Journal of Universal Computer Science, vol. 28, no. 2 (2022), 141-159
submitted: 23/6/2021, accepted: 15/11/2021, appeared: 28/2/2022 CC BY-ND 4.0

LogSE: An Uncertainty-Based Multi-Task Loss Function for Learning Two Regression Tasks

Zeinab Ghasemi-Naraghi
(Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran
z_naraghi@aut.ac.ir)

Ahmad Nickabadi (Corresponding author)
(Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran
nickabadi@aut.ac.ir)

Reza Safabakhsh
(Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran
safa@aut.ac.ir)

Abstract: Multi-task learning (MTL) is a popular method in machine learning which utilizes related information of multi tasks to learn a task more efficiently and accurately. Naively, one can benefit from MTL by using a weighted linear sum of the different tasks loss functions. Manual specification of appropriate weights is difficult and typically does not improve performance, so it is critical to find an automatic weighting strategy for MTL. Also, there are three types of uncertainties that are captured in deep learning. Epistemic uncertainty is related to the lack of data. Heteroscedastic aleatoric uncertainty depends on the input data and differs from one input to another. In this paper, we focus on the third type, homoscedastic aleatoric uncertainty, which is constant for different inputs and is task-dependent. There are some methods for learning uncertainty-based weights as the parameters of a model. But in this paper, we introduce a novel multi-task loss function to capture homoscedastic uncertainty in multi regression tasks models, without increasing the complexity of the network. As the experiments show, the proposed loss function aids in learning a multi regression tasks network fairly with higher accuracy in fewer training steps.

Keywords: Multi-Task Learning, Bayesian Deep Network, Homoscedastic Uncertainty, Loss Function
Categories: I.1, I.2
DOI: 10.3897/jucs.70549

1 Introduction

Multi-task learning (MTL) is a standard methodology in machine learning. It is used to improve the learning efficiency and prediction accuracy of each task by using related tasks [Baxter 2000, Thrun 1996]. Multi-task learning (MTL) is a standard methodology in machine learning. Actually, the MTL's goal is to improve the learning efficiency and prediction accuracy of each task by using the related tasks [Baxter 2000, Thrun 1996]. MTL is widely used in many applications, but how to learn the multi tasks together is still a challenging issue. The straightforward way is to minimize the weighted sum of the tasks' loss functions. The weight values of different loss functions are critical. In this research, we aim to automatically learn these weights by capturing task-based

uncertainties without increasing complexity or decreasing accuracy. In consideration of the Gaussian likelihood assumption of regression tasks, and undetermined uncertainties of the tasks, the base objective function is optimized and a novel loss function is derived mathematically to find a simple and effective solution for the MTL problem. The key contribution of this paper is the introduction of a new multi-task loss function, called LogSE (Log Squared Error), to capture homoscedastic uncertainty in multi regression task models. We show that LogSE is more accurate and efficient than the base loss functions. In addition, we show LogSE can learn multi tasks more fairly. In MTL, unfair weighting strategies cause the target model to learn a task better than the other tasks. It is shown that LogSE can learn multi tasks more fairly.

MTL is inspired by human abilities in the real world. Humans often learn a task by using the information of other tasks. For example, in the communication world, we understand each other by interpreting intonation cues and body language signs simultaneously. As another example, we may use the experience of riding a bicycle to learn how to ride a motorcycle. MTL provides many advantages over single-task learning. 1) Deep learning methods often need a large number of training data to achieve satisfying accuracy. Sometimes, providing such large data sets is difficult or impossible. In these cases, the performance of MTL is better than a single task learning method with smaller amounts of data [Kaiser et al. 2017, Zhang and Yang 2017, Caruana 1997]. 2) In MTL, what is learned for each task can help other tasks to be learned better by learning tasks simultaneously and using a shared representation [Caruana 1997]. 3) MTL improves generalization by sharing the information between related tasks, so it reduces the risk of overfitting and can be used as a regularization mechanism [Zhang and Yang 2017]. 4) A multi-task model with a shared feature extractor network is more efficient than a single task learning method when the training samples are sparse [Gong et al. 2019]. 5) The number of parameters in a multi-task model would be fewer than the parameters of multiple models for different tasks [Gong et al. 2019].

MTL is prevalent in widespread fields, especially deep neural networks [Cao et al. 2018, Sermanet et al. 2013, Liao et al. 2016] and multi-criteria decision-making tools [Deveci et al. 2021, Torkayesh and Deveci 2021]. A popular version of MTL models is a neural network with some shared layers followed by multiple branches for learning different tasks. The overall loss function would be the combination of each task's loss function. A straightforward loss function is the weighted linear sum of task-based losses. Some researchers use the constant weights obtained manually [Teichmann et al. 2018, Cao et al. 2018]. But, due to the importance of determining the weight values [Gong et al. 2019], other researchers have studied the aggregation methods. The cross-stitch network [Misra et al. 2016] learns the weights of a linear combination of multi tasks' loss functions by adding a unit to the network. As another example, Gradient Normalization (GradNorm) [Chen et al. 2018] is similar to batch normalization, but it normalizes across tasks instead of batch data. GradNorm optimizes the weights of the loss function by penalizing the network when backpropagated gradients of a task are too large or too small. The Multi-Task Attention Network (MTAN) [Liu et al. 2019] uses a simple effective weighting method and learns task-specific feature-level attention. A shared global feature pool is followed by a soft-attention module for each task. In the two uncertainty-based algorithms, [Kendall et al. 2018] and [Liebel and Körner 2018], the variances are learned as parameters of the neural networks.

