hyperparameter search to find an appropriate constant c. Below, we describe how we can compute the value of c based on the uncertainty vector, in order to eliminate the necessity for a hyperparameter search.

Furthermore, this offers the advantage, that c is not required to be a constant, and can rather be computed individually for each input based on the estimated uncertainty.

The basic observation that led us towards deriving a formula for calculating c based on an uncertainty vector was that the best value $c = 0.2435$ for the standard deviation version of the estimated model uncertainty was approximately twice the best value $c = 0.1225$ for the variance version of the estimated model uncertainty. Also, if we optimize c specifically for a single trained model, we get a similar observation. We hypothesize that by fulfilling Equation (33), we may be able to derive a formula to compute c. First, we can write the uncertainty vectors in the detailed version, to get Equation (34). Now it is more easily observable that the square root is applied element-wise, as mentioned above.

$$f\left(\sqrt{\Sigma^{(M)}}\right) = 2 \cdot f\left(\Sigma^{(M)}\right) \tag{33}$$

$$f\left(\left[\sqrt{\Sigma_1^{(M)}} \ \sqrt{\Sigma_2^{(M)}} \ \cdots \ \sqrt{\Sigma_d^{(M)}}\right]\right) = 2 \cdot f\left(\left[\Sigma_1^{(M)} \ \Sigma_2^{(M)} \ \cdots \ \Sigma_d^{(M)}\right]\right) \tag{34}$$

A function that solves Equation (34) is shown in Equation (35).

$$f\left(\Sigma^{(M)}\right) = \frac{\lambda}{\left|\ln\left(\prod_{i=1}^{d} \Sigma_i^{(M)}\right)\right|} \tag{35}$$

The mathematical proof that the function described in Equation (35) does fulfill Equations (33) and (34), is given in Equations (36)–(43).

$$f\left(\sqrt{\Sigma^{(M)}}\right) = f\left(\left[\sqrt{\Sigma_1^{(M)}} \ \sqrt{\Sigma_2^{(M)}} \ \cdots \ \sqrt{\Sigma_d^{(M)}}\right]\right) \tag{36}$$

$$= \frac{\lambda}{\left|\ln\left(\prod_{i=1}^{d} \sqrt{\Sigma_i^{(M)}}\right)\right|} \tag{37}$$

$$= \frac{\lambda}{\left|\ln\left(\prod_{i=1}^{d} \left(\Sigma_i^{(M)}\right)^{0.5}\right)\right|} \tag{38}$$

$$= \frac{\lambda}{\left|\ln\left(\left(\prod_{i=1}^{d} \Sigma_i^{(M)}\right)^{0.5}\right)\right|} \tag{39}$$

$$= \frac{\lambda}{\left|0.5 \cdot \ln\left(\prod_{i=1}^{d} \Sigma_i^{(M)}\right)\right|} \tag{40}$$

$$= \frac{\lambda}{0.5 \cdot \left|\ln\left(\prod_{i=1}^{d} \Sigma_i^{(M)}\right)\right|} \tag{41}$$

$$= 2 \cdot \frac{\lambda}{\left|\ln\left(\prod_{i=1}^{d} \Sigma_i^{(M)}\right)\right|} \tag{42}$$

$$= 2 \cdot f\left(\Sigma^{(M)}\right) \qquad \square \tag{43}$$

Now, we can reformulate Equation (35) as a function of a norm of an uncertainty vector (Equations (44)–(47)):

$$f\left(\Sigma^{(M)}\right) = \frac{\lambda}{\left|\ln\left(\prod\limits_{i=1}^{d} \Sigma_i^{(M)}\right)\right|} \tag{44}$$

$$= \frac{\lambda}{\left|\sum\limits_{i=1}^{d} \ln\left(\Sigma_i^{(M)}\right)\right|} \tag{45}$$

$$\begin{cases} = \dfrac{\lambda}{\sum\limits_{i=1}^{d}\left|\ln\left(\Sigma_i^{(M)}\right)\right|} & \text{if } \Sigma_i^{(M)} < 1 \ \forall \ i \in \{1,2,\ldots,d\} \\[4ex] \approx \dfrac{\lambda}{\sum\limits_{i=1}^{d}\left|\ln\left(\Sigma_i^{(M)}\right)\right|} & \text{otherwise} \quad \text{\small (in 2.88\% of uncertainty vectors)} \end{cases} \tag{46}$$

$$\begin{cases} = \dfrac{\lambda}{\left\|\ln\left(\Sigma^{(M)}\right)\right\|_1} & \text{if } \Sigma_i^{(M)} < 1 \ \forall \ i \in \{1,2,\ldots,d\} \\[3ex] \approx \dfrac{\lambda}{\left\|\ln\left(\Sigma^{(M)}\right)\right\|_1} & \text{otherwise} \quad \text{\small (in 2.88\% of uncertainty vectors)} \end{cases} \tag{47}$$

Note, that the derived formula using the norm (Equation (47)) is only equivalent to Equation (44) if all elements of the vector are smaller than 1. This requirement is fulfilled in the majority of cases. However, in 2.88% of the uncertainty vectors, at least one element is larger than 1. For these vectors, we performed a worst-case analysis, that showed, that the influence of the large values for some elements of the vector is marginal. Equation (48) describes the extent to which the two computations deviate in cases where $\Sigma_i^{(M)} > 1$ for some i. For the vector with the largest deviation over all runs, we observed a factor $\alpha = 1.0024$. This means that the error never exceeds 0.24%. This minimal error is negligible given that the achieved performance, as measured by mAP, is nearly unaffected when varying c by this amount.

$$\frac{\lambda}{\left\|\ln\left(\Sigma^{(M)}\right)\right\|_1} \leq \frac{\lambda}{\left|\sum\limits_{i=1}^{d} \ln\left(\Sigma_i^{(M)}\right)\right|} < \alpha \cdot \frac{\lambda}{\left\|\ln\left(\Sigma^{(M)}\right)\right\|_1} \tag{48}$$

We conclude that we can use Equation (47) to compute c. Unfortunately, in Equation (47) a new parameter λ is introduced. Our hyperparameter search experiments show that while the best values for c deviate strongly for different trained models, they do not deviate as much if we use λ. The best value for λ is approximately half the dimensionality of the uncertainty vector $\frac{d}{2}$. Using $\lambda = \frac{d}{2}$, we can now compute an individual c for each input. Table 5 shows that applying this function to each input produces almost identical performance compared to the best results achieved with a constant c.

Table 5. Validation: Can the estimated model uncertainty vector be used to estimate the hyperparameter c for each input individually? To simplify the comparison to previous results, the results of Table 4 are repeated in the 3rd and 4th row.

Distance Matrix	mAP [%]	rank-1 [%]
BOT baseline [66]	86.41 ± 0.09	94.42 ± 0.23
D^μ (identical to UAL [25])	86.77 ± 0.14	94.64 ± 0.21
$D^{\mu,\Sigma^{(M)}}, c = 0.1225$	87.21 ± 0.17	94.70 ± 0.17
$D^{\mu,\sqrt{\Sigma^{(M)}}}, c = 0.2435$	87.20 ± 0.17	94.70 ± 0.21
$D^{\mu,\Sigma^{(M)}}, c = f\left(\Sigma^{(M)}\right)$	87.21 ± 0.18	94.68 ± 0.18
$D^{\mu,\sqrt{\Sigma^{(M)}}}, c = f\left(\sqrt{\Sigma^{(M)}}\right)$	87.19 ± 0.17	94.69 ± 0.20

With the approximation $\lambda = \frac{d}{2}$ and the application of Equation (47) for each input, we got almost identical results while removing the need for a hyperparameter search. We are now able to use the estimated model uncertainty to adjust the feature vector in a straightforward way.

