Uncertainty Estimates and Multi-hypotheses Networks for Optical Flow

Eddy Ilg(✉), Özgün Çiçek, Silvio Galesso, Aaron Klein, Osama Makansi, Frank Hutter, and Thomas Brox

University of Freiburg, Freiburg, Germany
{ilg,cicek,galessos,kleinaa,makansio,fh,brox}@cs.uni-freiburg.de

Abstract. Optical flow estimation can be formulated as an end-to-end supervised learning problem, which yields estimates with a superior accuracy-runtime tradeoff compared to alternative methodology. In this paper, we make such networks estimate their local uncertainty about the correctness of their prediction, which is vital information when building decisions on top of the estimations. For the first time we compare several strategies and techniques to estimate uncertainty in a large-scale computer vision task like optical flow estimation. Moreover, we introduce a new network architecture and loss function that enforce complementary hypotheses and provide uncertainty estimates efficiently with a single forward pass and without the need for sampling or ensembles. We demonstrate the quality of the uncertainty estimates, which is clearly above previous confidence measures on optical flow and allows for interactive frame rates.

Keywords: Convolutional neural networks · Optical flow estimation Uncertainty estimation

1 Introduction

Recent research has shown that deep networks typically outperform handcrafted approaches in computer vision in terms of accuracy and speed. Optical flow estimation is one example: FlowNet [6,12] yields high accuracy optical flow at interactive frame rates, which is relevant for many applications in the automotive domain or for activity understanding.

A valid critique of learning-based approaches is their black-box nature: since all parts of the problem are learned from data, there is no strict understanding

E. Ilg, Ö. Çiçek and S. Galesso—Equal contribution.

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-01234-2_40) contains supplementary material, which is available to authorized users.

on how the problem is solved by the network. Although FlowNet 2.0 [12] was shown to generalize well across various datasets, there is no guarantee that it will also work in different scenarios that contain unknown challenges. In real-world scenarios, such as control of an autonomously driving car, an erroneous decision can be fatal; thus it is not possible to deploy such a system without information about how reliable the underlying estimates are. We should expect an additional estimate of the network's own uncertainty, such that the network can highlight hard cases where it cannot reliably estimate the optical flow or where it must decide among multiple probable hypotheses; see Fig. 1. However, deep networks in computer vision typically yield only their single preferred prediction rather than the parameters of a distribution.

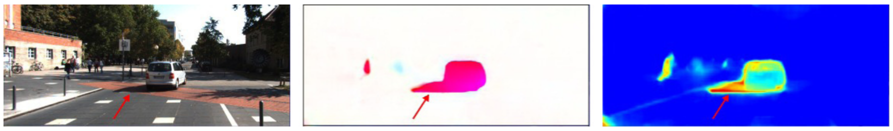
Fig. 1. Joint estimation of optical flow and its uncertainty. **Left:** Image from a KITTI 2015 sequence. **Middle:** Estimated optical flow. **Right:** The estimated uncertainty (visualized as heatmap) marks the optical flow in the shadow of the car as unreliable (pointed by the red arrow), contrary to the car itself, which is estimated with higher certainty. Marked as most reliable is the optical flow for the static background. (Color figure online)

The first contribution of this paper is an answer to the open question which of the many approaches for uncertainty estimation, most of which have been applied only to small problems so far, are most efficient for high-resolution encoder-decoder regression networks. We provide a comprehensive study of empirical ensembles, predictive models, and predictive ensembles. The first category comprises frequentist methods, the second one relies on the estimation of a parametric output distribution, and the third one combines the properties of the previous two. We implemented these approaches for FlowNet using the common MC dropout technique [7], the less common Bootstrapped Ensembles [19] and snapshot ensembles [11]. We find that in general all these approaches yield surprisingly good uncertainty estimates, where the best performance is achieved with uncertainty estimates derived from Bootstrapped Ensembles of predictive networks.

While such ensembles are a good way to obtain uncertainty estimates, they must run multiple networks to create sufficiently many samples. This drawback increases the computational load and memory footprint at training and test time linearly with the number of samples, such that these approaches are not applicable in real-time.

As a second contribution, we present a multi-headed network architecture that yields multiple hypotheses in a single network without the need of sampling. We use a loss that only penalizes the best hypothesis. This pushes the network to make multiple different predictions in case of doubt. We train a second network to optimally combine the hypotheses and to estimate the final uncertainty. This network yields the same good uncertainty estimates as Bootstrapped Ensembles, but allows for interactive frame rates. Thus, in this paper, we address all three important aspects for deployment of optical flow estimation in automotive systems: high accuracy inherited from the base network, a measure of reliability, and a fast runtime.

2 Related Work

Confidence Measures for Optical Flow. While there is a large number of optical flow estimation methods, only few of them provide uncertainty estimates.

Post-hoc methods apply post-processing to already estimated flow fields. Kondermann et al. [16] used a learned linear subspace of typical displacement neighborhoods to test the reliability of a model. In their follow-up work [17], they proposed a hypothesis testing method based on probabilistic motion models learned from ground-truth data. Aodha et al. [1] trained a binary classifier to predict whether the endpoint error of each pixel is bigger or smaller than a certain threshold and used the predicted classifier's probability as an uncertainty measure. All post-hoc methods ignore information given by the model structure.

Model-inherent methods, in contrast, produce their uncertainty estimates using the internal estimation model, i.e., energy minimization models. Bruhn and Weickert [3] used the inverse of the energy functional as a measure of the deviation from the model assumptions. Kybic and Nieuwenhuis [18] performed bootstrap sampling on the data term of an energy-based method in order to obtain meaningful statistics of the flow prediction. The most recent work by Wannenwetsch et al. [29] derived a probabilistic approximation of the posterior of the flow field from the energy functional and computed flow mean and covariance via Bayesian optimization. Ummenhofer et al. [28] presented a depth estimation CNN that internally uses a predictor for the deviation of the estimated optical flow from the ground-truth. This yields a confidence map for the intermediate optical flow that is used internally within the network. However, this approach treats flow and confidence separately and there was no evaluation for the reliability of the confidence measure.