While deep neural networks are the preferred advanced approaches used in many applications, they fail to model uncertainty. So, Bayesian deep networks were proposed to model the uncertainties. Epistemic, heteroscedastic aleatoric, and homoscedastic aleatoric are three types of uncertainties existing in deep networks. Usually, a lack of

data causes epistemic uncertainty. Epistemic Uncertainty can be decreased by using more training data or considering some prior distributions over the weights of the neural network. Heteroscedastic aleatoric uncertainty is related to the input data and it differs from one input to another. Furthermore, homoscedastic aleatoric uncertainty depends on tasks and can be captured as the output of a model. Homoscedastic uncertainty does not depend on the inputs and is constant for all of them, so it is called task-dependent uncertainty [Kendall and Gal 2017].

Specifically, in [Kendall et al. 2018], an uncertainty-based weighting method is proposed for learning depth regression, and semantic and instance segmentation simultaneously. They capture homoscedastic uncertainty by learning some additional parameters for each task in the multi-task network and use them as the weights of multi tasks loss functions. Maybe some deep learning tools can learn the weights of loss functions easily without changing the network architecture. But in other tools, it is not possible to implement this idea without adding parameters to the network. Practically, it is costly to change the architecture or tool, it is preferred to have a simpler way which only changes the loss function.

In summary, the main novelty of the new loss function, LogSE, is that the proposed multi-task loss function can capture task-based uncertainties. So, the fairness in learning different tasks is improved by LogSE compared to the base methods. Moreover, it is efficient because no computational cost is added and the target accuracy is obtained in fewer training epochs.

In the following sections, we first review the existing literature on MTL. Then, we describe our novel method and show how we derive the multi regression tasks loss function. Finally, we carry out some experiments and demonstrate how LogSE can facilitate learning multi regression tasks fairly with higher accuracy in fewer training steps.

2 Related Work

In this section, we review the related work in two parts: multi-task learning and uncertainties in deep learning.

2.1 Multi-Task Learning

In the light of the potency of multi-task learning (MTL) to improve the learning and generalization performance of a task, MTL is prevalent in numerous applications [Caruana 1997, Stein 1956]. It improves learning efficiency by combining multiple objectives from a shared representation. MTL improves generalization performance by leveraging domain-specific information of related tasks. Since each task can help in better training other tasks, the generated features are more representative. In addition, it reduces the risk of overfitting, and thus it can be used as a regularizer [Ruder 2017].

MTL has been widely used in many learning methods [Amit et al. 2007, Evgeniou and Pontil 2004, Jalali et al. 2010, Paredes et al. 2012, Xue et al. 2007, Yang and Hospedales 2014]. In some studies, MTL has been applied implicitly. For example, in [Torkayesh and Deveci 2021], a novel robust decision-making tool is proposed for locating battery swapping stations in populated cities. The authors apply a multi-normalization procedure whose aggregation operator is the weighted sum of some normalization functions. Also, to solve multi-criteria decision-making problems, a variety of integration functions and weighting methods are introduced by Deveci et al. [Deveci et al. 2021].

A popular type of MTL is a network with shared layers followed by multi branches to learn multi tasks simultaneously and the loss function is a combination of the multi tasks loss functions. It is important to know 'what' tasks should be combined and 'how' they could be combined. A simple way is to perform a weighted linear sum of the losses of different tasks (eq. (1)). A straightforward approach to combine the loss functions of multi tasks is to consider constant weights obtained manually. While, it is a preferred method in some papers [Teichmann et al. 2018, Cao et al. 2018], the main issue with this method is tuning the weights. Tuning these hyperparameters takes some days for each trial. Also, the accuracy of the multi-task model depends sensitively on weight values [Kendall et al. 2018].

$$\mathcal{L} = \alpha_1 \mathcal{L}_1 + \alpha_2 \mathcal{L}_2 \tag{1}$$

The network loss function, $\mathcal{L}$, is a weighted linear sum of the tasks loss functions. Coefficients α_1 and α_2 are the weights of the two tasks loss functions, $\mathcal{L}_1$ and $\mathcal{L}_2$.

In consideration of the boring process of the manual determination of weights [Gong et al. 2019], finding appropriate weighting strategies in MTL appears as a crucial goal. Some weighting methods will be described briefly in this section.

The Cross-stitch network proposes a new unit for learning optimal combination of shared and task-specific representations [Misra et al. 2016]. This unit can be added to any multi-task network and can be trained end-to-end. Cross-stitch is modeled as a linear combination of multi tasks (eq. (2)).

$$\begin{bmatrix} \tilde{x}_A^{ij} \\ \tilde{x}_B^{ij} \end{bmatrix} = \begin{bmatrix} \alpha_{AA} & \alpha_{AB} \\ \alpha_{BA} & \alpha_{BB} \end{bmatrix} \begin{bmatrix} x_A^{ij} \\ x_B^{ij} \end{bmatrix} \tag{2}$$

For learning two tasks A and B, a unit is added to a layer of the network. The outputs of the layer, x_A and x_B, are the inputs of the unit. A linear combination of inputs $\tilde{x}_A$ and $\tilde{x}_B$ are the outputs of the unit. α_{AB} and α_{BA} are called different-task values since they weight another task output. α_{BB} and α_{AA} are the same-task values, as they weight the same task output. In the training process, the unit can freely move between shared and task-specific representations to find the best point.

Gradient Normalization (GradNorm) is an adaptive multi-task loss balancing technique [Chen et al. 2018]. GradNorm is similar to batch normalization, but it normalizes across tasks instead of batch data. In addition, it uses rate balancing as an objective of normalization. GradNorm optimizes the weights of the loss function by penalizing the network when backpropagated gradients of a task are too large or too small. GradNorm considers a loss function to optimize the weights of the loss function.

The Multi-Task Attention Network (MTAN) [Liu et al. 2019] learns task-specific feature-level attention. A shared global feature pool is followed by a soft-attention module for each task. In this model, a simple effective weighting method, called Dynamic Weight Average (DWA), is proposed as the model objective to find the correct balance between tasks. DWA learns weights based on the rate of the change of loss for each task and the relative rate of tasks to each other during training epochs. The greater the relative descending rate is, the bigger weights in the multi-task loss function will be. DWA is inspired by GradNorm, but whilst GradNorm needs internal gradients of the network, DWA only requires the numerical task losses and therefore its implementation is simpler.