Conclusion on the Utilization of Uncertainties

Data uncertainty is already utilized in our architecture to divide the result of the mean head by the result of the variance head, as in the UAL approach [25]. We showed that the estimated *model uncertainty* vector can be utilized to further adjust the feature vector.

The estimated distributional uncertainty is very helpful for identifying out-of-distribution data, which is explained in more detail in Section 4.4. Additionally, the estimated distributional uncertainty can be utilized to derive a scalar value to further improve the ReID performance, but only marginally, as explained below.

Further Investigations

To verify that our derived formula is a good way to estimate the hyperparameter c, we also experimented with a wide variety of possibilities. We examined the three types of uncertainties as the basis for the computation of a constant c, both as standard deviation and as variance. We derived a scalar value by applying the inverse of the l_1-norm of the logarithm $((\|\ln(\cdot)\|_1)^{-1})$, the inverse of the l_1-norm $((\|\cdot\|_1)^{-1})$, the entropy $(\mathcal{H}(\cdot))$, and the vector length $(\|\cdot\|_2)$. For each of these scalar values, we performed a hyperparameter search to find an adequate multiplication factor λ. Furthermore, instead of the model uncertainty we also employed the other uncertainty types, both as standard deviation and as variance, to refine the feature vector as explained above. Overall, we checked 150 variations, each optimized via hyperparameter search. None of these variants improved the performance noticeably over the approach described above.

The best result is achieved by applying the inverse of the l_1-norm of the distributional uncertainty estimate as variance, and to refine the feature vector with the model uncertainty estimate as variance and the derived scalar constant. However, this result (mAP [%] = 87.213 ± 0.170) is only 0.007 percentage points better than the result when employing Equation (47) (87.206 ± 0.178). Therefore, it can be considered approximately equal. Nevertheless, the result described here, represents the best result achieved by our approach UBER in this paper. Therefore, this result is also shown in Table 1 and Figure 5 to compare with the state of the art. Specifically in Figure 5, we can see the significant improvement over UAL [25], which served as the basis for our network architecture.

Using the entropy, the inverse of the l_1-norm of the logarithm, or the inverse of the l_1-norm of any of the uncertainty types to derive the scalar value also achieved good results. The vector length was only appropriate in combination with the data uncertainty estimate.

4.4. Experimental Validation of the Distributional Uncertainty Estimate

As previously discussed in Section 3.1, distributional uncertainty is higher the more dissimilar the input is to the training data. The objective of this experiment is to verify that our distributional uncertainty estimate $\Sigma^{(V)}$ fits this description. Additionally, we test whether the other uncertainty estimates exhibit characteristics of a distributional uncertainty estimate. This is done to demonstrate that our distributional uncertainty estimate is the most appropriate for estimating distributional uncertainty. To that end, we construct four sets of out-of-distribution (OOD) images such that each successive set (D1)–(D4) represents a class of progressively greater dissimilarity to the training data (Section 4.4.1). We use the entropy function derived in Section 3.3.6 to translate our vector-based uncertainty estimates into scalar uncertainty scores and analyze the distribution of these scores for the different OOD sets in comparison to an in-distribution set (Section 4.4.2). Finally, we examine the score distribution of one uncertainty type in cases where another uncertainty type yields very low or very high scores (Section 4.4.3).

4.4.1. Out-of-Distribution Sets

The source of the images in these sets are the 2798 distractor images contained in the gallery partition of the Market-1501 dataset [68]. First, we manually categorized the distractors in detail and then summarized, arriving at four classes of similar difference to the training data. In this process, we also ensured that each category is composed of a reasonable number of images. Examples for the resulting categories can be seen in Figure 7 and a description of each set is provided below.
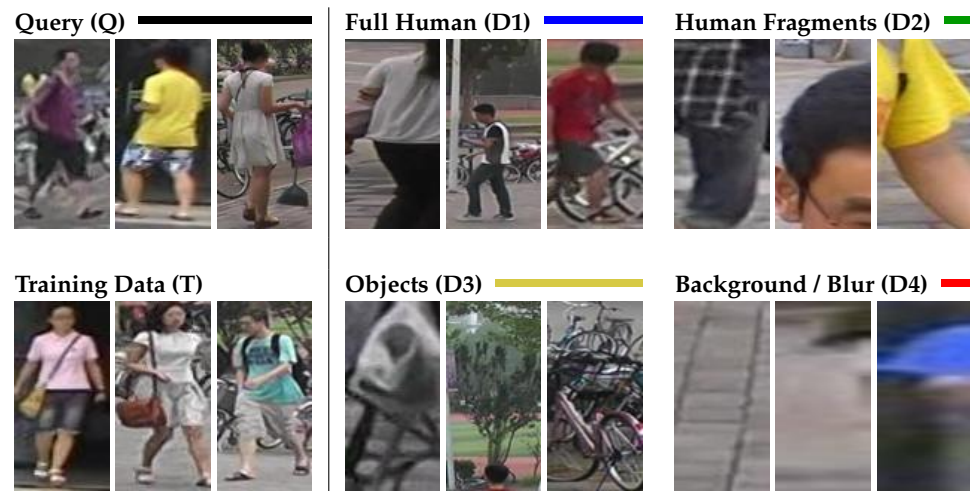
Figure 7. Examples of training images (T), the in-distribution set (Q), and the distractor sets (D1)–(D4). Images in (D1) contain large parts of the depicted humans. In (D2) they contain only fragments of a person. In (D3) they contain non-human objects. In (D4) they contain only background or blur. All images are part of the Market-1501 dataset [68].

Training Data (T) and Query (Q)

The images used during training (T) and as queries (Q) during inference both depict persons centered in the image. Since the training data (T) was already used for model training, we employ the query set (Q) as the in-distribution set for validation.

Full Human (D1)

The set (D1) contains images with imperfect bounding box placement, where the level of detail visible is nonetheless sufficient to enable a confident prediction of the identity depicted. Most of these images either have parts of the person missing (too zoomed-in) or show a lot of background (too zoomed-out). Some images exhibit bad image quality due to low resolution. This set is very similar to the training data. However, some relevant parts of the person might be missing and others are at different positions than expected.

Human Fragments (D2)

The set (D2) contains images that are zoomed-in to such an extent that only fragments of a person are visible which are not enough to confidently make a prediction of the identity depicted. Both the main content of the image and the background are known to the network but appear in different sizes, positions, and contexts in the training data.

Objects (D3)

The set (D3) contains images of non-human objects which can certainly be called a separate domain from what the model was trained on, even if it has seen them in the background of the training data. However, we still find the same general setup of a structured object placed in front of the known background, although now the foreground does not depict humans but objects such as bikes or trees.

Background/Blur (D4)

The set (D4) contains images where only background is visible or where only unrecognizable blobs of color can be seen that are caused by a combination of zoom and motion blur or focus blur. Here, not even the general setup remains the same: While the model may have seen some of the backgrounds in training images, it has not seen background without any foreground object. Furthermore, the blurred images are completely different from anything the model has seen. Both kinds of images do not contain the structured shapes the model expects.

In summary, the query set (Q) is used as the in-distribution set, while the distractor sets (D1)–(D4) represent out-of-distribution sets with increasing level of difference to the training set (T).

Labeling Process

For the labeling process, the distractor images were presented in random order and manually annotated according to the four categories. The labeling took 1–2 s per image. Due to the ambiguous nature of the images, imperfections in the composition of the distractor sets are to be expected. The number of images in the resulting sets (D1)-(D4) is given in Table 6. The labels for the distractors of the Market-1501 dataset are publicly available as part of the source code of this paper.

Table 6. Number of images in the training (T) and query (Q) set, as well as in the annotated subsets (D1)–(D4) of the distractors in Market-1501.

