Bayesian 3D ConvNets for Action Recognition from Few Examples

Martin de la Riva
University of Amsterdam
martin7557@gmail.com

Pascal Mettes
University of Amsterdam
P.S.M.Mettes@uva.nl

Abstract

A core challenge in action recognition from videos is obtaining sufficient training examples to train deep networks. This holds especially for action tasks from non-standard sensors such as infra-red cameras. In this work, we investigate Bayesian 3D ConvNets for action recognition when training examples are scarce. This work connects 3D ConvNets, a state-of-the-art approach for action recognition, with Bayesian networks, which have shown to be effective regularizers for deep networks. We do so by extending Bayes by Backprop to 3D ConvNets. Experimental evaluation on three small-scale action datasets from both RGB and infra-red sensors shows that Bayesian 3D ConvNets have a better test generalizing than standard 3D ConvNets. We find that, the more scarce the number of training examples per action, the better our Bayesian 3D ConvNets performs, highlighting the potential of Bayesian learning in the video domain.

1. Introduction

This work strives for action recognition in videos given limited training data. In action recognition, 3D ConvNets have shown to be a state-of-the-art approach [12, 13, 30]. While effective, 3D ConvNets require a wealth of examples for training. This demand is practically often infeasible because certain actions occur rarely, or because actions are captured by non-standard sensors such as infra-red cameras [5], see Figure 1.

We investigate the potential of Bayesian learning for 3D ConvNets in few-example action recognition. For a Bayesian integration, we focus on Bayes by Backprop [6], a method to perform variational inference in neural networks by introducing uncertainty into the weights of the network. Bayes by Backprop has shown to enable regularization in plain deep networks [1], recurrent neural networks [2], and 2D ConvNets [26]. Surprisingly, Bayes by Backprop has not been investigated in action recognition, while the limited availability of data is apparent in the video domain. This work aims to fill that void.

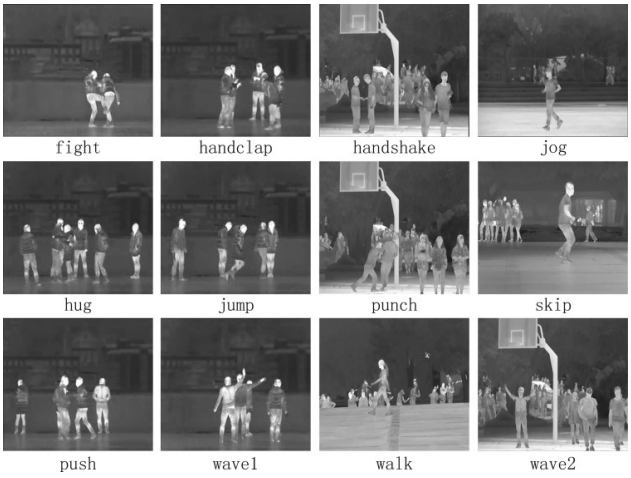


Figure 1: Video examples captured with non-standard sensors such as infrared are quite scarce. We propose Bayesian 3D ConvNets to get the most out of the few examples from such video examples through strong regularization. Image courtesy of [5].

Experiments on three action datasets from both RGB and infra-red inputs shows that replacing standard (frequentist) 3D ConvNets with Bayesian 3D ConvNets results in competitive recognition performance. We also find that Bayesian 3D ConvNets are preferred when reducing the amount of training examples results. Overall, Bayesian 3D ConvNets provide strongly regularized networks for action recognition from videos, with immediate benefits for training from limited examples.

2. Related Work

2.1. Bayesian networks

In deep learning literature, Bayesian inference has been actively researched to deal with the big data demand and overfitting in neural networks [1, 6]. In this work, we expand upon the Bayesian deep learning approach using variational inference. This line of work is characterised by defin-

ing the network as a probabilistic model $P(\mathbf{y}|\mathbf{x}, \mathbf{w})$, where $\mathbf{x} \in$ inputs, $\mathbf{y} \in$ targets, $\mathbf{w} \in$ weights. Since neural networks are large in size, this distribution is computationally intractable and it therefore has to be approximated [6].