An uncertainty-based weighting method is proposed in [Kendall et al. 2018] for learning depth regression, and semantic and instance segmentation jointly. The method considers the uncertainties of the different tasks as parameters of the network and learns them during training the network. These uncertainty values are then used to combine

Table 1: Overview of the studies on multi-task learning.

	Name and Reference	Application	Number of Tasks/Criterias	Aggregating Tasks' Loss Functions Method
1	MABAC [Deveci et al. 2021]	offshore wind farm site selection	18 criteria	type-2 neutrosophic fuzzy scores
2	TRUST [Torkayesh and Deveci 2021]	locating a BSS for electric scooters	10 criteria	Shannon's Entropy
3	OpenPose [Cao et al. 2018]	multi-person 2D pose estimation	2 tasks	constant weights
4	Multi-net [Teichmann et al. 2018]	autonomous driving	3 tasks	constant weights
5	Cross-stitch [Misra et al. 2016]	object detection and attribute prediction	2 tasks	optimal linear combination learning by NN
6	GradNorm [Chen et al. 2018]	depth estimation, surface normals, and semantic segmentation	2 or more tasks	adaptive multi-task loss balancing
7	MTAN [Liu et al. 2019]	semantic segmentation and depth estimation, and surface normals	2 or 3 tasks	task-specific attention modules
8	Multi-task learning using uncertainty [Kendall et al. 2018]	depth regression, and semantic and instance segmentation	3 tasks	learning uncertainty-based weights by NN
9	Auxiliary Tasks in Multi-task Learning [Liebel and Körner 2018]	depth estimation and semantic segmentation	2 tasks	learning weights by CNN

multi-task loss functions.

A new multi-task CNN architecture is proposed in [Liebel and Körner 2018]. The loss function of this model is inspired by [Kendall et al. 2018]. In combination of multi tasks loss functions, the coefficients of each loss function can be added to the network parameters. The overall loss function of N tasks is proposed as eq. (3).

$$\mathcal{L} = \sum_{i=1}^{N} \left(\frac{1}{2\alpha_i^2} \mathcal{L}_i + \ln\left(1 + \alpha_i^2\right) \right) \tag{3}$$

where α_i is the variance of task i, and $\ln\left(1 + \alpha_i^2\right)$ is an added regularization term. The MTL researches are summarized in Table 1.

2.2 Uncertainties in Deep Learning

The two basic abilities of humans are 1) perception and 2) thinking. Humans can perceive the environment and gather information by their senses like seeing and hearing. On the other hand, it is the ability of thinking and relational reasoning that makes humans special. We are interested to have in our machine learning models abilities similar to the humans' perception and thinking abilities.

Deep neural networks (DNNs) and probabilistic graphical models (PGMs) are two widely preferred machine learning paradigms that are complementary to each other.

While deep learning has the significant ability in perception and what humans do by seeing and hearing, it fails in thinking well. On the other hand, PGMs are not as good as deep networks in perceiving large-scale data, but they are excellent in dealing with uncertainty and probability. As a result, it is advantageous to unify DNNs with PGMs, resulting in Bayesian deep learning [Wang and Yeung 2020].

Unlike a human who can learn from the knowledge, deep networks are data-driven and need a lot of data to learn a task fairly. In practice, the lack of training data and diversity of it has bad consequences and the need for modeling uncertainties is increased substantially. In summary, three types of uncertainties are captured by Bayesian deep learning [Gal 2016, Kendall and Gal 2017]:

(1) Epistemic uncertainty is relevant to the lack of training data. It increases considerably for data different from the training set. Epistemic uncertainty determines how much the weights of the network vary given some data. Increasing training data and defining a prior distribution over the weights of the neural network are the ways of capturing the epistemic uncertainty. There are some effective and simple algorithms to estimate epistemic uncertainty which are widely employed [Gustafsson et al. 2020]. Epistemic uncertainty can be predicted by variational inference [Graves 2011] or sampling [Gal and Ghahramani 2016]. For example, deterministic uncertainty quantification (DUQ) [Van Amersfoort et al. 2020] is a method upon ideas of RBF networks for training a deterministic deep model. They declare their method captures both aleatoric and epistemic uncertainty in practice. Also, to decrease classification error for a dataset with unbalanced classes, a loss function is proposed with intra-class uncertainty [Zhu and Shan 2021].

(2) Heteroscedastic aleatoric uncertainty is predicted by considering a distribution over the model outputs. Whilst aleatoric uncertainty does not decrease with more data and it does not increase for out-of-date samples, it depends on the input data. It differs from one input to another input. For example, the depth regression of a textureless image is more uncertain than that of an image with a lot of lines and textures. Modeling heteroscedastic uncertainties can be simple with less complexity. In practice, aleatoric uncertainty is captured by using Monte Carlo dropout [Feng et al. 2018, Miller et al. 2018], learning the parameters in the probabilistic distribution of bounding boxes [Meyer et al. 2019, Feng et al. 2019], or pre-estimate the uncertainty of the label and learn the probabilistic distribution by minimizing the KLD of predicted from preset ones [Meyer and Thakurdesai 2019, He et al. 2019]. For example, to improve 3D object detection in [Pan et al. 2020], they estimate the aleatoric uncertainties of the corners by learning the parameters of the probabilistic distribution.