Uncertainty Estimation with CNNs. Bayesian neural networks (BNNs) have been shown to obtain well-calibrated uncertainty estimates while maintaining the properties of standard neural networks [22,24]. Early work [24] mostly used Markov Chain Monte Carlo (MCMC) methods to sample networks from the distribution of the weights, where some, for instance Hamiltonian Monte Carlo, can make use of the gradient information provided by the backpropagation algorithm. More recent methods generalize traditional gradient based MCMC methods to the stochastic mini-batch setting, where only noisy estimates of the true gradient are available [5,30]. However, even these recent MCMC methods do not scale well to high-dimensional spaces, and since contemporary encoder-decoder networks like FlowNet have millions of weights, they do not apply in this setting.

Instead of sampling, variational inference methods try to approximate the distribution of the weights by a more tractable distribution [2,8]. Even though they usually scale much better with the number of datapoints and the number of weights than their MCMC counterparts, they have been applied only to much smaller networks [2,10] than in the present paper.

Gal and Ghahramani [7] sampled the weights by using dropout after each layer and estimated the *epistemic* uncertainty of neural networks. In a follow-up work by Kendall and Gal [15], this idea was applied to vision tasks, and the *aleatoric* uncertainty (which explains the noise in the observations) and the epistemic uncertainty (which explains model uncertainty) were studied in a joint framework. We show in this paper, that the dropout strategy used in all previous computer vision applications [15,26] is not the best one per-se, and other strategies yield better results.

In contrast to Bayesian approaches, such as MCMC sampling, bootstrapping is a frequentist method that is easy to implement and scales nicely to high-dimensional spaces, since it only requires point estimates of the weights. The idea is to train M neural networks independently on M different bootstrapped subsets of the training data and to treat them as independent samples from the weight distribution. While bootstrapping does not ensure diversity of the models and in the worst case could lead to M identical models, Lakshminarayanan et al. [19] argued that ensemble model averaging can be seen as dropout averaging. They trained individual networks with random initialization and random data shuffling, where each network predicts a mean and a variance. During test time, they combined the individual model predictions to account for the epistemic uncertainty of the network. We also consider so-called *snapshot ensembles* [11] in our experiments. These are obtained rather efficiently via Stochastic Gradient Descent with warm Restarts (SGDR) [21].

Multi-hypotheses Estimation. The loss function for the proposed multi-hypotheses network is related to Guzman-Rivera et al. [9], who proposed a similar loss function for SSVMs. Lee et al. [20] applied the loss to network ensembles and Chen and Koltun [4] to a single CNN.

3 Uncertainty Estimation with Deep Networks

Assume we have a dataset $\mathcal{D} = \{(\mathbf{x}_0, \mathbf{y}_0^{\text{gt}}), \ldots, (\mathbf{x}_N, \mathbf{y}_N^{\text{gt}})\}$, which is generated by sampling from a joint distribution $p(\mathbf{x}, \mathbf{y})$. In CNNs, it is assumed that there is a unique mapping from $\mathbf{x}$ to $\mathbf{y}$ by a function $f_{\mathbf{w}}(\mathbf{x})$, which is parametrized by weights $\mathbf{w}$ that are optimized according to a given loss function on $\mathcal{D}$.

For optical flow, we denote the trained network as a mapping from the input images $\mathbf{x} = (\mathbf{I}_1, \mathbf{I}_2)$ to the output optical flow $\mathbf{y} = (\mathbf{u}, \mathbf{v})$ as $\mathbf{y} = f_{\mathbf{w}}(\mathbf{I}_1, \mathbf{I}_2)$, where $\mathbf{u}, \mathbf{v}$ are the x- and y-components of the optical flow. The FlowNet by Dosovitskiy et al. [6] minimizes the per-pixel endpoint error

$$\text{EPE} = \sqrt{(u - u^{\text{gt}})^2 + (v - v^{\text{gt}})^2}, \tag{1}$$

where the pixel coordinates are omitted for brevity. This network, as depicted in Fig. 2a, is fully deterministic and yields only the network's preferred output $\mathbf{y} = f_{\mathbf{w}}(\mathbf{x})$. Depending on the loss function, this typically corresponds to the mean of the distribution $p(\mathbf{y}|\mathbf{x}, \mathcal{D})$. In this paper, we investigate three major approaches to estimate also the variance σ^2. These are based on the empirical variance of the distribution of an ensemble, a parametric model of the distribution, and a combination of both. The variance in all these approaches serves as an estimate of the uncertainty.

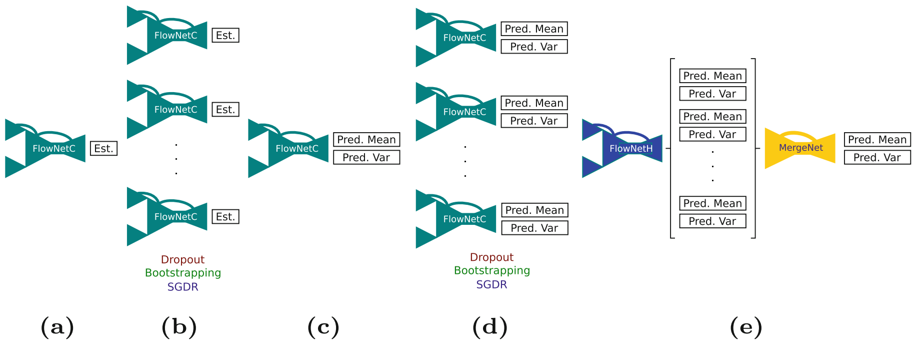

Fig. 2. Overview of the networks considered in this paper. **(a)** FlowNetC trained with EPE. **(b)** Same network as (a), where an ensemble is built using dropout, bootstrapping or SGDR. **(c)** FlowNetC trained with -log-likelihood to predict mean and variance. **(d)** Same network as (c), where an ensemble is built using dropout, bootstrapping or SGDR. **(e)** FlowNetH trained to predict multiple hypotheses with variances, which are merged to a single distributional output. Only **(a)** exists in this form for optical flow.