Set		# of Images
Training Data	(T)	12,936
Query	(Q)	3368
Full Human	(D1)	278
Human Fragments	(D2)	1729
Objects	(D3)	567
Background/Blur	(D4)	224

4.4.2. Analysis of Uncertainty Score Distributions

As mentioned above, the OOD sets (D1)–(D4) presented in the previous section represent classes of progressively greater dissimilarity to the training data. This means that the OOD degree, i.e., the amount to which an image deviates from the training distribution, increases from (D1) to (D4).

If our distributional uncertainty estimate $\Sigma^{(V)}$ is indeed a good estimate of distributional uncertainty, it should measure the OOD degree. Thus we expect to observe a gradual increase of the distributional uncertainty score from the distractor sets (D1)–(D4). Since the sets are not perfectly separable, an overlap between the distributions is to be expected. However, the distributions should be somewhat separable. At the very least, the means of the uncertainty score distributions for the sets (D1) to (D4) should appear in this order. Furthermore, we expect the distributional uncertainty estimate to outperform the other uncertainty estimates regarding the ability to separate the distributions.

Experimental Setup

As was laid out in detail in Section 3.3, the data uncertainty estimate $\Sigma^{(D)}$ is the diagonal of the covariance matrix of the probabilistic embedding, while the model and distributional uncertainty estimates $\Sigma^{(M)}$ and $\Sigma^{(V)}$ are the variance induced by the Bayesian module in the raw mean vector $\widehat{\mu}$ and data uncertainty estimate $\Sigma^{(D)}$.

All three uncertainty estimates are vectors, so it is not immediately clear how to calculate a scalar uncertainty score from them. As described in Section 3.3.6, we have decided in favor of entropy for this purpose.

After training our model on Market-1501, we generate the three uncertainty estimates for all relevant images regarding this experiment and compute the scalar uncertainty scores according to Equation (17) for each of them. Then, we employ kernel density estimation (KDE) to model the probability density functions corresponding to the observed values for each of the considered sets, i.e., the query set (Q) as the in-distribution set and the OOD sets (D1)–(D4). The resulting KDE plots are shown in Figure 8a–c.
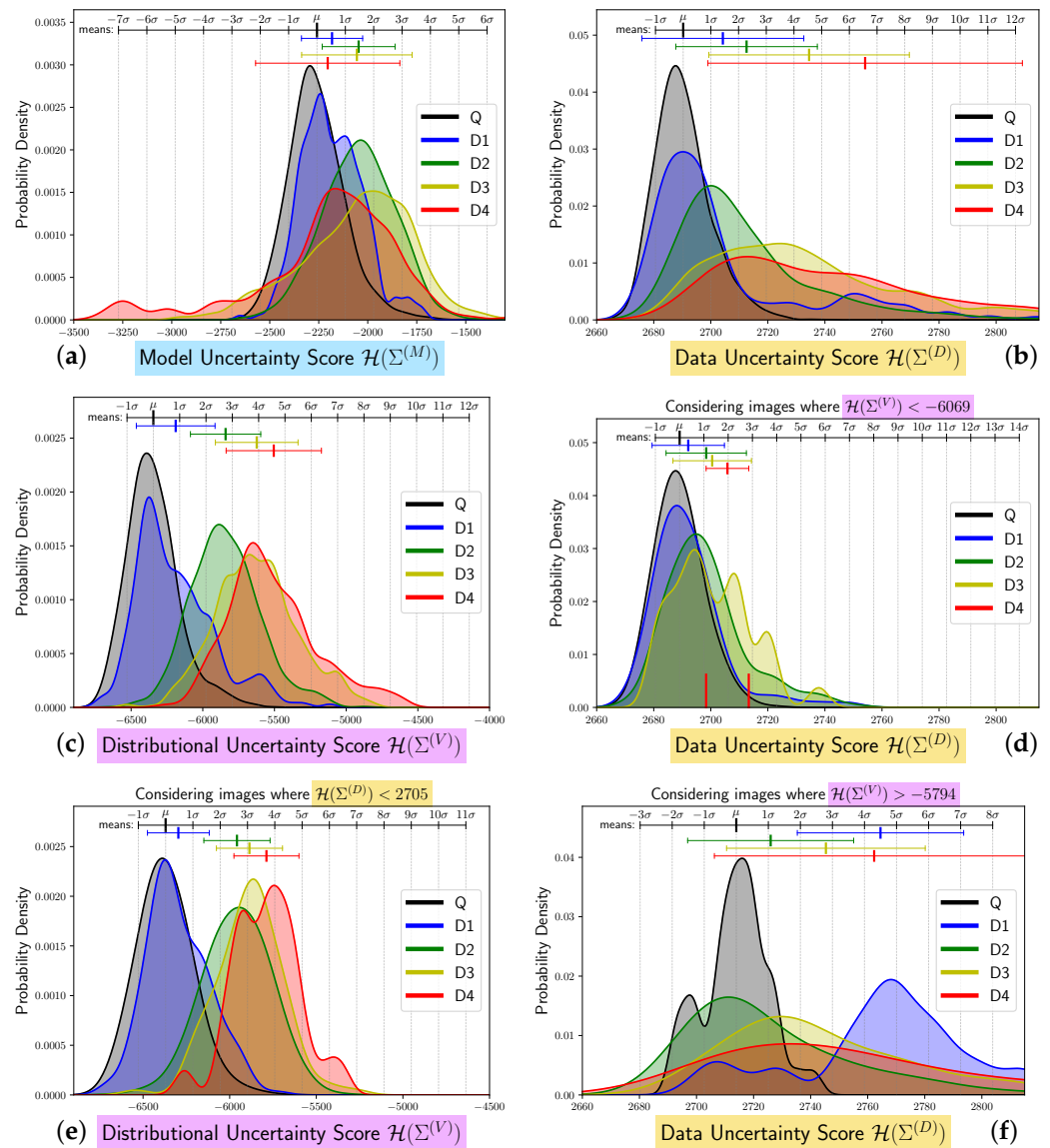

Figure 8. Distribution of uncertainty scores over all images in certain subsets of Market-1501 for the three uncertainty types (**a**) model uncertainty, (**b**) data uncertainty, and (**c**) distributional uncertainty. The five subsets (Q), (D1)–(D4) are described in Section 4.4.1 and examples are provided in Figure 7. (**d**) Data uncertainty in case of low distributional uncertainty. (**e**) Distributional uncertainty in case of low data uncertainty. (**f**) Data uncertainty in case of high distributional uncertainty. On top of each of the probability density plots, the means and standard deviations of each distribution are plotted in relation to the query set (Q).

Model Uncertainty Score Distribution

In the mathematical model (Equation (2)), the model uncertainty term $p(\theta \mid \mathcal{D})$ is both separate and independent of the distributional uncertainty term $p(\xi \mid x, \theta)$. Therefore, we would expect the model uncertainty score distributions for the different sets to be indifferent to the OOD degree, i.e., we should see no ordered separation of the sets.

Figure 8a shows the mean of the model uncertainty scores for all sets on top of the distributions. Table 7a lists how far away the mean of the uncertainty scores of the distractor sets (D1)–(D4) are from the in-distribution set (Q) in terms of the standard deviation of the in-distribution set. We can immediately see that the means of the distributions are not in the correct order. Furthermore, there exists a large overlap between the distributions of all sets. Surprisingly, some of the lowest model uncertainty values are observed for images in the distractor set (D4) with the highest OOD degree.

Table 7. Number of standard deviations of the query score distribution that the mean of the uncertainty score distribution of the respective set is above the mean of the query score distribution for (**a**) model uncertainty, (**b**) data uncertainty, (**c**) distributional uncertainty, (**d**) data uncertainty in case of low distributional uncertainty, (**e**) distributional uncertainty in case of low data uncertainty, and (**f**) data uncertainty in case of high distributional uncertainty. For a visualization, we refer to Figure 8.