(3) Homoscedastic aleatoric uncertainty is captured as the output of a model. It does not depend on the inputs and is constant for all input data. It is caused because of the common noises inherent in the observations or sensors. Actually, it is related to tasks and hence is called task-dependent uncertainty. For example, for estimation of human pose in a single 2D image, lack of deep estimation causes uncertainty which is held in common in all outputs. Homoscedastic uncertainty can be decreased by utilizing other information e.g. estimated depth of an image. An uncertainty-driven multi-loss function is proposed in [BenTaieb and Ghassan 2017] to improve the classification and segmentation accuracy. They learn the uncertainty as the parameters of the neural network. In addition, two studies which are reviewed earlier capture homoscedastic uncertainty as the parameters of their model [Liebel and Körner 2018, Kendall et al. 2018]. In this paper, we attempt to model only homoscedastic uncertainty.

3 Uncertainty-Based Multi-Task Loss Function

In this section, an efficient method for learning multi regression tasks is proposed which incorporates capturing homoscedastic uncertainty inside the loss function without increasing complexity. In the following section, we first show that homoscedastic uncertainty can be captured as an MTL. After that, the new loss function is derived. In the end, we explain some properties of this new loss function.

3.1 Uncertainty-Based Weight Learning

In this work, we focus on learning two regression tasks. The base neural network used in our experiments has some shared layers followed by two-stream networks. The objective function can be the sum of weighted loss functions that are related to the two tasks. Due to the importance of appropriately weighting the losses, some methods have been developed to learn the weights of each task loss function in the network objective function more efficiently [Liu et al. 2019, Liebel and Körner 2018, Chen et al. 2018]. In [Kendall et al. 2018], an uncertainty-based weighting method is proposed. The weights are learned as a part of the convolutional neural network. If the probabilistic likelihood of a regression task is a Gaussian distribution, the variance parameter represents the noise and uncertainty of the task, which is called homoscedastic uncertainty. We describe the method as a solution to problem 1 defined below.

Problem 1(Uncertainty learning): Given the data$\{(x^{(i)}, {y_1}^{(i)}, {y_2}^{(i)}) : i = 1, \ldots, N\}$, find the best weights w for the multi-task network f as

$$\arg\max_{w} \quad \mathcal{J}(w) \tag{4}$$

where

$$\begin{aligned} \mathcal{J}(w) &= \prod_{i=1}^{N} \quad p(y_1^{(i)}, y_2^{(i)} | x^{(i)}, w) \\ y_1 &= \mathcal{N}(f^{w_1}(x), \sigma_1^2); \quad w_1 \subset w \\ y_2 &= \mathcal{N}(f^{w_2}(x), \sigma_2^2); \quad w_2 \subset w \end{aligned} \tag{5}$$

Where w_1 and w_2 are related to each task and have some shared weights. The two tasks are independent, so $\mathcal{J}(w)$ can be written as eq. (6).

$$\begin{aligned} \mathcal{J}(w) &= \prod_{i=1}^{N} \quad p(y_1^{(i)} | x^{(i)}, w_1) p(y_2^{(i)} | x^{(i)}, w_2) \\ &= \prod_{i=1}^{N} \quad [\frac{1}{\sqrt{2\pi}\sigma_1} \exp\left(- \frac{\| y_1^{(i)} - f^{w_1}(x^{(i)}) \|^2}{2\sigma_1^2} \right) \\ &\qquad \frac{1}{\sqrt{2\pi}\sigma_2} \exp\left(- \frac{\| y_2^{(i)} - f^{w_2}(x^{(i)}) \|^2}{2\sigma_2^2} \right)] \end{aligned} \tag{6}$$

We solve the problem by minimizing the loss function, $\mathcal{L}$, instead of maximizing $\mathcal{J}$

(eq. (7)).

$$
\begin{aligned}
&\max_{w} \quad \mathcal{J}(w) \equiv \min_{w} \quad \mathcal{L}(w) \\
&\mathcal{L}(w) = -\log\left(\mathcal{J}(w)\right) \\
&\qquad = \sum_{i=1}^{N} \quad [\log\left(\sqrt{2\pi}\sigma_1\right) + \frac{\|y_1^{(i)} - f^{w_1}(x^{(i)})\|^2}{2\sigma_1^2} + \\
&\qquad\qquad\qquad \log\left(\sqrt{2\pi}\sigma_2\right) + \frac{\|y_2^{(i)} - f^{w_2}(x^{(i)})\|^2}{2\sigma_2^2}] \\
&\mathcal{L}(w) = N\log\left(\sigma_1\right) + N\log\left(\sigma_2\right) \\
&\qquad + \frac{1}{2\sigma_1^2}\sum_{i=1}^{N}\|y_1^{(i)} - f^{w_1}(x^{(i)})\|^2 + \frac{1}{2\sigma_2^2}\sum_{i=1}^{N}\|y_2^{(i)} - f^{w_2}(x^{(i)})\|^2
\end{aligned}
\tag{7}
$$

$$
\begin{aligned}
&\mathcal{L}(w) = \log\left(\sigma_1\right) + \log\left(\sigma_2\right) + \frac{1}{2\sigma_1^2}\mathcal{L}_1(w_1) + \frac{1}{2\sigma_2^2}\mathcal{L}_2(w_2) \\
&\mathcal{L}_1(w_1) = \frac{1}{N}\sum_{i=1}^{N}\|y_1^{(i)} - f^{w_1}(x^{(i)})\|^2 \\
&\mathcal{L}_2(w_2) = \frac{1}{N}\sum_{i=1}^{N}\|y_2^{(i)} - f^{w_2}(x^{(i)})\|^2
\end{aligned}
\tag{8}
$$

Equation (8) shows the final solution of the proposed loss function. σ_1 and σ_2 should be estimated as the parameters of the network. In practice, the log variance, $log(\sigma^2)$, is learned instead of the variance [Kendall et al. 2018].

As shown in this section, homoscedastic uncertainty can be captured by multi-task learning. While some deep learning tools can simply estimate the weights of loss functions in the training process, most deep learning tools need to change the network architectures and increase the parameters.