Bayes by Backprop [1] is a method that builds upon the variational inference scheme for neural networks introduced by Graves [6], who built upon Hilton and Van Camp [9]. In this method, all the weights are represented as probability distributions instead of single values , adding uncertainty into the weights.

Gal and Ghahramani [4] and Shridhar *et al*. [25, 26] have investigated the extension of Bayes by Backprop to convolutional networks. Where the former adds Dropout layers during training and testing, equivalent to having a Bernoulli posterior with a Gaussian Prior, the latter generates Gaussian distributions directly in its weights. This line of work have been evaluated in the image domain, with results competitive to non-Bayesian approaches. An extension to the spatio-temporal domain is however lacking, while the problems of data demand and overfitting are ever present. In this work, we focus on filling the Bayesian void in action recognition.

2.2. Few-example Action Recognition

In early studies of few shot action recognition, work focused on scripted action datasets such as KTH or Wiezmann [17, 22, 33], for example using Hidden Markov Models [23] or dense trajectories [31]. More recent works employ larger datasets such as UCF101 [28] for few shot action recognition, either taking a portion of the training set (from 10% to 50%) [32] or a subset of the classes [19]. Xu *et al*. [32] propose networks with dense connections [10], which have a regularization effect to prevent overfitting; and dilated temporal convolutions [35] to help explore temporal relations between different snippets. In this work, we start from Bayesian networks, which have readily shown their effectiveness in other visual tasks, and bring them to the video domain, for action recognition with strong regularization.

3. Method

3.1. Bayes By Backprop

Let's start by defining the neural network as a probabilistic model $P(\mathbf{y}|\mathbf{x}, \mathbf{w})$. Given a set of inputs $\mathbf{x} \in \mathcal{X}$ and the set of weights $\mathbf{w}$ of the neural network, it returns a probability for each output $\mathbf{y} \in Y$. We are interested in learning the weights that optimize this probabilistic distribution. These weights $\mathbf{w}$ can be learnt using Maximum Likelihood Estimation (MLE). With $\mathcal{D} = (\mathbf{x}, \mathbf{y})$ a set of training examples:

$$\begin{aligned} \mathbf{w}^{MLE} &= \arg\max_{\mathbf{w}} \log P(\mathcal{D}|\mathbf{w}) \\ &= \arg\max_{\mathbf{w}} \sum_i \log P(\mathbf{y}_i|\mathbf{x}_i, \mathbf{w}). \end{aligned} \tag{1}$$

Regularization can be added into the weights by introducing a prior on them and using Maximum A Posteriori (MAP). Using a Laplace prior would yield L1 regularization while a Gaussian prior would yield L2 regularization (weight decay) [6]:

$$\begin{aligned} \mathbf{w}^{MAP} &= \arg\max_{\mathbf{w}} \log P(\mathbf{w}|\mathcal{D}) \\ &= \arg\max_{\mathbf{w}} \log P(\mathcal{D}|\mathbf{w}) + \log P(\mathbf{w}). \end{aligned} \tag{2}$$

Bayes by Backprop is interested in learning the posterior distribution on the weights given the training examples $P(\mathbf{w}|\mathcal{D})$. In general, computing the true posterior through Bayes' rule $P(\mathbf{w}|\mathcal{D}) = \frac{P(\mathcal{D}|\mathbf{w})P(\mathbf{w})}{P(\mathcal{D})}$ involves computationally intractable integrals for neural networks [1, 6, 9]. This is due to the large number of parameters and the fact that the functional form of a neural network does not lend itself to exact integration.

Hinton and Van Camp [9] and Graves [6] propose to use an approximation $q_\theta(\mathbf{w}|\theta)$ to the true posterior $P(\mathbf{w}|\mathcal{D})$, $q_\theta(\mathbf{w}|\theta) \approx P(\mathbf{w}|\mathcal{D})$. This is generally a more tractable distribution, such as a Gaussian distribution, that we could sample from. This approximation to the true posterior is called the variational posterior. This posterior is composed of a set of parameters θ. These parameters are to be optimized using the Kullback-Leibler (KL) divergence [16]. KL divergence is used to measure how well a probability distribution Q approximates the probability distribution P.