Set		(a)	(b)	(c)	(d)	(e)	(f)
Full Human	(D1)	0.53	1.43	0.85	0.35	0.47	4.50
Human Fragments	(D2)	1.47	2.29	2.74	1.10	2.61	1.07
Objects	(D3)	1.41	4.55	3.93	1.34	3.07	2.80
Background/Blur	(D4)	0.38	6.57	4.57	1.97	3.69	4.30

It appears that the model uncertainty estimate is unsuited to predicting the OOD degree. This fits our expectations based on the mathematical theory of how a model uncertainty estimate should behave in this situation.

Data Uncertainty Score Distribution

The data uncertainty term $p(z \mid \xi)$ in the mathematical model is separate from but not independent of the distributional uncertainty term $p(\xi \mid x, \theta)$. Ideally, the data uncertainty score distributions would be indifferent to the OOD degree, as it is a separate quantity. However, the conditional dependence of the data uncertainty term on the distributional uncertainty term likely means that this is not the case. As we saw in Figure 1, the reliability of the data uncertainty estimate decreases with increasing distributional uncertainty. Therefore, images with higher distributional uncertainty should exhibit larger spread in data uncertainty estimates around their actual data uncertainty.

As can be seen in Figure 8b and Table 7b, the means of the distributions are ordered by the OOD-degree of the respective set. In Figure 8b and Table 8b, we also observe that the uncertainty score distributions for the OOD sets exhibit standard deviations that are considerably higher than the one of the query set and are higher for greater OOD degree. Only the distractor set (D1) with the lowest OOD degree does not quite fit this pattern, since its standard deviation is larger than that of the distractor set (D2) with the slightly higher OOD degree.

Table 8. Factor by which the standard deviation of the uncertainty score distribution for the respective set is larger than the standard deviation of the query score distribution for (**a**) model uncertainty, (**b**) data uncertainty, (**c**) distributional uncertainty, (**d**) data uncertainty in case of low distributional uncertainty, (**e**) distributional uncertainty in case of low data uncertainty, and (**f**) data uncertainty in case of high distributional uncertainty. For a visualization, we refer to Figure 8.

Set		(a)	(b)	(c)	(d)	(e)	(f)
Full Human	(D1)	1.08	2.92	1.50	1.49	1.13	2.60
Human Fragments	(D2)	1.28	2.55	1.33	1.66	1.21	2.59
Objects	(D3)	1.95	3.62	1.57	1.62	1.21	3.10
Background/Blur	(D4)	2.54	5.68	1.81	0.88	1.19	4.98

The data uncertainty estimate appears to be somewhat suitable for predicting the OOD degree, though the large standard deviations and largely overlapping distributions make it less than ideal. While this predictive ability seemingly contradicts the theoretical considerations, this may be due to the issue with its reliability in cases of high distributional uncertainty mentioned above. This is supported by the fact that the standard deviation of the data uncertainty score distributions generally increases with OOD degree, which is exactly what we expected from the mathematical model. While the distractor set (D1) with the lowest OOD degree does not quite fit this pattern, we can clearly see in Figure 8b that the main body of the distribution of (D1) does have a smaller standard deviation than that of the distractor set (D2), which has a slightly higher OOD degree. Furthermore, we can see that the unexpectedly high standard deviation of the distractor set (D1) is due to a smaller second mode in the distribution with images of higher data uncertainty scores. This is likely to be the part of the images in (D1) which show noticeably worse image quality compared to the others, which correctly is associated with a higher data uncertainty.

Overall, we have some supporting and some unclear evidence in relation to the theoretical considerations. Therefore, we also examine the distributions of only those images with low distributional uncertainty, which should be free of the observed effect. This investigation is further discussed in Section 4.4.3.

Distributional Uncertainty Score Distribution

Since distributional uncertainty is precisely the type of uncertainty that measures the OOD degree, we would expect a clear ordering of the distributional uncertainty score distributions and a clear distinction between the in-distribution set (Q) and the OOD sets (D1)–(D4). However, the distractor set (D1) with the lowest OOD degree is still largely similar to the in-distribution set (Q) and contains those images that still contain enough information to make a confident prediction regarding the identity. Therefore, we expect the distribution of the distractor set (D1) to be quite similar to that of the in-distribution set (Q). Conversely, the distractor set (D4) with the highest OOD degree should have a clear distinction from the in-distribution set (Q), even if it does not necessarily have to be perfectly separable due to the ambiguous nature of many images, as discussed in Section 4.4.1.

We can immediately see in Figure 8c and Table 7c that the means of the distributions are ordered by the OOD degree of the respective set. In Figure 8c and Table 8c, we can also compare the relative standard deviations of the distributions in terms of multiples of the standard deviation of the in-distribution set (Q) for the distractor sets (D1)–(D4). We observe that the relative standard deviations of the distributional uncertainty score distributions for the distractor sets (D1)–(D4) in Figure 8c and Table 8c are smaller than those of the data uncertainty score distributions for the distractor sets (D1)–(D4) in Figure 8b and Table 8b. Furthermore, we observe the same pattern of increase in standard deviation for an increase in OOD degree as for the data uncertainty score distributions, including the exception for the distractor set (D1). However, we observe much smaller relative standard deviations for the distributional uncertainty score distributions.

This seems to indicate a confident prediction of distributional uncertainty that does not become more unreliable with increased OOD degree of the distractor sets. The degree of overlap between the in-distribution set (Q) and the distractor set (D1) with the lowest OOD degree is very high. Therefore, the distractor set (D1) is rather judged as an in-distribution set, which is plausible considering that a re-identification of the depicted individuals is still possible. The degree of overlap between the in-distribution set (Q) and the respective distractor sets (D2)–(D4) is smaller the higher the OOD degree is. Therefore, the distributional uncertainty estimate seems to be a good predictor of the OOD degree. While the distributions are not perfectly separated, a cutoff can be chosen that excludes only very few images of the in-distribution set (Q) while still excluding the vast majority of the images in the distractor set (D4) with the highest OOD degree. Essentially, our observations perfectly matched our expectations from mathematical theory.

4.4.3. Uncertainty Score Distributions When Controlling for Another Uncertainty Type

Since our results regarding the prediction of the OOD degree by utilizing the data uncertainty estimate were somewhat inconclusive, in the following, we conduct further investigations. First, we examine the estimated data uncertainty for subsets of images for which low distributional uncertainty was estimated. Furthermore, we also examine the estimated distributional uncertainty for subsets of images for which low data uncertainty was estimated. This is intended to confirm that, as suggested by the mathematical model (Equation (2)), the dependency between data uncertainty and distributional uncertainty is unidirectional. Additionally, we examine the estimated data uncertainty for subsets of images for which high distributional uncertainty was estimated. This serves to verify the derivation from the mathematical model that given high distributional uncertainty, the data uncertainty estimate becomes unreliable.

Data Uncertainty in Case of Low Distributional Uncertainty

As discussed in Section 3.1.2, we should not trust the data uncertainty estimate in case of high distributional uncertainty. For this reason, we investigate the data uncertainty score distribution of images with low distributional uncertainty scores. Based on the mathematical theory, we expect that in case of low distributional uncertainty, the width of the data uncertainty score distributions should be small for all sets (Q), (D1)–(D4). Furthermore, we expect that a low distributional uncertainty score indicates a low OOD degree. Therefore, the ability of data uncertainty to appropriately order and separate the distractor sets should be strongly diminished.