3.1 Empirical Uncertainty Estimation

A straightforward approach to get variance estimates is to train M different models independently, such that the mean and the variance of the distribution $p(\mathbf{y}|\mathbf{x}, \mathcal{D})$ can be approximated with the empirical mean and variance of the individual model's predictions. Let $f_{\mathbf{w}_i}(\mathbf{x})$ denote model i of an ensemble of M models with outputs $\mathbf{u}_{\mathbf{w}_i}$ and $\mathbf{v}_{\mathbf{w}_i}$. We can compute the empirical mean and variance for the $\mathbf{u}$-component by:

$$\boldsymbol{\mu}_{\mathbf{u}} = \frac{1}{M} \sum_{i=1}^{M} \mathbf{u}_{\mathbf{w}_i}(\mathbf{x}) \tag{2}$$

$$\boldsymbol{\sigma}^2_{\mathbf{u}} = \frac{1}{M} \sum_{i=1}^{M} (\mathbf{u}_{\mathbf{w}_i}(\mathbf{x}) - \boldsymbol{\mu}_{\mathbf{u}})^2 \tag{3}$$

and accordingly for the $\mathbf{v}$-component of the optical flow. Such an ensemble of M networks, as depicted in Fig. 2b, can be built in multiple ways. The most common way is via Monte Carlo Dropout [7]. Using dropout also at test time, it is possible to randomly sample from network weights M times to build an ensemble. Alternatively, ensembles of individual networks can be trained with random weight initialization, data shuffling, and bootstrapping as proposed by Lakshminarayanan et al. [19]. A more efficient way of building an ensemble is to use M pre-converged snapshots of a single network trained with the SGDR [21] learning scheme, as proposed by Huang et al. [11]. We investigate these three ways of building ensembles for flow estimation and refer to them as Dropout, Bootstrapped Ensembles and SGDR Ensembles, respectively.

3.2 Predictive Uncertainty Estimation

Alternatively, we can train a network to output the parameters $\boldsymbol{\theta}$ of a parametric model of the distribution $p(\mathbf{y}|\mathbf{x}, \mathcal{D})$ as introduced by Nix and Weigend [25]. In the literature, Gaussian distributions (where $\boldsymbol{\theta}$ parameterizes the distribution's mean and the variance) are most common, but any type of parametric distribution is possible. Such networks can be optimized by maximizing their log-likelihood:

$$\log p(\mathcal{D} \mid \mathbf{w}) = \frac{1}{N} \sum_{i=1}^{N} \log p(\mathbf{y}_i \mid \boldsymbol{\theta}(\mathbf{x}_i, \mathbf{w})) \tag{4}$$

w.r.t. $\mathbf{w}$. The predictive distribution for an input $\mathbf{x}$ is then defined as:

$$p(\mathbf{y} \mid \mathbf{x}, \mathbf{w}) \equiv p(\mathbf{y} \mid \boldsymbol{\theta}(\mathbf{x}, \mathbf{w})). \tag{5}$$

While negative log-likelihood of a Gaussian corresponds to L_2 loss, FlowNet is trained with an EPE loss, which has more robustness to outliers. Thus, we model the predictive distribution by a Laplacian, which corresponds to an L_1 loss. The univariate Laplace distribution has two parameters a and b and is defined as:

$$\mathcal{L}(u|a, b) = \frac{1}{2b} e^{-\frac{|u-a|}{b}}. \tag{6}$$

As Wannewetsch et al. [29], we model the u and v components of the optical flow to be independent. The approximation yields:

$$\mathcal{L}(u, v|a_u, a_v, b_u, b_v) \approx \mathcal{L}(u|a_u, b_u) \cdot \mathcal{L}(v|a_v, b_v). \tag{7}$$

We obtain a probabilistic version of FlowNet with outputs a_u, a_v, b_u, b_v by minimizing the negative log-likelihood of Eq. 7:

$$-\log(\mathcal{L}(u|a_u, b_u) \cdot \mathcal{L}(v|a_v, b_v)) = \frac{|u - a_u|}{b_u} + \log b_u + \frac{|v - a_v|}{b_v} + \log b_v. \tag{8}$$

As an uncertainty estimate we use the variance of the predictive distribution, which is $\sigma^2 = 2b^2$ in this case. This case corresponds to a single FlowNetC predicting flow and uncertainty as illustrated in Fig. 2c.