We seek to minimize the KL divergence (i.e. maximize the similarity) between the variational posterior and the true posterior. Inference is performed by optimizing the approximate parameters θ:

$$\begin{aligned} \theta_{opt} &= \arg\min_{\theta} \mathbf{KL}\big[q_\theta(\mathbf{w}|\theta) \;||\; P(\mathbf{w}|\mathcal{D})\big] \\ &= \arg\min_{\theta} \mathbf{KL}\big[q_\theta(\mathbf{w}|\theta) \;||\; P(\mathbf{w})\big] \\ &\quad - \mathop{\mathbb{E}}_{q_\theta(\mathbf{w}|\theta)}\big[\log P(\mathcal{D}|\mathbf{w})\big]. \end{aligned} \tag{3}$$

This can be modelled as a cost function, consisting of a likelihood cost (data-dependent) and a complexity cost (prior-dependent). It therefore trades off between satisfying the data $\mathcal{D}$ and the prior $P(\mathbf{w})$. This cost function is known as the variational free energy [3, 34, 20] or evidence lower bound [11, 20, 24].

Exactly minimising the cost function in Equation 3 is computationally prohibitive. Therefore, gradient descent and various approximations are used. In this work, we use a stochastic variational method [1, 6], a generalization of the Local Reparameterization Trick [15]. Then, by using Monte Carlo methods to sample $\mathbf{w}$ instances from the approximate q and then applying a backpropagation algorithm, Blundell *et al*. achieve variational inference in neural networks.

This last step is the crux of Bayes by Backprop: first, we approximate the true distribution P with an approximate

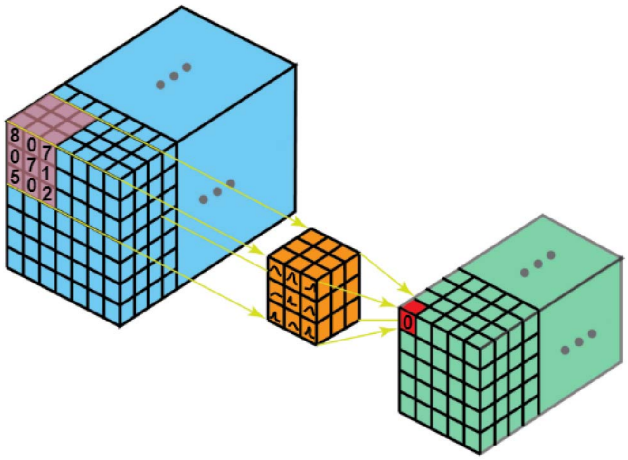


Figure 2: Bayesian 3D Convolution. In a Bayes by Backprop 3D ConvNet the weights of the convolution (orange block) are formed by probability distributions. The weights are sampled from these distributions, and then applied to the input (blue block) to produce an output (green block).

distribution q formed by the parameters θ that are learnt; and second we sample $\mathbf{w}$ from that q while seeing data. After sampling $\mathbf{w}$ instances the cost function is approximated as followed:

$$\begin{aligned}\mathcal{F}(\mathcal{D},\theta) \approx &\sum_{i=1}^{n} \log q_\theta(w^{(i)}|\mathcal{D}) - \log P(w^{(i)}) \\ &- \log P(\mathcal{D}|w^{(i)}),\end{aligned} \tag{4}$$

sampling $w^{(i)}$ from $q_\theta(\mathbf{w}|\mathcal{D})$ using Monte Carlo.