In contrast to the previous experiments, we now do not compute the distributions on the entire sets (Q), (D1)–(D4), but instead filter the sets. First, we examine data uncertainty in case of low distributional uncertainty. Therefore, we use only the samples of the sets (Q), (D1)–(D4) that had a low distributional uncertainty score in Figure 8c. As criteria, we use a distance of 1.5 standard deviations with respect to the query set (Q) to the mean of the query set (Q), as defined in Equation (49).

$$\mathcal{H}\left(\Sigma^{(V)}\right) < \mathbb{E}\left\{\mathcal{H}\left(\Sigma_i^{(V),\mathcal{Q}}\right)\right\}_{i=1}^{Q} + 1.5 \cdot \sqrt{\mathbb{V}\mathrm{ar}\left\{\mathcal{H}\left(\Sigma_i^{(V),\mathcal{Q}}\right)\right\}_{i=1}^{Q}} \tag{49}$$

Therefore, the threshold for filtering the sets is $\tau_{\downarrow\mathcal{H}(\Sigma^{(V)})} = -6069$. The remaining number of images in each set, along with the percentage of the original set size that remains, is provided in Table 9.

Table 9. Statistics for subsets of images with low distributional uncertainty ($\downarrow\mathcal{H}(\Sigma^{(V)})$).

Set		# of Images	Remaining
Query	$(Q)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$	3119	92.6%
Full Human	$(D1)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$	199	71.6%
Human Fragments	$(D2)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$	304	17.6%
Objects	$(D3)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$	31	5.5%
Background/Blur	$(D4)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$	2	0.9%

We can see, that only two images of the distractor set (D4) with the highest OOD degree remain. Similarly, only 31 images remain of the distractor set (D3) with the second-highest OOD degree.

Figure 8d shows the distributions of the reduced sets. Since the distractor set $(D4)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$ contains only 2 samples, their uncertainty values are plotted instead of a distribution. We observe that the sets are still ordered correctly by the data uncertainty estimate. However, the distributions for the different sets share a large overlap. The distractor set $(D1)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$ almost perfectly overlays the in-distribution set $(Q)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$, with only some exceptions

that have a higher data uncertainty score. The distractor sets $(D2)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$ and $(D3)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$ with slightly higher OOD degree still share a large overlap with the in-distribution set $(Q)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$. The two samples of the distractor set $(D4)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$ with the highest OOD degree also fall within the overlapping area of the in-distribution set $(Q)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$.

Even though the means of the distributions of the sets remain in the correct order, the large overlap between the distributions of the in-distribution set $(Q)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$ and the distractor sets $(D1)$–$(D4)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$ shows that the data uncertainty estimate is not very useful for distinguishing the in-distribution set $(Q)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$ and the distractor sets $(D1)$–$(D4)^{\downarrow\mathcal{H}(\Sigma^{(V)})}$ in case of low distributional uncertainty. The reason that the data uncertainty estimate is still able to order the sets correctly to a certain extent can be attributed to three factors: (1) measurement errors and image corruptions, which contribute to both kinds of uncertainty; (2) labeling errors stemming from the ambiguous nature of the sets; and (3) the high correlation between data and distributional uncertainty, as discussed in Section 4.2.

Overall, we observe exactly what we can expect based on the mathematical theory. In case of low estimated distributional uncertainty, the distributions of the data uncertainty estimates are much sharper and the difference between the distractor sets is strongly diminished.

Distributional Uncertainty in Case of Low Data Uncertainty

Now, we investigate whether there are similar results for the distributional uncertainty score distributions when considering images with low data uncertainty scores. From Figure 4d, we can already observe that this is likely not the case. Instead, in case of low estimated data uncertainty, the ability of the distributional uncertainty estimate to differentiate the in-distribution set (Q) and the distractor sets (D1)–(D4) should not be affected. Moreover, also based on the mathematical theory where only data uncertainty is conditioned on distributional uncertainty and not the other way around, the distributional uncertainty score distributions of the sets should not be affected in case of low data uncertainty.

Similarly to the previous experiment, we first filter the sets before computing the distributions. This time, we use only the samples of the sets (Q), (D1)–(D4) that had a low data uncertainty score in Figure 8b. As criteria, we use a distance of 1.5 standard deviations with respect to the query set (Q) to the mean of the query set (Q), as defined in Equation (50).

$$\mathcal{H}\left(\Sigma^{(D)}\right) < \mathbb{E}\left\{\mathcal{H}\left(\Sigma_i^{(D),\mathcal{Q}}\right)\right\}_{i=1}^{Q} + 1.5 \cdot \sqrt{\mathbb{V}\mathrm{ar}\left\{\mathcal{H}\left(\Sigma_i^{(D),\mathcal{Q}}\right)\right\}_{i=1}^{Q}} \tag{50}$$

Therefore, the threshold for filtering the sets is $\tau_{\downarrow\mathcal{H}(\Sigma^{(D)})} = 2705$. The remaining number of images in each set, along with the percentage of the original set size that remains, is provided in Table 10.

Table 10. Statistics for subsets of images with low data uncertainty ($\downarrow\mathcal{H}(\Sigma^{(D)})$).

Set		# of Images	Remaining
Query	$(Q)^{\downarrow\mathcal{H}(\Sigma^{(D)})}$	3110	92.3%
Full Human	$(D1)^{\downarrow\mathcal{H}(\Sigma^{(D)})}$	204	73.4%
Human Fragments	$(D2)^{\downarrow\mathcal{H}(\Sigma^{(D)})}$	825	47.7%
Objects	$(D3)^{\downarrow\mathcal{H}(\Sigma^{(D)})}$	110	19.4%
Background/Blur	$(D4)^{\downarrow\mathcal{H}(\Sigma^{(D)})}$	30	13.4%

In all sets, at least 30 images remain. Therefore, we can display the distributions for all sets. Figure 8e shows the distributions of the reduced sets. As expected, the distributional

uncertainty score distributions of the reduced sets (Q), (D1)–(D4)$^{\downarrow\mathcal{H}(\Sigma^{(D)})}$ in Figure 8e are very similar to the distributional uncertainty score distributions of the original sets (Q), (D1)–(D4) in Figure 8c. The means of the distributions of the sets (Q), (D1)–(D4)$^{\downarrow\mathcal{H}(\Sigma^{(D)})}$ remain in the correct order. The distractor set (D1)$^{\downarrow\mathcal{H}(\Sigma^{(D)})}$ still largely overlaps with the in-distribution set (Q)$^{\downarrow\mathcal{H}(\Sigma^{(D)})}$ for the previously described reasons. The distractor sets (D2)–(D4)$^{\downarrow\mathcal{H}(\Sigma^{(D)})}$ with a higher OOD degree share only a small overlap with the in-distribution set (Q)$^{\downarrow\mathcal{H}(\Sigma^{(D)})}$. The mean of the distributions of the two distractor sets (D3)$^{\downarrow\mathcal{H}(\Sigma^{(D)})}$ and (D4)$^{\downarrow\mathcal{H}(\Sigma^{(D)})}$ with the highest OOD degree move slightly closer to the in-distribution set (Q)$^{\downarrow\mathcal{H}(\Sigma^{(D)})}$. Based on Figure 4d, this can be attributed to the fact that the samples in the original sets (D3) and (D4) with the highest estimated distributional uncertainty also have an estimated data uncertainty score higher than the threshold $\tau_{\downarrow\mathcal{H}(\Sigma^{(D)})}$.

We essentially observe what we expected from the mathematical theory. The distributional uncertainty score distributions are barely affected by filtering based on low estimated data uncertainty. The distributional uncertainty estimate is still able to predict the OOD degree. Therefore, this observation gives strong support to our hypothesis that our estimate of distributional uncertainty is a much better fit than the data uncertainty estimate for predicting the OOD degree. Furthermore, the alignment of the observation and the mathematical theory supports our hypothesis that the proposed extraction of distributional uncertainty as variance of the learned variance vector is a mathematically grounded distributional uncertainty estimate.