3.2 Uncertainty-Based Loss Function

In this section, we attempt to find a simple way to capture homoscedastic uncertainty without increasing network parameters. So, we propose the problem 2 and try to solve it by expanding the solution of problem 1 which was explained earlier. The result is a multi-task loss function.

Problem 2 (Uncertainty-based loss function): Given a data set $\{(x^{(i)}, {y_1}^{(i)}, {y_2}^{(i)}) : i = 1, 2, \ldots, N\}$, find the best weights w for the multi-task network f, without changing the network parameters (eq. (9)):

$$
\arg\min_{w} \quad \mathcal{L}(w) \tag{9}
$$

where

$$\begin{aligned}\mathcal{L}(w) = &\quad \log(\sigma_1) + \log(\sigma_2) + \frac{1}{2\sigma_1^2}\mathcal{L}_1(w_1) + \frac{1}{2\sigma_2^2}\mathcal{L}_2(w_2)\\ \mathcal{L}_1(w_1) = &\frac{1}{N}\sum_{i=1}^{N}\|y_1^{(i)} - f^{w_1}(x^{(i)})\|^2\\ \mathcal{L}_2(w_2) = &\frac{1}{N}\sum_{i=1}^{N}\|y_2^{(i)} - f^{w_2}(x^{(i)})\|^2\end{aligned} \tag{10}$$

When the network's parameters are learned through the training process, the network's output for each task approaches the mean of the corresponding response variable given the input, i.e. $\overline{y}_1$ and $\overline{y}_2$ for our two tasks example. So, σ_1^2 and σ_2^2 can be estimated with sample variances $\overline{S}_1^2$ and $\overline{S}_2^2$, respectively;

$$\begin{aligned}\mathcal{L}_1(w_1) &= \frac{1}{N}\sum_{i=1}^{N}\|y_1^{(i)} - \overline{y}_1\|^2 = \overline{S}_1^2\\ \mathcal{L}_2(w_2) &= \frac{1}{N}\sum_{i=1}^{N}\|y_2^{(i)} - \overline{y}_2\|^2 = \overline{S}_2^2\end{aligned} \tag{11}$$

The variances σ_1^2 and σ_2^2 are replaced with the estimated values obtained in eq. (11). More details are added to eq. (12) to show how the novel loss function is derived from the base loss function.

$$\begin{aligned}\mathcal{L}(w) &= \frac{1}{2}\log\big(\sigma_1^2\big) + \frac{1}{2}\log\big(\sigma_2^2\big) + \frac{1}{2\sigma_1^2}\mathcal{L}_1(w_1) + \frac{1}{2\sigma_2^2}\mathcal{L}_2(w_2)\\ &= \frac{1}{2}\log(\mathcal{L}_1(w_1)) + \frac{1}{2}\log(\mathcal{L}_2(w_2)) + \frac{1}{2\mathcal{L}_1(w_1)}\mathcal{L}_1(w_1) + \frac{1}{2\mathcal{L}_2(w_2)}\mathcal{L}_2(w_2)\\ &= \frac{1}{2}\log(\mathcal{L}_1(w_1)) + \frac{1}{2}\log(\mathcal{L}_2(w_2)) + 1\end{aligned} \tag{12}$$

As a result, the simple form of eq. (13) is obtained for the loss function. we call it 'LogSE' as it uses the log of the squared error. This new loss function means that the average of logs of the tasks' loss functions should be considered in this case.

$$\mathcal{L}(w) = \frac{1}{2}(\log(\mathcal{L}_1(w_1)) + \log(\mathcal{L}_2(w_2))) \tag{13}$$

The diagram of the proposed model is shown in fig. 1. First, the input (X) is fed into the feature extractor block. These shared features (F) are the inputs to the two branches of the network. The outputs of the two branches of the network provided predicted values, and the uncertainty and loss values of each task are estimated and integrated into the LogSE loss function to estimate the total loss value of the network which is then used to train the network.

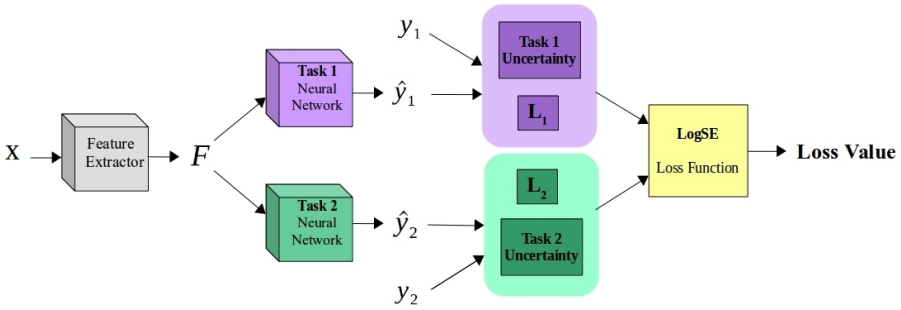


Figure 1: The framework of the proposed method.

3.3 LogSE Properties

We find that LogSE has three main properties. LogSE aids in learning a multi regression task network 1) more fairly, 2) with higher accuracy, and 3) in fewer training steps. LogSE loss function is defined as the sum of the log of squared errors of two tasks. Since it captures homoscedastic uncertainty implicitly, we expect it to help the network to learn the two tasks more fairly than the MSE loss function.

Unlike a human who can learn from the knowledge, deep networks are data-driven and need a lot of data to learn a task fairly. In practice, the lack of training data and innate diversity of it have improper consequences and the need for modeling uncertainties is increased substantially. In MTL, unfair weighting strategies cause learning a task better than the other tasks, and other tasks are dominated by one task. So, we capture homoscedastic uncertainty implicitly and expect it to help the network to learn the two tasks more fairly. By assuming Gaussian likelihood, variance parameters determine the uncertainty rate of a task. Actually, we propose a method to estimate the uncertainty implicitly. Therefore, it is expected that the LogSE loss function makes the network learn the desired task better and yields more accuracy. In consideration of uncertainty capturing in learning the multi tasks, the fairness is more observed.