3.2. Bayesian 3D ConvNets

In convolutional layers, we do not deal with weights alone but also filters, which can be seen as collection of weights. Here we again replace the single values of the filters weights with probability distributions, and apply variational inference through Bayes by Backprop to approximate the intractable true posterior probability distribution. To approximate the true posterior in 3D Convolutional layers, we employ Gaussian distributions. However this distribution will differ to the one defined in Section 3.1. The hyperparameters of the variational posterior $q_\theta(\mathbf{w}|\theta)$ will now be $\theta = (\mu, \alpha)$. The Gaussian distribution is formed by a mean μ and a standard deviation $\sigma = \sqrt{\alpha \cdot \mu^2}$. This results in the following variational posterior:

$$\begin{aligned}\theta =&(\mu, \alpha), \\ \sigma^2 =&\alpha \cdot \mu^2, \\ q_\theta(\mathbf{w}_{ijhwt}|\mathcal{D}) =&\mathcal{N}(\mu_{ijhwt}, \alpha_{ijhwt}\mu^2_{ijhwt}),\end{aligned} \tag{5}$$

where i is input, j output, h filter height, w filter width, and t represents the time dimension. In regular Bayesian feed-forward network weights, we build a variational approximation q to the true posterior p, sample from it and apply the local reparameterization trick. For 3D ConvNets, this is done in a slightly truncated way. We do not sample the weights from q anymore, instead we sample layer activations b [21, 25] which helps to accelerate computation:

$$b_j = A_i * \mu_i + \epsilon_j \cdot \sqrt{A_i^2 * (\alpha_i \cdot \mu_i^2)}, \tag{6}$$

where ϵ are noise samples from the standard normal distribution, A_i is the input, $*$ symbolices the 3D convolutional operation, and $\cdot$ the element-wise multiplication. A visualization of a Bayesian 3D convolution is sohnw in Figure 2.

This means that we perform two convolutional operations to obtain the output. First, we use the output b as a frequentist ConvNets output optimising it using a backpropagation algorithm. Second, we learn the variance $\sigma^2 = \alpha * \mu^2$ in the second convolutional operation. The intuition is that, on the first convolution we learn the Maximum A Posteriori (MAP) of the approximate distribution q. In the second convolution we observe how much do the values of the weights deviate from the MAP.

4. Experimental Setup

4.1. Datasets

All three datasets used in this experiments are small-sized datasets, to investigate how the model react to overfitting. All datasets represent either human actions or sports.

The **KTH** [17] dataset consists of 6 classes, with a total of 600 videos. Most of them are filmed in homogeneous and static backgrounds, in four different scenarios. The Actions are: *walking, jogging, running, boxing, hand waving*, and *hand clapping*.

The **UCF11** [18] dataset consists of 11 sports classes with more than 100 videos per class. Action categories include: *basketball shooting, biking/cycling, diving, golf swinging, horse back riding, soccer juggling, swinging, tennis swinging,trampoline jumping, volleyball spiking*, and *walking with a dog*.

The **InfAR** [5] dataset consists of action recognition videos filmed with infra-red cameras. It is formed by 600 videos of 12 classes (50 videos per class) of a few seconds each. All videos are filmed in 3 different static backgrounds. The actions are *one hand wave (wave1), multiple hands wave (wave2), handclap, walk, jog, jump, skip, handshake, hug, push, punch,* and *fight*.

4.2. Implementation Details

We use the 3D version of the ResNet-18 network architecture [8] due to its facility to learn and the good results

	KTH		UCF11		InfAR	
	Valid	**Train**	**Valid**	**Train**	**Valid**	**Train**
Frequentist	75.0	80.8	60.4	82.2	34.2	100.0
Bayesian	**78.1**	77.9	**61.3**	82.0	**45.0**	100.0

Table 2: Comparison between standard (frequentist) 3D ConvNets and Bayesian 3D ConvNets on three small-scale action datasets. Across all datasets, incorporating Bayesian inference is preferred. We attribute this to a stronger regularization, exemplified by the lower training accuracies.