Data Uncertainty in Case of High Distributional Uncertainty

In case of low estimated distributional uncertainty, we saw very sharp distributions of the data uncertainty scores for all the sets. Below, we investigate how the data uncertainty estimate behaves in case of high estimated distributional uncertainty. Based on the mathematical theory, in this case, we expect the data uncertainty score distributions to be very wide since the model's ability to accurately predict the data uncertainty is expected to be reduced under the condition of high distributional uncertainty. If we verify this, it would imply that we should not trust the data uncertainty estimate in this situation.

Again, we first filter the sets before computing the distributions. This time, we use only the samples of the sets (Q), (D1)–(D4) that had a high distributional uncertainty score in Figure 8c. As criteria, we use a distance of 3 standard deviations with respect to the query set (Q) to the mean of the query set (Q), as defined in Equation (51).

$$\mathcal{H}\left(\Sigma^{(V)}\right) > \mathbb{E}\left\{\mathcal{H}\left(\Sigma_i^{(V),\mathcal{Q}}\right)\right\}_{i=1}^{Q} + 3 \cdot \sqrt{\mathbb{V}\mathrm{ar}\left\{\mathcal{H}\left(\Sigma_i^{(V),\mathcal{Q}}\right)\right\}_{i=1}^{Q}} \tag{51}$$

Therefore, the threshold for filtering the sets is $\tau_{\uparrow\mathcal{H}(\Sigma^{(V)})} = -5794$. The remaining number of images in each set, along with the percentage of the original set size that remains, is provided in Table 11.

Table 11. Statistics for subsets of images with high distributional uncertainty ($\uparrow\mathcal{H}(\Sigma^{(V)})$).

Set		# of Images	Remaining
Query	(Q)$^{\uparrow\mathcal{H}(\Sigma^{(V)})}$	38	1.1%
Full Human	(D1)$^{\uparrow\mathcal{H}(\Sigma^{(V)})}$	26	9.4%
Human Fragments	(D2)$^{\uparrow\mathcal{H}(\Sigma^{(V)})}$	685	39.6%
Objects	(D3)$^{\uparrow\mathcal{H}(\Sigma^{(V)})}$	401	70.7%
Background/Blur	(D4)$^{\uparrow\mathcal{H}(\Sigma^{(V)})}$	188	83.9%

In all sets, at least 26 images remain. Therefore, we can display the distributions for all sets. Figure 8f shows the distributions of the reduced sets.

As expected, the distributions for the distractor sets $(D1)–(D4)^{\uparrow \mathcal{H}(\Sigma^{(V)})}$ are very wide. The order of the distractor sets $(D2)–(D4)^{\uparrow \mathcal{H}(\Sigma^{(V)})}$ with the highest OOD degree are correct. We hypothesize that this observation can be attributed to the correlation between data uncertainty and distributional uncertainty. However, for the distractor set $(D1)^{\uparrow \mathcal{H}(\Sigma^{(V)})}$ with the lowest OOD degree, we observe the highest mean value of the data uncertainty score distribution. This can be attributed to the previous observation that some samples in (D1) have poor image quality and are correctly estimated to have high data uncertainty. Simultaneously, these samples have relatively high estimated distributional uncertainty. This would suggest that the image corruptions are so severe that the images are identified as being out-of-distribution.

Essentially, our observation of wide data uncertainty score distributions in case of high distributional uncertainty matches the expectation from mathematical theory. This provides further evidence that our proposed method for estimating data uncertainty and distributional uncertainty is mathematically grounded.

Summary

From our experiments regarding the ability to predict the extent to which an input is out-of-distribution, we conclude that our *distributional uncertainty* estimate is a suitable measure of the OOD degree. Since all our observations align with mathematical theory, we have strong support for our hypothesis that the proposed extraction of the variance of the learned variance vector is a mathematically grounded estimate of distributional uncertainty. Furthermore, we observed that the model uncertainty and data uncertainty estimates are less suitable for predicting the OOD degree. This also aligns with mathematical theory and findings in related work [57].

5. Conclusions

In the context of autonomous agents, it is of critical importance to estimate the uncertainty in the prediction of deep-learning-based modules utilized for the perception of their surrounding environment. In the event that an autonomous agent interacts with humans, the deployment of uncertainty as a metric for trustworthiness is of particular significance, as misjudgments have the potential to result in substantial consequences, including damage or injury. However, if the network output is not a regression or classification, but instead a feature vector destined for subsequent processing modules, the estimation of uncertainty becomes more challenging. Furthermore, if the estimated uncertainty indicates that the prediction is unreliable, it may be beneficial to identify the underlying cause in order to enhance the prediction. Accordingly, three types of uncertainty can be extracted: model uncertainty, data uncertainty, and distributional uncertainty.

We proposed an architecture that builds upon the UAL approach [25] to estimate the three types of uncertainty. In accordance with related work [22,23,28], we utilized the person re-identification task for benchmarking purposes. In particular, we conducted our experiments on the Market-1501 dataset [68], which is the most frequently used within this context.

Our comprehensive experimental analysis demonstrated that the estimates for the three types of uncertainties exhibited the characteristics expected from mathematical theory. Furthermore, we showed that the data uncertainty and model uncertainty estimates provide valuable information for refining the feature vector, thereby enhancing the re-identification quality. The utilization of refined feature vectors by our approach, named <u>u</u>ncertainty-<u>b</u>ased <u>e</u>mbedding <u>r</u>efinement (UBER), results in a superior performance compared to state-of-the-art approaches that also employ uncertainty estimation for feature vectors. The improvement of our approach UBER over UAL [25] is higher than the improvement of UAL [25] over the already strong baseline BOT [66]. Additionally, we demonstrated that the distributional uncertainty estimate can reliably quantify the extent to which an input image is out-of-distribution. In contrast, the other types of uncertainties, which were taken into consideration in this context by Schwaiger et al. [57], are less reliable measures. The

capacity to predict when an image is out-of-distribution, and therefore when the prediction cannot be trusted, is an important ability for an autonomous agent.

In summary, we proposed an architecture to extract mathematically grounded estimates for the three types of uncertainties and demonstrated the individual value of these uncertainty types in our experiments.

Future Work

Our findings demonstrate that the utilization of uncertainties for refining the feature vector is advantageous for enhancing the performance of person re-identification on the Market-1501 dataset. Future work should investigate whether our approach, UBER, can also be employed to enhance the performance in related feature-vector-based tasks, such as face recognition, object re-identification, content-based image retrieval, zero-shot action recognition, or natural language processing. It should also be investigated whether it can be employed for cross-dataset person re-identification. In this context, it would be particularly interesting to examine whether other backgrounds are assessed as being out-of-distribution and whether this information can be attributed to specific feature vector components that encode context information.

Furthermore, as a methodical investigation, future work could incorporate loss functions that operate directly on the basis of distributions, such as the Bayesian Triplet Loss [36].

Our comprehensive experiments regarding the distribution of uncertainty scores for different distractor sets revealed some relations between estimated data uncertainty and estimated distributional uncertainty. In future work, qualitative analyses could examine the characteristics of images that are judged to have high data uncertainty and low distributional uncertainty, or similar combinations.

Moreover, we provided a mathematical and methodological foundation for the utilization of uncertainties in feature-vector-based tasks. It is now up to future research to demonstrate the potential of this approach when deployed in practical autonomous systems. This will reveal whether the utilization and visualization of uncertainties enhance user trust and acceptance of these autonomous systems.