Also, we will prove training a network with LogSE loss function saturates the network faster. The partial derivatives of MSE and LogSE loss functions with respect to w_1 and w_2 are shown in eqs. (14) and (15), respectively. w_1 is the weight vector of the first branch which is related to the first task and w_2 is related to the second task. As described in eq. (16), training the network with the LogSE loss function has smaller steps in the first epochs to find a better point. But, in the last epochs, training uses larger steps and makes the training faster.

$$\begin{aligned}
&\mathcal{L}_{MSE} = 1/2\|Y_1 - f^{w_1}(I)\|^2 + \|Y_1 - f^{w_2}(I)\|^2 \\
&\frac{\partial \mathcal{L}_{MSE}}{\partial w_1} = 2\|Y_1 - f^{w_1}(I)\|^2 \frac{\partial f^{w_1}(I)}{\partial w_1} \\
&\frac{\partial \mathcal{L}_{MSE}}{\partial w_2} = 2\|Y_2 - f^{w_2}(I)\|^2 \frac{\partial f^{w_2}(I)}{\partial w_2}
\end{aligned} \tag{14}$$

$$\mathcal{L}_{\text{LogSE}} = 1/2log(\|Y_1 - f^{w_1}(I)\|^2) + log(\|Y_1 - f^{w_2}(I)\|^2)$$
$$\frac{\partial \mathcal{L}_{\text{LogSE}}}{\partial w_1} = \frac{2\frac{\partial f^{w_1}(I)}{\partial w_1}}{\|Y_1 - f^{w_1}(I)\|^2} \tag{15}$$
$$\frac{\partial \mathcal{L}_{\text{LogSE}}}{\partial w_2} = \frac{2\frac{\partial f^{w_2}(I)}{\partial w_2}}{2\|Y_2 - f^{w_2}(I)\|^2}$$

$$\begin{aligned} &\text{if} \quad \|Y_1 - f^{w_1}(I)\| > 1 : \mathcal{L}_{\text{LogSE}} < \mathcal{L}_{\text{MSE}} \\ &\text{if} \quad \|Y_1 - f^{w_1}(I)\| < 1 : \mathcal{L}_{\text{LogSE}} > \mathcal{L}_{\text{MSE}} \end{aligned} \tag{16}$$

4 Experiments

This section reports comparative results between our loss function and three other base methods. The new loss function, 'LogSE', was driven in Section 3.2. As a base method, 'Uncertainty Weighting' [Kendall et al. 2018] was described in Section 3.1. 'MSE' and 'Best Variances' are other base methods. 'MSE' refers to a network that uses the popular loss function, Mean Square Error. Furthermore, the network loss function in the 'Best Variances' method is obtained by inserting the known and ground-truth variances in eq. (8).

We described our expectations of 'LogSE' in Section 3.3. 'LogSE' aids in learning a multi regression task network 1) more fairly, 2) with higher accuracy, and 3) in fewer training steps. In this section, we show them empirically.

A simple model is trained with the shared layers which are followed by two branches that the outputs of them are the estimations of y_1 and y_2. We train a model as an uncertainty weighting method that learns the uncertainty-based weights by adding two parameters to the network.

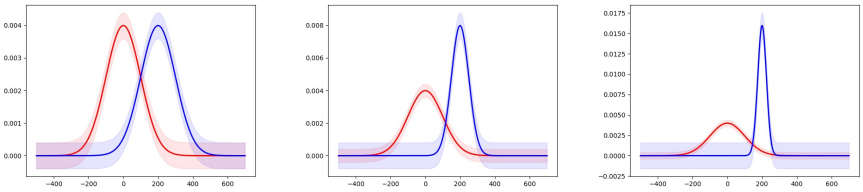
The methods are compared by two metrics: error and unfairness. Error is the mean square difference between the estimated and the actual values. Also, unfairness is the absolute difference between the two tasks errors.

Experiments are conducted for three data sets of size N, $D = \{(x^{(i)}, y_1^{(i)}, y_2^{(i)}) : i = 1, 2, \ldots, N\}$. $x^{(i)}$ uniformly distributed in $[-500, 700]^d$. y_1 and y_2 are generated from Gaussian distributions:

$$\begin{aligned} y_1^{(i)} &= \frac{1}{\sqrt{2\pi}\sigma_1} \exp\left(-\frac{(x^{(i)} - \mu_1)^2}{2\sigma_1^2}\right) \\ y_2^{(i)} &= \frac{1}{\sqrt{2\pi}\sigma_2} \exp\left(-\frac{(x^{(i)} - \mu_2)^2}{2\sigma_2^2}\right) \end{aligned} \tag{17}$$

where μ_1 and μ_2 are the mean parameters of y_1 and y_2 distributions which are selected as 0 and 200, respectively. Variances of y_1 and y_2 distributions are shown by $\sigma_1$and σ_2. The difference between the three data sets is the values of variances. As it is shown in fig. 2, the parameters of the three data sets are as follows: 1) $\sigma_1 = 100$ and $\sigma_2 = 100$, 2) $\sigma_1 = 100$ and $\sigma_2 = 50$ and 3) $\sigma_1 = 100$ and $\sigma_2 = 25$. Shuffled data is divided to three parts, 70 percent for training data, 10 percent for validation data and 20 percent for test data.

Two experiments are conducted on three data sets. In the first experiment, after each epoch, we evaluate the model trained on the test data. The errors are the mean



(a) *(b)* *(c)*

Figure 2: Three data sets with different distributions are shown. The σ values are respectively: (a) (100, 100), (b) (100, 50), and (c) (100, 25).