	KTH	UCF11	InfAR
Total	73.8	57.6	**45.0**
Average	**78.1**	**61.3**	**45.0**
Voting	77.8	60.8	44.2

Table 1: Comparison of different aggregation methods for the multiple samples obtained per test videos in Bayesian 3D ConvNets. Averaging is preferred and we will use this approach throughout further experiments.

delivered using 3D models in video datasets [7]. We use the Adam optimizer [14] as our backpropagation algorithm. Learning rate is chosen by the method introduced by Smith [27]. This method consists of training a network starting from a low learning rate and increase the learning rate exponentially for every batch, until divergence. Then, analyzing how the loss advanced through these first batches, we choose the learning rate where it decreased the fastest. We found 10^{-6} to be a good starting learning rate.

5. Experimental Results

5.1. Inference in Bayesian 3D ConvNets

In Bayesian networks, we introduce uncertainty in the form of variance to every weight in the network. This means that, every time that you introduce a data point to the network, the value in the weights will be different and therefore the final output will change. As a result, we have a non-deterministic network, that can assign different classes to the same input.

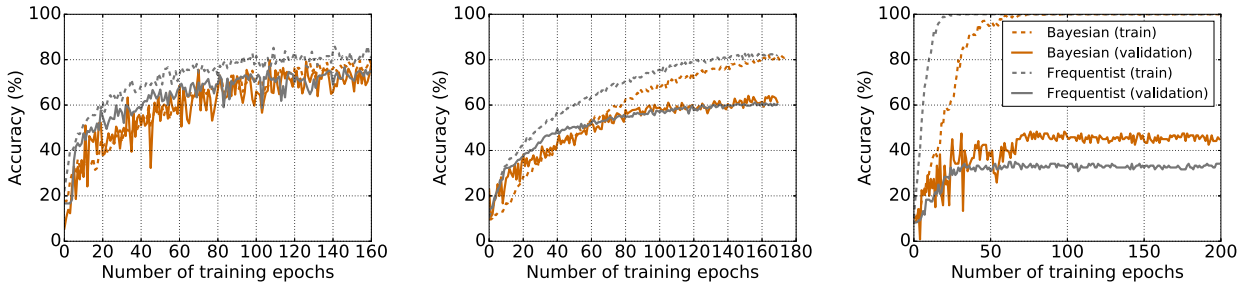
To asses the accuracy of our Bayesian network we run several samples of the input and obtain several different outputs. As we are dealing with a classification problem, this means we obtain different probability distributions over actions. In the first experiment, we investigate three ways to obtain a final prediction from multiple samples. The first, total accuracy, computes the total accuracy over all samples. The second, averaged accuracy, first averages the softmax scores over all samples and then selects the highest score action as the prediction. This can be seen as an early fusion of the samples. For the third, voting accuracy, each sample result returns a class. We consider this class as a vote and declare the more voted class as the final result. This can be seen as a late fusion of the samples.

The results for the different sample aggregation methods is shown in Table 1. Across all three datasets, the averaging method obtains the highest scores. The total method performs the same on InfAR, while performing 4.3 percent point (p.p.) and 3.7 p.p. lower on KTH and UCF11 respectively. The voting method performs between 0.8 p.p. and 0.3 p.p. lower across the three datasets. Based on this analysis, we recommend the use of averaging in Bayesian 3D ConvNets and we will use this approach throughout further experiments.

5.2. Comparison to standard 3D ConvNets

The second experiment performs a direct comparison between standard 3D ConvNets and Bayesian 3D ConvNets. For this experiment, we use the same architecture and settings for both and report on the three datasets. The results are shown in Table 2. The table shows that on all three datasets, Bayesian 3D ConvNets perform better. On InfAR, Bayesian 3D ConvNets obtain an accuracy of 45.0%, compared to 34.2% for the baseline, an improvement of 11.2 p.p. On KTH, Bayesian 3D ConvNets outperform the baseline by 3.1 p.p. (from 75.0% to 78.1%), while on UCF11, the improvement is 0.9 p.p. (from 60.4% to 61.3%). We also provide the per epoch accuracies in Figure 3, which show that Bayesian 3D ConvNets continue improving over training epochs, while the baseline validation performance flattens quicker.