3.3 Bayesian Uncertainty Estimation

From a Bayesian perspective, to obtain an estimate of model uncertainty, rather than choosing a point estimate for $\mathbf{w}$, we would marginalize over all possible values:

$$p(\mathbf{y} \mid \mathbf{x}, \mathcal{D}) = \int p(\mathbf{y} \mid \mathbf{x}, \mathbf{w}) p(\mathbf{w} \mid \mathcal{D}) d\mathbf{w} \tag{9}$$

$$= \int p(\mathbf{y} \mid \boldsymbol{\theta}(\mathbf{x}, \mathbf{w})) p(\mathbf{w} \mid \mathcal{D}) d\mathbf{w}. \tag{10}$$

This integral cannot be computed in closed form, but by sampling M networks $\mathbf{w}_i \sim p(\mathbf{w}|\mathcal{D})$ from the posterior distribution and using a Monte-Carlo approximation [24], we can approximate its mean and variance as:

$$p(\mathbf{y} \mid \mathbf{x}, \mathcal{D}) \approx \sum_{i=1}^{M} p(\mathbf{y} \mid \boldsymbol{\theta}(\mathbf{x}, \mathbf{w}_i)). \tag{11}$$

Since every parametric distribution has a mean and a variance, also the distributions predicted by each individual network with weights $\mathbf{w}_i$ yield a mean $\boldsymbol{\mu}_i$ and a variance $\boldsymbol{\sigma}_i^2$. The mean and variance of the mixture distribution in Eq. 11 can then be computed by the law of total variance for the $\mathbf{u}$-component (as well as for the $\mathbf{v}$-component) as:

$$\boldsymbol{\mu}_{\mathbf{u}} = \frac{1}{M} \sum_{i=1}^{M} \boldsymbol{\mu}_{\mathbf{u},i} \tag{12}$$

$$\boldsymbol{\sigma}_{\mathbf{u}}^2 = \frac{1}{M} \sum_{i=1}^{M} \Big((\boldsymbol{\mu}_{\mathbf{u},i} - \boldsymbol{\mu}_{\mathbf{u}})^2 + \boldsymbol{\sigma}^2{}_{\mathbf{u},i} \Big). \tag{13}$$

This again can be implemented as ensembles obtained by predictive variants of dropout [7], bootstrapping [19] or SGDR [11], where the ideas from Sects. 3.1 and 3.2 are combined as shown in Fig. 2d.

4 Predicting Multiple Hypotheses Within a Single Network

The methods presented in the Sects. 3.1 and 3.3 require multiple forward passes to obtain multiple samples with the drawback of a much increased computational cost at runtime. In this section, we propose a loss function to make multiple predictions within a single network. We call these predictions *hypotheses*. For the predicted hypotheses, we encourage multimodality by the design of the loss function [4,9,20]. This makes the predictions more diverse and leads to capturing more different solutions, but does not allow for merging by simply computing the mean as for the ensembles presented in the last section. Therefore, we also propose to use a second network that merges the hypotheses to a single prediction and variance, as depicted in Fig. 2e.

Since a ground-truth is available only for the single true solution, the question arises of how to train a network to predict multiple hypotheses and how to ensure that each hypothesis comprises meaningful information. To this end, we use a loss that punishes only the best among the network output hypotheses $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_M$ [9]. Let the loss between a predicted flow vector $\mathbf{y}(i,j)$ and its ground-truth $\mathbf{y}^{\mathrm{gt}}(i,j)$ at pixel i,j be defined by a loss functon l. We minimize:

$$L_{hyp} = \sum_{i,j} l(\boldsymbol{y}_{\mathrm{best\_idx}(i,j)}, \boldsymbol{y}^{\mathrm{gt}}(i,j)) + \varDelta(i,j), \tag{14}$$

where best_idx(i,j) selects the best hypothesis per pixel according to the ground-truth:

$$\mathrm{best\_idx}(i,j) = \operatorname*{argmin}_k \left[\mathrm{EPE}(\boldsymbol{y}_k(i,j), \boldsymbol{y}^{\mathbf{gt}}(i,j))\right]. \tag{15}$$

$\varDelta = \varDelta_u + \varDelta_v$ encourages similar solutions to be from the same hypothesis k via one-sided differences, e.g. for the $\mathbf{u}$ component:

$$\begin{aligned} \varDelta_u(i,j) = &\sum_{k;i>1;j} |y_{k,u}(i,j) - y_{k,u}(i-1,j)| \\ &+ \sum_{k;i;j>1} |y_{k,u}(i,j) - y_{k,u}(i,j-1)| \end{aligned} \tag{16}$$

For l, we either use the endpoint error from Eq. 1 or the negative log-likelihood from Eq. 8. In the latter case, each hypothesis is combined with an uncertainty estimation and l also operates on a variance $\boldsymbol{\sigma}$. Equations 15 and 16 remain unaffected. For the best index selection we stick to the EPE since it is the main optimization goal.

To minimize L_{hyp}, the network must make a prediction close to the ground-truth in at least one of the hypotheses. In locations where multiple solutions exist and the network cannot decide for one of them, the network will predict several different likely solutions to increase the chance that the true solution is among these predictions. Consequently, the network will favor making diverse hypotheses in cases of uncertainty. In Tables 3 and 4 of the supplemental material we provide visualizations of such hypotheses.

In principle, L_{hyp} could collapse to use only one of the hypotheses' outputs. In this case the other hypotheses would have very high error and would never be selected for back-propagation. However, due to the variability in the data and the stochasticity in training, such collapse is very unlikely. We never observed that one of the hypotheses was not used by the network, and for the oracle merging we observed that all hypotheses contribute more or less equally. We show this diversity in our experiments.

5 Experiments

To evaluate the different strategies for uncertainty estimation while keeping the computational cost tractable, we chose as a base model the FlowNetC architecture from Dosovitsky et al. [6] with improved training settings by Ilg et al. [12] and by us. A single FlowNetC shows a larger endpoint error (EPE) than the full, stacked FlowNet 2.0 [12], but trains much faster. Note that this work aims for uncertainty estimation and not for improving the optical flow over the base model. The use of ensembles may lead to minor improvements of the optical flow estimates due to the averaging effect, but these improvements are not of major concern here. In the end, we will also show results for a large stacked network to demonstrate that the uncertainty estimation as such is not limited to small, simple networks.