References

1. Daily, M.; Medasani, S.; Behringer, R.; Trivedi, M. Self-driving cars. *Computer* **2017**, *50*, 18–23. [CrossRef]
2. Badue, C.; Guidolini, R.; Carneiro, R.V.; Azevedo, P.; Cardoso, V.B.; Forechi, A.; Jesus, L.; Berriel, R.; Paixao, T.M.; Mutz, F.; et al. Self-driving cars: A survey. *Expert Syst. Appl.* **2021**, *165*, 113816. [CrossRef]
3. Wang, L.; Gao, R.; Váncza, J.; Krüger, J.; Wang, X.V.; Makris, S.; Chryssolouris, G. Symbiotic human-robot collaborative assembly. *CIRP Ann.* **2019**, *68*, 701–726. [CrossRef]
4. Eisenbach, M.; Aganian, D.; Köhler, M.; Stephan, B.; Schröter, C.; Gross, H.M. Visual Scene Understanding for Enabling Situation-Aware Cobots. In Proceedings of the International Conference on Automation Science and Engineering (CASE), IEEE, Lyon, France, 23–27 August 2021.
5. Gal, Y.; Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In Proceedings of the International Conference on Machine Learning (ICML), PMLR, New York, NY, USA, 19–24 June 2016; pp. 1050–1059.

6. Kendall, A.; Gal, Y. What uncertainties do we need in bayesian deep learning for computer vision? In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017; Volume 31, pp. 5580–5590.
7. Gawlikowski, J.; Tassi, C.R.N.; Ali, M.; Lee, J.; Humt, M.; Feng, J.; Kruspe, A.; Triebel, R.; Jung, P.; Roscher, R.; et al. A survey of uncertainty in deep neural networks. *Artif. Intell. Rev.* **2023**, *56*, 1513–1589. [CrossRef]
8. Abdar, M.; Pourpanah, F.; Hussain, S.; Rezazadegan, D.; Liu, L.; Ghavamzadeh, M.; Fieguth, P.; Cao, X.; Khosravi, A.; Acharya, U.R.; et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Inf. Fusion* **2021**, *76*, 243–297. [CrossRef]
9. Mena, J.; Pujol, O.; Vitrià, J. A survey on uncertainty estimation in deep learning classification systems from a bayesian perspective. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–35. [CrossRef]
10. He, W.; Jiang, Z. A survey on uncertainty quantification methods for deep neural networks: An uncertainty source perspective. *arXiv* **2023**, arXiv:2302.13425.
11. Kraus, F.; Dietmayer, K. Uncertainty estimation in one-stage object detection. In Proceedings of the 2019 IEEE Intelligent Transportation Systems Conference (itsc), IEEE, Auckland, New Zealand, 27–30 October 2019; pp. 53–60.
12. Harakeh, A.; Smart, M.; Waslander, S.L. Bayesod: A bayesian approach for uncertainty estimation in deep object detectors. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Paris, France, 31 May–31 August 2020; pp. 87–93.
13. Gasperini, S.; Haug, J.; Mahani, M.A.N.; Marcos-Ramiro, A.; Navab, N.; Busam, B.; Tombari, F. CertainNet: Sampling-free uncertainty estimation for object detection. *IEEE Robot. Autom. Lett.* **2021**, *7*, 698–705. [CrossRef]
14. Schubert, M.; Kahl, K.; Rottmann, M. Metadetect: Uncertainty quantification and prediction quality estimates for object detection. In Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN), IEEE, Shenzhen, China, 18–22 July 2021; pp. 1–10.
15. Jungo, A.; Reyes, M. Assessing reliability and challenges of uncertainty estimations for medical image segmentation. In Proceedings of the Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, 13–17 October 2019; Proceedings, Part II 22; Springer: Berlin/Heidelberg, Germany, 2019; pp. 48–56.
16. Mehrtash, A.; Wells, W.M.; Tempany, C.M.; Abolmaesumi, P.; Kapur, T. Confidence calibration and predictive uncertainty estimation for deep medical image segmentation. *IEEE Trans. Med. Imaging* **2020**, *39*, 3868–3878. [CrossRef]
17. Hu, P.; Sclaroff, S.; Saenko, K. Uncertainty-aware learning for zero-shot semantic segmentation. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 21713–21724.
18. Huang, P.Y.; Hsu, W.T.; Chiu, C.Y.; Wu, T.F.; Sun, M. Efficient uncertainty estimation for semantic segmentation in videos. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 520–535.
19. Guo, H.; Wang, H.; Ji, Q. Uncertainty-guided probabilistic transformer for complex action recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 20052–20061.
20. Vilnis, L.; McCallum, A. Word representations via gaussian embedding. *arXiv* **2014**, arXiv:1412.6623.
21. Shi, Y.; Jain, A.K.; Kalka, N.D. Probabilistic Face Embeddings. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 6901–6910.
22. Taha, A.; Chen, Y.T.; Yang, X.; Misu, T.; Davis, L. Exploring uncertainty in conditional multi-modal retrieval systems. *arXiv* **2019**, arXiv:1901.07702.
23. Taha, A.; Chen, Y.T.; Misu, T.; Shrivastava, A.; Davis, L. Unsupervised data uncertainty learning in visual retrieval systems. *arXiv* **2019**, arXiv:1902.02586.
24. Hama, K.; Matsubara, T.; Uehara, K.; Cai, J. Exploring uncertainty measures for image-caption embedding-and-retrieval task. *ACM Trans. Multimed. Comput. Commun. Appl. (TOMM)* **2021**, *17*, 1–19. [CrossRef]
25. Dou, Z.; Wang, Z.; Chen, W.; Li, Y.; Wang, S. Reliability-aware prediction via uncertainty learning for person image retrieval. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 588–605.
26. Malinin, A.; Gales, M.J.F. Predictive Uncertainty Estimation via Prior Networks. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017; Volume 31, pp. 7047–7058.
27. Nandy, J.; Hsu, W.; Lee, M.L. Towards Maximizing the Representation Gap between In-Domain and Out-of-Distribution Examples. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Virtual, 6–12 December 2020; Volume 33, pp. 9239–9250.
28. An, S.; Jammalamadaka, N.; Chong, E. Maximum entropy information bottleneck for uncertainty-aware stochastic embedding. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17–24 June 2023; pp. 3809–3818.
29. Scott, T.R.; Ridgeway, K.; Mozer, M.C. Stochastic prototype embeddings. *arXiv* **2019**, arXiv:1909.11702.
30. Neelakantan, A.; Shankar, J.; Passos, A.; McCallum, A. Efficient non-parametric estimation of multiple embeddings per word in vector space. *arXiv* **2015**, arXiv:1504.06654.
31. Chang, J.; Lan, Z.; Cheng, C.; Wei, Y. Data uncertainty learning in face recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 5710–5719.
32. Chen, K.; Lv, Q.; Yi, T. Fast and reliable probabilistic face embeddings in the wild. *arXiv* **2021**, arXiv:2102.04075.

33. Li, S.; Xu, J.; Xu, X.; Shen, P.; Li, S.; Hooi, B. Spherical confidence learning for face recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 19–25 June 2021; pp. 15629–15637.

34. Zhu, Q.; Mao, Q.; Zhang, J.; Huang, X.; Zheng, W. Towards A Robust Group-level Emotion Recognition via Uncertainty-Aware Learning. *arXiv* **2023**, arXiv:2310.04306.

35. Sun, J.J.; Zhao, J.; Chen, L.C.; Schroff, F.; Adam, H.; Liu, T. View-invariant probabilistic embedding for human pose. In Proceedings of the Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Proceedings, Part V 16; Springer: Berlin/Heidelberg, Germany, 2020; pp. 53–70.

36. Warburg, F.; Jørgensen, M.; Civera, J.; Hauberg, S. Bayesian triplet loss: Uncertainty quantification in image retrieval. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, BC, Canada, 11–17 October 2021; pp. 12158–12168.

37. Li, X.; Dai, Y.; Ge, Y.; Liu, J.; Shan, Y.; Duan, L.Y. Uncertainty modeling for out-of-distribution generalization. *arXiv* **2022**, arXiv:2202.03958.