To gain insight as to why the Bayesian 3D ConvNet are interesting for action recognition when training examples are scarce, we also show the training accuracies in Table 2 and Figure 3. The Bayesian 3D ConvNet obtains consistently lower training accuracies, which is attributed to the stronger regularization that is imposed. We conclude from this experiment that Bayesian 3D ConvNets are effective for



(a) KTH. (b) UCF11. (c) InfAR.

Figure 3: Per epoch training and validation comparison between standard and Bayesian 3D ConvNets. On all datasets, we find that the Bayesian variant perform a stronger regularization than the standard 3D ConvNet, which results in validation improvements.

	KTH		UCF11		InfAR	
	Valid	**Train**	**Valid**	**Train**	**Valid**	**Train**
Dropout	**78.5**	83.8	60.4	88.8	41.7	100.0
Bayesian	78.1	77.9	**61.3**	82.0	**45.0**	100.0

Table 3: Comparison between DropOut regularization in 3D ConvNets and Bayesian 3D ConvNets. Bayesian 3D ConvNets provide competitive or better results to DropOut regularization.

action recognition on smaller datasets. This holds for both actions captured with RGB and infar-red sensors.

5.3. Comparison to DropOut

For the third experiment, we compare Bayesian 3D ConvNets to a DropOut variant of 3D ConvNets. To do so we follow Zagoruyko *et al*. [36], that efficiently implemented DropOut into ResNets. DropOut [29] is a widely known regularization method to prevent overfitting, and is also used as baseline in other Bayesian networks [4]. The comparison on both datasets is shown in Table 3. Akin to the first experiment, we observe improvements on the InfAR dataset. DropOut improves over standard 3D ConvNets (41.7% accuracy), but is 3.2 p.p. lower than Bayesian 3D ConvNets. On KTH, DropOut obtains competitive validation scores compared to Bayesian 3D ConvNets (78.5% versus 78.1%). On UCF11, Bayesian 3D ConvNets (61.3%) outperform the DropOut baseline (60.4%), which does not improve over standard 3D ConvNets. We conclude that Bayesian 3D ConvNets provide competitive regularizers for action recognition on small datasets.

5.4. Few example evaluation

For the fourth experiment, we dive deeper into the potential of Bayesian deep learning for videos in the few example setting. We again take the KTH and InfAR datasets and reduce the training set to a quarter of the original size through random selection of training examples. We maintain the same size for the test set. The comparison to both standard 3D ConvNets and the DropOut baseline are shown in Table 4. On both datasets, the Bayesian 3D ConvNet outperforms the two baselines. The improvement is 4.3 p.p. on KTH and 11.6 p.p. on InfAR. In this setting, DropOut failed to converge on KTH, indicating that this regularizer is not as stable as Bayesian 3D ConvNets. We conclude here that especially when examples are scarce, Bayesian deep learning is effective on 3D ConvNets for action recognition.

6. Conclusions

We have extended Bayes By Backprop to 3D convolutional neural networks and investigated how it behaves on various action datasets with limited examples. We have found Bayesian 3D ConvNets to outperform both standard 3D ConvNets and its variant with Dropout as regularizer. The main factor in this improvement was because overfitting was combatted through strong regularization. This improvement holds especially when when training examples become more scarce. For future work, an important step is to reduce the computational complexity in Bayesian deep learning, which is a strain when dealing with large networks, as is the case in the spatio-temporal video domain.

	KTH		InfAR	
	Valid	**Train**	**Valid**	**Train**
Frequentist	54.2	75.0	20.8	83.3
Dropout	d.n.c	d.n.c	23.3	62.5
Bayesian	**64.8**	68.3	**30.8**	99.2

Table 4: Bayesian 3D ConvNets compared to standard and DropOut 3D ConvNets on KTH and InfAR using a quarter of the training data. d.n.c. denotes did not converge. On both datasets, the Bayesian 3D ConvNet provides clear improvements over the baselines, highlighting its effectiveness for few-example action recognition.

References

[1] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. 2015.

[2] Meire Fortunato, Charles Blundell, and Oriol Vinyals. Bayesian recurrent neural networks. *CoRR*, 2017.