5.1 Training Details

In contrast to Ilg et al. [12], we use Batch Normalization [13] and a continuously dropping cosine learning rate schedule [21]. This yields shorter training times and improves the results a little; see Table 1. We train on FlyingChairs [6] and start with a learning rate of $2e-4$. For all networks, we fix a training budget of 600k iterations per network, with an exception for SGDR, where we also evaluate performing some pre-cycles. For SGDR Ensembles, we perform restarts every 75k iterations. We fix the T_{mult} to 1, so that each annealing takes the same number of iterations. We experiment with different variants of building ensembles using snapshots at the end of each annealing. We always take the latest M snapshots when building an ensemble. For dropout experiments, we use a dropout ratio of 0.2 as suggested by Kendall et al. [15]. For Bootstrapped Ensembles, we train M FlowNetC in parallel with bootstrapping, such that each network sees different 67% of the training data. For the final version of our method, we perform an additional training of 250k iterations on FlyingThings3D [23] per network, starting with a learning rate of $2e-5$ that is decaying with cosine annealing. We use the Caffe [14] framework for network training and evaluate all runtimes on an Nvidia GTX 1080Ti. We will make the source code and the final models publicly available.

Table 1. Optical flow quality on Sintel train clean with the original FlowNetC [12] and our implementation.

	Iter.	EPE
FlowNetC [12]	600k	3.77
FlowNetC [12]	1.2m	3.58
FlowNetC ours	600k	3.40

For the ensembles, we must choose the size M of the ensemble. The sampling error for the mean and the variance decreases with increasing M. However, since networks for optical flow estimation are quite large, we are limited in the tractable sample size and restrict it to $M = 8$. We also use $M = 8$ for FlowNetH.

For SGDR there is an additional pre-cycle parameter: snapshots in the beginning have usually not yet converged and the number of pre-cycles is the number of snapshots we discard before building the ensemble. In the supplemental material we show that the later the snapshots are taken, the better the results are in terms of EPE and AUSE. We use 8 pre-cycles in the following experiments.

5.2 Evaluation Metrics

Sparsification Plots. To assess the quality of the uncertainty measures, we use so-called sparsification plots, which are commonly used for this purpose [1,17, 18,29]. Such plots reveal on how much the estimated uncertainty coincides with the true errors. If the estimated variance is a good representation of the model uncertainty, and the pixels with the highest variance are removed gradually, the error should monotonically decrease. Such a plot of our method is shown in Fig. 3. The best possible ranking of uncertainties is ranking by the true error to the ground-truth. We refer to this curve as *Oracle Sparsification*. Figure 3 reveals that our uncertainty estimate is very close to this oracle.

Sparsification Error. For each approach the oracle is different, hence a comparison among approaches using a single sparsification plot is not possible. To this end, we introduce a measure, which we call *Sparsification Error*. It is defined as the difference between the sparsification and its oracle. Since this measure normalizes the oracle out, a fair comparison of different methods is possible. In Fig. 4a, we show sparsification errors for all methods we present in this paper. To quantify the sparsification error with a single number, we use the Area Under the Sparsification Error curve ($AUSE$).

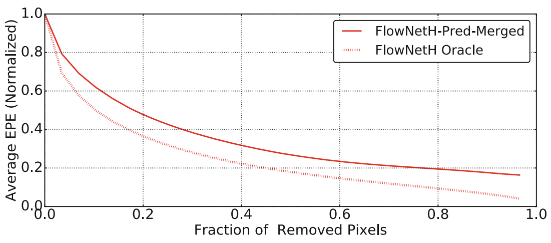

Fig. 3. Sparsification plot of FlowNetH-Pred-Merged for the Sintel train clean dataset. The plot shows the average endpoint error (AEPE) for each fraction of pixels having the highest uncertainties removed. The oracle sparsification shows the lower bound by removing each fraction of pixels ranked by the ground-truth endpoint error. Removing 20 percent of the pixels results in halving the average endpoint error.

Oracle EPE. For each ensemble, we also compute the hypothetical endpoint error by considering the pixel-wise best selection from each member (decided by the ground-truth). We report this error together with the empirical variances among the members in Table 2.

5.3 Comparison Among Uncertainties from CNNs

Nomenclature. When a single network is trained against the endpoint error, we refer to this single network and the resulting ensemble as empirical (abbreviated as *Emp*; Figs. 2a and b), while when the single network is trained against the negative log-likelihood, we refer to the single network and the ensemble as predictive (*Pred*; Figs. 2c and d). When multiple samples or solutions are merged with

Table 2. Comparison of flow and uncertainty predictions of all proposed methods with $M = 8$ on the Sintel train clean dataset. Oracle-EPE is the EPE of the pixel-wise best selection from the samples or hypotheses determined by the ground-truth. Var. is the average empirical variance over the 8 samples or hypotheses. Predictive versions (Pred) generally outperform empirical versions (Emp). Including a merging network increases the performance. FlowNetH-Pred-Merged performs best for predicting uncertainties and has a comparatively low runtime.

	Empirical (Emp)				Predictive (Pred)				Runtime
	AUSE	EPE	Oracle EPE	Var.	AUSE	EPE	Oracle EPE	Var.	
FlowNetC	-	3.40	-	-	0.133	3.62	-	-	**38** ms
Dropout	0.212	3.67	2.56	5.05	0.158	3.99	2.96	3.80	320 ms
SGDREnsemble	**0.191**	3.25	2.56	3.50	0.134	3.40	2.87	1.52	304 ms
BootstrappedEnsemble	0.209	3.41	2.17	9.52	0.127	3.46	2.49	6.15	304 ms
BootstrappedEnsemble-Merged					0.102	3.20	2.49	6.15	332 ms
FlowNetH-Merged	-	3.50	1.73	**83.32**	**0.095**	3.36	1.89	**52.85**	60 ms

a network, we add *Merged* to the name. E.g. FlowNetH-Pred-Merged refers to a FlowNetH that predicts multiple hypotheses and merges them with a network, using the loss for a predictive distribution for both, hypotheses and merging, respectively (Fig. 2e). Table 2 and Figs. 4a, b show results for all models evaluated in this paper.