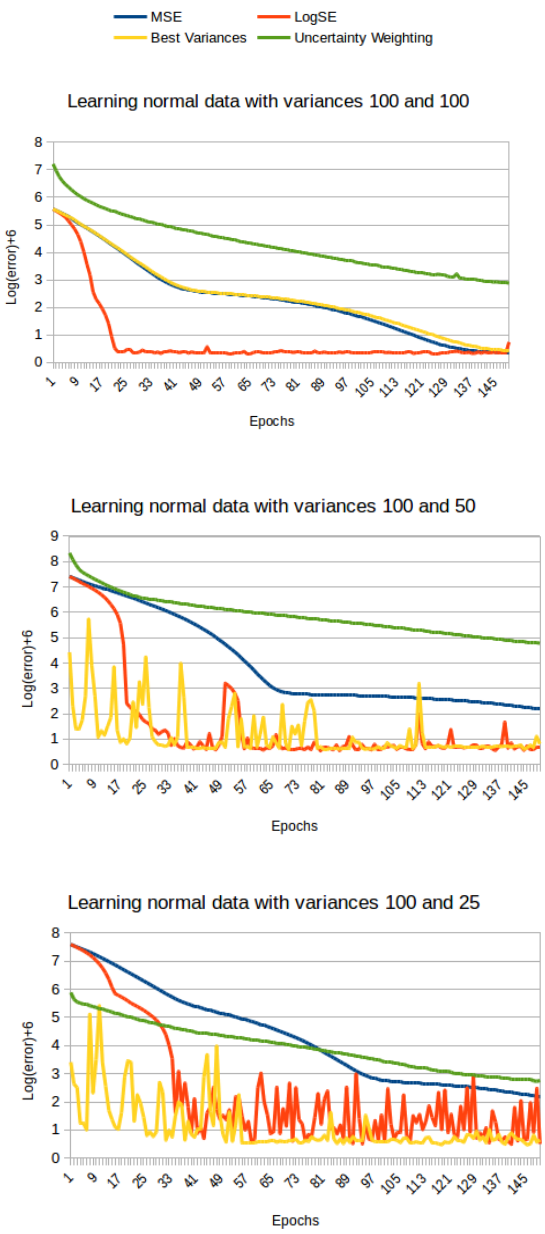
square difference of the actual and the estimated values. Results are shown in fig. 3. It is obvious that if the variances are equal, 'MSE' and 'Best Variances' have the same results. 'Uncertain Weighting' is not as good as other methods in the two-regression-task model. Also, the 'LogSE' method reaches the best accuracy and makes the network saturate sooner. The difference between the variances of the tasks causes some variations in the errors.

In the second experiment, we compute an unfairness metric for the learned model after each epoch. It is defined as the absolute difference between the errors of the two tasks. The results are shown in fig. 4. We conclude that training a network with 'LogSE' is almost as fair as 'Best Variances' and that the 'LogSE' is fairer than the 'MSE'. There are some variations in the plots. They are related to the difference between task-based variances.

Three attributes of the three loss functions (MSE, LogSE, and MSLE) are studied and compared in Table 2. The mean squared logarithmic error (MSLE) [Kiapour 2018, Brown 1968] is defined as the mean over the squared differences between the logarithm of real and estimated values. Actually, it measures the ratio between the true and predicted values. MSLE is used in regression tasks whose target, conditioned on the input, is normally distributed. As the first attribute and a limitation of the MSLE loss function, both of the actual and estimated values should be bigger than zero (bigger than '-1' when'+1' is added to both the true and predicted values) while MSE and LogSE are not limited to positive or negative values. As the second attribute, the loss functions have different behavior in considering the relative difference between the true and target values. For example, the big differences of big values would be treated as the small differences of small values by MSLE, while in MSE and LogSE, higher (smaller) relative differences result in higher (lower) loss values. Finally, MSE and LogSE penalize overestimate and underestimate similarly, but MSLE penalizes the overestimate more than the underestimate. Table 2 compares these three attributes with numerical examples.

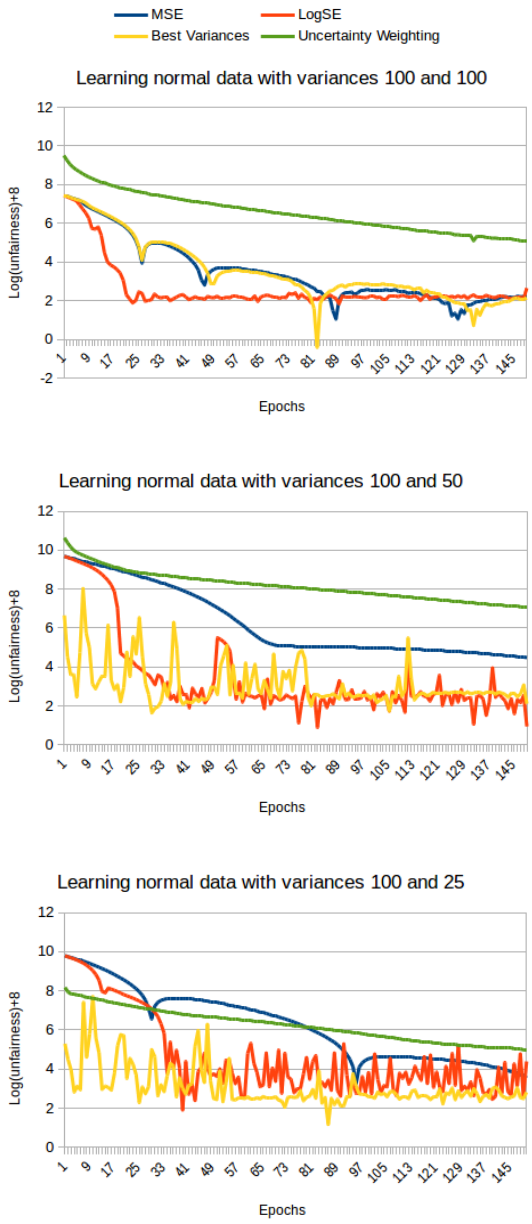
5 Conclusion

In this work, we have demonstrated that homoscedastic uncertainty can be captured in the loss function of the two-regression-task model. We showed that the network training by the new loss function achieves better accuracy in fewer training steps. Furthermore, we showed that the new loss function, LogSE, is fairer than the MSE loss function. In another word, LogSE makes fewer distance between the two tasks errors. Although the proposed loss function has provided insights into the multi-task learning, there are still