38. Karpukhin, I.; Dereka, S.; Kolesnikov, S. Probabilistic embeddings revisited. *Vis. Comput.* **2023**, *40*, 4373–4386. [CrossRef]

39. Liang, S.; Dai, W.; Wei, Y. Uncertainty Learning for Noise Resistant Sketch-Based 3D Shape Retrieval. *IEEE Trans. Image Process.* **2021**, *30*, 8632–8643. [CrossRef] [PubMed]

40. Zhang, B.; Wonka, P. Point cloud instance segmentation using probabilistic embeddings. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 19–25 June 2021; pp. 8883–8892.

41. Li, W.; Huang, X.; Lu, J.; Feng, J.; Zhou, J. Learning probabilistic ordinal embeddings for uncertainty-aware regression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 19–25 June 2021; pp. 13896–13905.

42. Zhou, J.; Tang, Y.; Su, B.; Wu, Y. Unsupervised Embedding Learning from Uncertainty Momentum Modeling. *arXiv* **2021**, arXiv:2107.08892.

43. Park, J.; Lee, J.; Kim, I.J.; Sohn, K. Probabilistic representations for video contrastive learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 14711–14721.

44. Kirchhof, M.; Kasneci, E.; Oh, S.J. Probabilistic contrastive learning recovers the correct aleatoric uncertainty of ambiguous inputs. In Proceedings of the International Conference on Machine Learning (ICML). PMLR, Seattle, WA, USA, 30 November–30 December 2023; pp. 17085–17104.

45. Upadhyay, U.; Karthik, S.; Mancini, M.; Akata, Z. Probvlm: Probabilistic adapter for frozen vison-language models. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2–6 October 2023; pp. 1899–1910.

46. Janiak, D.; Binkowski, J.; Bielak, P.; Kajdanowicz, T. Enhancing Out-of-Distribution Detection Through Stochastic Embeddings in Self-supervised Learning. In Proceedings of the International Conference on Computational Science, Málaga, Spain, 1–4 June 2024; Springer: Berlin/Heidelberg, Germany; pp. 337–351.

47. Deng, Z.; Li, D.; Song, Y.Z.; Xiang, T. Robust Target Training for Multi-Source Domain Adaptation. In Proceedings of the British Machine Vision Conference (BMVC), London, UK, 21–24 November 2022.

48. Fang, B.; Wu, W.; Liu, C.; Zhou, Y.; Song, Y.; Wang, W.; Shu, X.; Ji, X.; Wang, J. Uatvr: Uncertainty-adaptive text-video retrieval. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2–6 October 2023; pp. 13723–13733.

49. Ji, Y.; Wang, J.; Gong, Y.; Zhang, L.; Zhu, Y.; Wang, H.; Zhang, J.; Sakai, T.; Yang, Y. Map: Multimodal uncertainty-aware vision-language pre-training model. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17–24 June 2023; pp. 23262–23271.

50. Wu, X.; Li, H.; Luo, Y.; Cheng, X.; Zhuang, X.; Cao, M.; Fu, K. Uncertainty-aware sign language video retrieval with probability distribution modeling. *arXiv* **2024**, arXiv:2405.19689.

51. Yu, T.; Li, D.; Yang, Y.; Hospedales, T.M.; Xiang, T. Robust person re-identification by modelling feature uncertainty. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 552–561.

52. Zhang, Y.; Zheng, Z.; He, B.; Sun, L. Learning Posterior and Prior for Uncertainty Modeling in Person Re-Identification. *arXiv* **2020**, arXiv:abs/2007.08785.

53. Long, X.; Hu, R.; Xu, X. Variance Weight Distribution Network Based Noise Sample Learning for Robust Person Re-identification. In Proceedings of the Computer Graphics International (CGI), Shanghai, China, 28 August–2 September 2021; pp. 101–112.

54. Li, Z.; Li, Z.; Shi, Y.; Ling, H.; Chen, J.; Wang, R.; Li, P. Uncertainty-Guided Person Search Model with Auxiliary Shallow Feature Exploration. In Proceedings of the ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, Seoul, Republic of Korea, 14–19 April 2024; pp. 4795–4799.

55. Kingma, D.P.; Welling, M. Auto-Encoding Variational Bayes. In Proceedings of the International Conference on Learning Representations (ICLR), Banff, AB, Canada, 14–16 April 2014.

56. Chen, K.; Yi, T.; Lv, Q. Fast and reliable probabilistic face embeddings based on constrained data uncertainty estimation. *Image Vis. Comput.* **2022**, *121*, 104429. [CrossRef]

57. Schwaiger, A.; Sinhamahapatra, P.; Gansloser, J.; Roscher, K. Is uncertainty quantification in deep learning sufficient for out-of-distribution detection? In Proceedings of IJCAI-Workshop on Artificial Intelligence Safety (AISafety), Yokohama, Japan, 7–8 January 2021.

58. Müller, S.; Wengefeld, T.; Trinh, T.Q.; Aganian, D.; Eisenbach, M.; Gross, H.M. A multi-modal person perception framework for socially interactive mobile service robots. *Sensors* **2020**, *20*, 722. [CrossRef]

59. Salehi, M.; Mirzaei, H.; Hendrycks, D.; Li, Y.; Rohban, M.H.; Sabokrou, M. A unified survey on anomaly, novelty, open-set, and out-of-distribution detection: Solutions and future challenges. *arXiv* **2021**, arXiv:2110.14051.

60. Yang, J.; Zhou, K.; Li, Y.; Liu, Z. Generalized out-of-distribution detection: A survey. *Int. J. Comput. Vis.* **2024**, 1–28. [CrossRef]

61. Jin, X.; Lan, C.; Zeng, W.; Chen, Z. Uncertainty-aware multi-shot knowledge distillation for image-based object re-identification. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 11165–11172.

62. Geng, Y.; Han, Z.; Zhang, C.; Hu, Q. Uncertainty-aware multi-view representation learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 19–21 May 2021; Volume 35, pp. 7545–7553.

63. Neculai, A.; Chen, Y.; Akata, Z. Probabilistic compositional embeddings for multimodal image retrieval. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 4547–4557.

64. Kiureghian, A.D.; Ditlevsen, O. Aleatory or epistemic? Does it matter? *Struct. Saf.* **2009**, *31*, 105–112. [CrossRef]

65. Gal, Y.; Ghahramani, Z. Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference. In Proceedings of the International Conference on Learning Representations (ICLR), San Juan, Puerto Rico, 2–4 May 2016.

66. Luo, H.; Gu, Y.; Liao, X.; Lai, S.; Jiang, W. Bag of Tricks and a Strong Baseline for Deep Person Re-Identification. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Long Beach, CA, USA, 16–17 June 2019; pp. 1487–1495.

67. Hermans, A.; Beyer, L.; Leibe, B. In defense of the triplet loss for person re-identification. *arXiv* **2017**, arXiv:1703.07737.

68. Zheng, L.; Shen, L.; Tian, L.; Wang, S.; Wang, J.; Tian, Q. Scalable Person Re-identification: A Benchmark. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1116–1124.

69. He, L.; Liao, X.; Liu, W.; Liu, X.; Cheng, P.; Mei, T. FastReID: A Pytorch Toolbox for General Instance Re-identification. *arXiv* **2020**, arXiv:abs/2006.02631.

70. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.

71. Zhong, Z.; Zheng, L.; Kang, G.; Li, S.; Yang, Y. Random erasing data augmentation. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 13001–13008.

72. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.

73. Ristani, E.; Tomasi, C. Features for multi-target multi-camera tracking and re-identification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 6036–6046.

74. Zheng, L.; Yang, Y.; Hauptmann, A.G. Person re-identification: Past, present and future. *arXiv* **2016**, arXiv:1610.02984.