Empirical Uncertainty Estimation. The results show that uncertainty estimation with empirical ensembles is good, but worse than the other methods presented in this paper. However, in comparison to predictive counterparts,

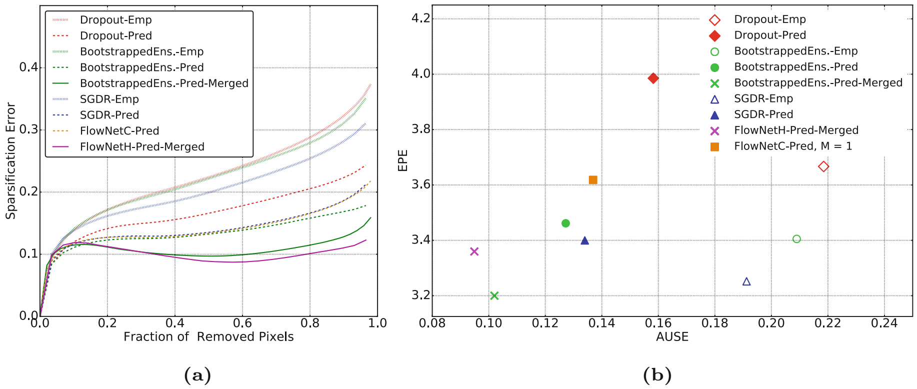

Fig. 4. (a) Sparsification error on the Sintel train clean dataset. The sparsification error (smaller is better) is the proposed measure for comparing the uncertainty estimates among different methods. FlowNetH-Pred-Merged and BootstrappedEnsemble-Pred-Merged perform best in almost all sections of the plot. **(b)** Scatter plot of AEPE vs. AUSE for the tested approaches visualizing some content of Table 2.

empirical ensembles tend to yield slightly better EPEs, as will be discussed in the following.

Predictive Uncertainty Estimation. The estimated uncertainty is better with predictive models than with the empirical ones. Even a single FlowNetC with predictive uncertainty yields much better uncertainty estimates than any empirical ensemble in terms of AUSE. This is because when training against a predictive loss function, the network has the possibility to explain outliers with the uncertainty. This is known as *loss attenuation* [15]. While the EPE loss tries to enforce correct solutions also for outliers, the log-likelihood loss attenuates them. The experiments confirm this effect and show that it is advantageous to let a network estimate its own uncertainty.

Predictive Ensembles. Comparing ensembles of predictive networks to a single predictive network shows that a single network is already very close to the predictive ensembles and that the benefit of an ensemble is limited. We attribute this also to loss attenuation: different ensemble members appear to attenuate outliers in a similar manner and induce less diversity, as can be seen by the variance among the members of the ensemble (column 'Var.' in Table 2).

When comparing empirical to predictive ensembles, we can draw the following conclusions: **(a)** empirical estimation provides more diversity within the ensemble (variance column in Table 2), **(b)** empirical estimation provides lower EPEs and Oracle EPEs, **(c)** all empirical setups provide worse uncertainty estimates than predictive setups.

Ensemble Types. We see that the commonly used dropout [7] technique performs worst in terms of EPE and AUSE, although the differences between the predictive ensemble types are not very large. SGDR Ensembles provide better uncertainties, yet the variance among the samples is the smallest. This is likely because later ensemble members are derived from previous snapshots of the same model. Furthermore, because of the 8 pre-cycles, SGDR experiments ran the largest number of training iterations, which could be an explanation why they provide a slightly better EPE than other ensembles. Bootstrapped Ensembles provide the highest sample variance and the lowest AUSE among the predictive ensembles.

FlowNetH and Uncertainty Estimation with Merging Networks. Besides FlowNetH we also investigated putting a merging network on top of the predictive Bootstrapped Ensembles. Results show that the multi-hypotheses network (FlowNetH-Pred-Merged) is on-par with BootstrappedEnsemble-Pred-Merged in terms of AUSE and EPE. However, including the runtime, FlowNetH-Pred-Merged yields the best trade-off; see Table 2. Only FlowNetC and FlowNetH-Pred-Merged allow a deployment at interactive frame rates. Table 2 also shows that FlowNetH has a much higher sample variance and the lowest oracle EPE. This indicates that it internally has very diverse and potentially useful hypotheses that could be exploited better in the future. For some visual examples, we refer to Tables 3 and 4 in the supplemental material.

5.4 Comparison to Energy-Based Uncertainty Estimation

We compare the favored approach from the previous section (FlowNetH-Pred-Merged) to ProbFlow [29], which is an energy minimization approach and currently the state-of-the-art for estimating the uncertainty of optical flow. Figure 5 shows the sparsification plots for the Sintel train final. ProbFlow has almost the same oracle as FlowNetH-Pred-Merged, i.e. the flow field from ProbFlow can equally benefit from sparsification, but the actual sparsification error due to its estimated uncertainty is higher. This shows that FlowNetH-Pred-Merged has superior uncertainty estimates. In Table 3 we show that this also holds for the KITTI dataset. FlowNetH outperforms ProbFlow also in terms of EPE in this case. This shows that the superior uncertainty estimates are not due to a weaker optical flow model, i.e. from obvious mistakes that are easy to predict.