[3] Karl Friston, Jérémie Mattout, Nelson Trujillo-Barreto, John Ashburner, and Will Penny. Variational free energy and the laplace approximation. *Neuroimage*, 34(1):220–234, 2007.

[4] Yarin Gal and Zoubin Ghahramani. Bayesian convolutional neural networks with bernoulli approximate variational inference. *ICLR workshops*, 2015.

[5] Chenqiang Gao, Yinhe Du, Jiang Liu, Jing Lv, Luyu Yang, Deyu Meng, and Alexander G Hauptmann. Infar dataset: Infrared action recognition at different times. *Neurocomputing*, 212:36–47, 2016.

[6] Alex Graves. Practical variational inference for neural networks. In *NeurIPS*, 2011.

[7] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *CVPR*, 2018.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[9] Geoffrey Hinton and Drew Van Camp. Keeping neural networks simple by minimizing the description length of the weights. In *CoLT*, 1993.

[10] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017.

[11] Tommi S Jaakkola and Michael I Jordan. Bayesian parameter estimation via variational methods. *Statistics and Computing*, 10(1):25–37, 2000.

[12] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *TPAMI*, 35(1):221–231, 2012.

[13] Zhuolin Jiang, Viktor Rozgic, and Sancar Adali. Learning spatiotemporal features for infrared action recognition with 3d convolutional neural networks. In *CVPR workshops*, 2017.

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

[15] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *ICLR*, 2014.

[16] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[17] Ivan Laptev, Barbara Caputo, et al. Recognizing human actions: a local svm approach. In *ICPR*, 2004.

[18] Jingen Liu, Jiebo Luo, and Mubarak Shah. Recognizing realistic actions from videos in the wild. In *CVPR*, 2009.

[19] Ashish Mishra, Vinay Kumar Verma, M Shiva Krishna Reddy, S Arulkumar, Piyush Rai, and Anurag Mittal. A generative approach to zero-shot and few-shot action recognition. In *WACV*, 2018.

[20] Radford M Neal and Geoffrey E Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. Springer, 1998.

[21] Kirill Neklyudov, Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. Variance networks: When expectation does not meet your expectations. *CoRR*, 2018.

[22] Carlos Orrite, Mario Rodríguez, and Miguel Montañés. One-sequence learning of human actions. In *International Workshop on Human Behavior Understanding*, 2011.

[23] Mario Rodriguez, Carlos Orrite, Carlos Medrano, and Dimitrios Makris. Fast simplex-hmm for one-shot learning activity recognition. In *CVPR workshops*, 2017.

[24] Lawrence K Saul, Tommi Jaakkola, and Michael I Jordan. Mean field theory for sigmoid belief networks. *JAIR*, 4:61–76, 1996.

[25] Kumar Shridhar, Felix Laumann, and Marcus Liwicki. Uncertainty estimations by softplus normalization in bayesian convolutional neural networks with variational inference. *CoRR*, 2018.

[26] Kumar Shridhar, Felix Laumann, and Marcus Liwicki. A comprehensive guide to bayesian convolutional neural network with variational inference. *CoRR*, 2019.

[27] Leslie N Smith. Cyclical learning rates for training neural networks. In *WACV*, 2017.

[28] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, 2012.

[29] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014.

[30] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015.

[31] Heng Wang, Alexander Kläser, Cordelia Schmid, and Liu Cheng-Lin. Action recognition by dense trajectories. In *CVPR*, 2011.

[32] Baohan Xu, Hao Ye, Yingbin Zheng, Heng Wang, Tianyu Luwang, and Yu-Gang Jiang. Dense dilated network for few shot action recognition. In *ICMR*, 2018.

[33] Yang Yang, Imran Saleemi, and Mubarak Shah. Discovering motion primitives for unsupervised grouping and one-shot learning of human actions, gestures, and expressions. *TPAMI*, 35(7):1635–1648, 2012.

[34] Jonathan S Yedidia, William T Freeman, and Yair Weiss. Generalized belief propagation. In *NeurIPS*, 2001.

[35] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *ICLR*, 2016.

[36] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *BMVC*, 2016.