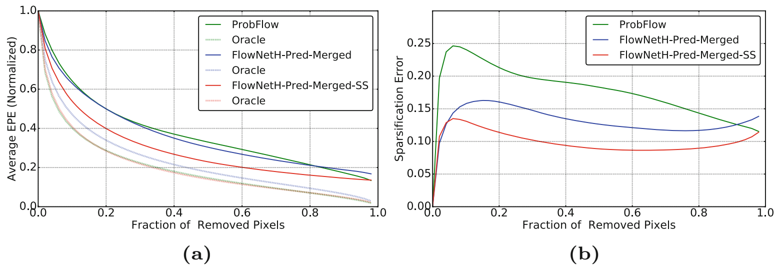

Fig. 5. Plots of the sparsification curves with their respective oracles **(a)** and of the sparsification errors **(b)** for ProbFlow, FlowNetH-Pred-Merged and FlowNetH-Pred-Merged-SS (version with 2 refinement networks stacked on top) on the Sintel train final dataset. KITTI versions are similar and provided in the supplemental material.

Table 3. Comparison of FlowNetH to the state-of-the-art uncertainty estimation method ProbFlow [29] on the Sintel train clean, Sintel train final and our KITTI 2012+2015 validation split datasets. The '-FT-KITTI' version is trained on FlyingChairs [6] first and then on FlyingThings3D [23], as described in Sect. 5.1 and subsequently fine-tuned on our KITTI 2012+2015 training split. FlowNetH-Pred-Merged, -S and -SS are all trained with the FlowNet2 [12] schedule described in supplemental material Fig. 6. Our method outperforms ProbFlow in AUSE by a large margin and also in terms of EPE for the KITTI dataset. †runtime taken from [29], please see the supplemental material for details on the computation of the ProbFlow outputs.

	Sintel clean		Sintel final		KITTI		Runtime
	AUSE	EPE	AUSE	EPE	AUSE	EPE	
ProbFlow [29]	0.162	1.87	0.173	3.34	0.466	8.95	38.1s†
FlowNetH-Pred-Merged-FT-KITTI	-	-	-	-	**0.086**	3.12	**60** ms
FlowNetH-Pred-Merged	0.117	2.58	0.128	3.78	0.151	7.84	**60** ms
FlowNetH-Pred-Merged-S	0.091	2.29	0.098	3.51	0.102	6.86	86 ms
FlowNetH-Pred-Merged-SS	**0.089**	2.19	**0.096**	3.40	0.091	6.50	99 ms

Table 3 further shows that the uncertainty estimation is not limited to simple encoder-decoder networks, but can also be applied successfully to state-of-the-art stacked networks [12]. To this end, we follow Ilg et al. [12] and stack refinement networks on top of FlowNetH-Pred-Merged. Different from [12], each refinement network yields the residual of the flow field and the uncertainty, as recently proposed by [27]. We refer to the network with the 1st refinement network as FlowNetH-Pred-Merged-S and with the second refinement network as FlowNetH-Pred-Merged-SS, since each refinement network is a FlowNetS [12].

The uncertainty estimation is not negatively influenced by the stacking, despite the improving flow fields. This shows again that the uncertainty estimation works reliably notwithstanding if the predicted optical flow is good or bad.

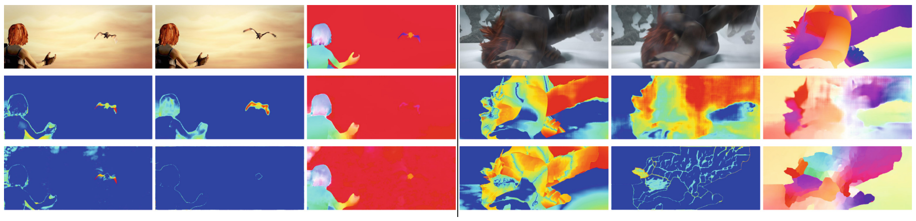

Fig. 6. Comparison between FlowNetH-Pred-Merged and ProbFlow [29]. The first row shows the image pair followed by its ground-truth flow for two different scenes from the Sintel final dataset. The second row shows FlowNetH-Pred-Merged results: entropy from a Laplace distribution with ground-truth error (we refer to this as *Oracle Entropy* to represent the optimal uncertainty as explained in the supplemental material), predicted entropy and predicted flow. Similar to the second row, the third row shows the results for ProbFlow. Although both methods fail at estimating the motion of the dragon on the left scene and the motion of the arm and the leg in the right scene, our method is better at predicting the uncertainties in these regions.

Figure 6 shows a qualitative comparison to ProbFlow. Clearly, the uncertainty estimate of FlowNet-Pred-Merged also performs well outside motion boundaries and covers many other causes for brittle optical flow estimates. More results on challenging real-world data are shown in the supplemental video which can also be found on https://youtu.be/HvyovWSo8uE.

6 Conclusion

We presented and evaluated several methods to estimate the uncertainty of deep regression networks for optical flow estimation. We showed that SGDR and Bootstrapped Ensembles perform better than the commonly used dropout technique. Furthermore, we found that a single network can estimate its own uncertainty surprisingly well and that this estimate outperforms every empirical ensemble. We believe that these results will apply to many other computer vision tasks, too. Moreover, we presented a multi-hypotheses network that shows very good performance and is faster than sampling-based approaches and ensembles. The fact that networks can estimate their own uncertainty reliably and in real-time is of high practical relevance. Humans tend to trust an engineered method much more than a trained network, of which nobody knows exactly how it solves the task. However, if networks say when they are confident and when they are not, we can trust them a bit more than we do today.

Acknowledgements. We gratefully acknowledge funding by the German Research Foundation (SPP 1527 grants BR 3815/8-1 and HU 1900/3-1, CRC-1140 KIDGEM Z02) and by the Horizon 2020 program of the EU via the ERC Starting Grant 716721 and the project Trimbot2020.

References

1. Aodha, O.M., Humayun, A., Pollefeys, M., Brostow, G.J.: Learning a confidence measure for optical flow. IEEE Trans. Pattern Anal. Mach. Intell. **35**(5), 1107–1120 (2013). https://doi.org/10.1109/TPAMI.2012.171
2. Blundell, C., Cornebise, J., Kavukcuoglu, K., Wierstra, D.: Weight uncertainty in neural network. In: Proceedings of the 32nd International Conference on Machine Learning (ICML 2015), pp. 1613–1622 (2015)
3. Bruhn, A., Weickert, J.: A confidence measure for variational optic flow methods. In: Klette, R., Kozera, R., Noakes, L., Weickert, J. (eds.) Geometric Properties for Incomplete data, pp. 283–298. Springer, Dordrecht (2006). https://doi.org/10.1007/1-4020-3858-8_15
4. Chen, Q., Koltun, V.: Photographic image synthesis with cascaded refinement networks. In: IEEE International Conference on Computer Vision (ICCV) (2017)
5. Chen, T., Fox, E., Guestrin, C.: Stochastic gradient Hamiltonian Monte Carlo. In: Proceedings of the 31st International Conference on Machine Learning (ICML 2014) (2014)
6. Dosovitskiy, A., et al.: FlowNet: learning optical flow with convolutional networks. In: IEEE International Conference on Computer Vision (ICCV) (2015)
7. Gal, Y., Ghahramani, Z.: Dropout as a Bayesian approximation: representing model uncertainty in deep learning. In: International Conference on Machine Learning (ICML) (2016)
8. Graves, A.: Practical variational inference for neural networks. In: Advances in Neural Information Processing Systems (NIPS), pp. 2348–2356 (2011)
9. Guzman-Rivera, A., Batra, D., Kohli, P.: Multiple choice learning: learning to produce multiple structured outputs. In: International Conference on Neural Information Processing Systems (NIPS) (2012)
10. Hernández-Lobato, J., Adams, R.: Probabilistic backpropagation for scalable learning of Bayesian neural networks. In: Proceedings of the 32nd International Conference on Machine Learning (ICML 2015) (2015)
11. Huang, G., Li, Y., Pleiss, G.: Snapshot ensembles: Train 1, get M for free. In: International Conference on Learning Representations (ICLR) (2017)
12. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: FlowNet 2.0: evolution of optical flow estimation with deep networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

13. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Bach, F., Blei, D. (eds.) Proceedings of the 32nd International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 37, pp. 448–456. PMLR, Lille, 07–09 July 2015. http://proceedings.mlr.press/v37/ioffe15.html
14. Jia, Y., et al.: Caffe: convolutional architecture for fast feature embedding. In: Proceedings of ACMMM, pp. 675–678 (2014)
15. Kendall, A., Gal, Y.: What uncertainties do we need in Bayesian deep learning for computer vision? In: International Conference on Neural Information Processing Systems (NIPS) (2017)
16. Kondermann, C., Kondermann, D., Jähne, B., Garbe, C.: An adaptive confidence measure for optical flows based on linear subspace projections. In: Hamprecht, F.A., Schnörr, C., Jähne, B. (eds.) DAGM 2007. LNCS, vol. 4713, pp. 132–141. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74936-3_14
17. Kondermann, C., Mester, R., Garbe, C.: A statistical confidence measure for optical flows. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008. LNCS, vol. 5304, pp. 290–301. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88690-7_22
18. Kybic, J., Nieuwenhuis, C.: Bootstrap optical flow confidence and uncertainty measure. Comput. Vis. Image Underst. **115**(10), 1449–1462 (2011). https://doi.org/10.1016/j.cviu.2011.06.008. http://www.sciencedirect.com/science/article/pii/S1077314211001536
19. Lakshminarayanan, B., Pritzel, A., Blundell, C.: Simple and scalable predictive uncertainty estimation using deep ensembles. In: NIPS Workshop (2016)
20. Lee, S., Purushwalkam, S., Cogswell, M., Ranjan, V., Crandall, D., Batra, D.: Stochastic multiple choice learning for training diverse deep ensembles. In: International Conference on Neural Information Processing Systems (NIPS) (2016)
21. Loshchilov, I., Hutter, F.: SGDR: stochastic gradient descent with warm restarts. In: International Conference on Learning Representations (ICLR) (2017)
22. MacKay, D.J.C.: A practical Bayesian framework for backpropagation networks. Neural Comput. **4**(3), 448–472 (1992)
23. Mayer, N., et al.: A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4040–4048, June 2016. https://doi.org/10.1109/CVPR.2016.438
24. Neal, R.: Bayesian learning for neural networks. Ph.D. thesis, University of Toronto (1996)
25. Nix, D.A., Weigend, A.S.: Estimating the mean and variance of the target probability distribution. In: Neural Networks: IEEE World Congress on Computational Intelligence, vol. 1, pp. 55–60, June 1994. https://doi.org/10.1109/ICNN.1994.374138
26. Novotny, D., Larlus, D., Vedaldi, A.: Learning 3D object categories by looking around them. In: IEEE International Conference on Computer Vision (ICCV) (2017)
27. Pang, J., Sun, W., Ren, J.S.J., Yang, C., Yan, Q.: Cascade residual learning: a two-stage convolutional neural network for stereo matching. In: IEEE International Conference on Computer Vision (ICCV) Workshop (2017)
28. Ummenhofer, B., et al.: Demon: Depth and motion network for learning monocular stereo. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017). http://lmb.informatik.uni-freiburg.de//Publications/2017/UZUMIDB17

29. Wannenwetsch, A.S., Keuper, M., Roth, S.: ProbFlow: joint optical flow and uncertainty estimation. In: IEEE International Conference on Computer Vision (ICCV), October 2017
30. Welling, M., Teh, Y.: Bayesian learning via stochastic gradient Langevin dynamics. In: Proceedings of the 28th International Conference on Machine Learning (ICML 2011) (2011)