(c)

Figure 3: The plots show the error of four methods across epochs on three data sets: (a) (μ, σ) = (0, 100) and (μ, σ) = (200, 100), (b) (μ, σ) = (0, 100) and (μ, σ) = (200, 50) and (c) (μ, σ) = (0, 100) and (μ, σ) = (200, 25).



(c)

Figure 4: The plots show the unfairness of four methods across epochs on three data sets: (a) (μ, σ) = (0, 100) and (μ, σ) = (200, 100), (b) (μ, σ) = (0, 100) and (μ, σ) = (200, 50) and (c) (μ, σ) = (0, 100) and (μ, σ) = (200, 25).

Table 2: The comparison of three loss functions.

Attributes	Actual Value	Estimated Value	MSE	MSLE	LogSE
Negative Actual/Estimated Values	-20	10	900	N/A	2.95
	20	-10	900	N/A	2.95
	-20	-10	100	N/A	2
Penalizing Big Errors vs Small Errors	30	20	100	0.31	2
	30000	20000	100000000	0.31	8
Penalizing Underestimates vs Overestimates	20	10	100	0.09	2
	20	30	100	0.31	2

some limitations. The new loss function is introduced under the assumption of Gaussian distribution of the likelihood and it has been derived for multiple regression tasks. Other situations such as the different combinations of classification and regression tasks should be examined. As we have considered two tasks in our model, a different number of tasks can be studied, too. Although LogSE is derived based on a mathematical formulation and has been compared with other loss functions on synthesized data, it should be tested in real-world applications.

References

[Agrawal et al. 2015] Agrawal, P., Carreira, J., Malik, J.: ”Learning to see by moving”; Proceedings Of The IEEE International Conference On Computer Vision, (2015), 37-45.

[Amit et al. 2007] Amit, Y., Fink, M., Srebro, N., Ullman, S.: ”Uncovering shared structures in multiclass classification”; Proceedings Of The 24th International Conference On Machine Learning, (2007), 17-24.

[Argyriou et al. 2007] Argyriou, A., Evgeniou, T., Pontil, M.: ”Multi-task feature learning”; Advances In Neural Information Processing Systems, (2007), 41-48.

[Argyriou et al. 2008] Argyriou, A., Evgeniou, T., Pontil, M.: ”Convex multi-task feature learning”; Machine Learning, 73, 3 (2008), 243-272.

[Baxter 2000] Baxter, J.: ”A model of inductive bias learning”; Journal Of Artificial Intelligence Research, 12 (2000), 149-198.

[BenTaieb and Ghassan 2017] BenTaieb, A., Ghassan, H.: ”Uncertainty driven multi-loss fully convolutional networks for histopathology.”; Intravascular Imaging and Computer Assisted Stenting, and Large-Scale Annotation of Biomedical Data and Expert Label Synthesis. Springer, Cham (2017), 155-163.

[Brown 1968] Brown, L.D.: ”Inadmisibility of the usual estimator of scale parameters in problems with unknown location and scale parameters”; Annals of Mathematical Statistics, 39 (1968), 29-48.

[Calandriello et al. 2014] Calandriello, D., Lazaric, A., Restelli, M.: ”Sparse multi-task reinforcement learning”; Advances In Neural Information Processing Systems, (2014), 819-827.

[Cao et al. 2018] Cao, Z., Hidalgo, G., Simon, T., Wei, S., Sheikh, Y.: "OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields"; ArXiv Preprint ArXiv:1812.08008, (2018).

[Caruana 1997] Caruana, R.: "Multitask learning"; Machine Learning, 28 (1997), 41-75.

[Chen et al. 2018] Chen, Z., Badrinarayanan, V., Lee, C., Rabinovich, A. "Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks"; International Conference On Machine Learning, PMLR, (2018), 794-803.

[Collobert and Weston 2008] Collobert, R., Weston, J., "A unified architecture for natural language processing: Deep neural networks with multitask learning"; Proceedings Of The 25th International Conference On Machine Learning, (2008), 160-167.

[Collobert et al. 2011] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: "Natural language processing (almost) from scratch"; Journal Of Machine Learning Research, 12 (2011), 2493-2537.

[Das et al. 2017] Das, A., Hasegawa-Johnson, M., Veselỳ, K.: "Deep Auto-Encoder Based Multi-Task Learning Using Probabilistic Transcriptions"; INTERSPEECH, (2017), 2073-2077.

[Deveci et al. 2021] Deveci, M., Erdogan, N., Cali, U., Stekli, J., Zhong, Sh.: "Type-2 neutrosophic number based multi-attributive border approximation area comparison (MABAC) approach for offshore wind farm site selection in USA."; Engineering Applications of Artificial Intelligence 103 (2021): 104311.

[Evgeniou and Pontil 2004] Evgeniou, T., Pontil, M.: "Regularized multi–task learning"; Proceedings Of The Tenth ACM SIGKDD International Conference On Knowledge Discovery And Data Mining, (2004), 109-117.

[Feng et al. 2019] Feng, D. , Rosenbaum, L., Timm, F., Dietmayer, K.: "Leveraging heteroscedastic aleatoric uncertainties for robust real-time lidar 3d object detection"; in 2019 IEEE Intelligent Vehicles Symposium (IV). IEEE (2019), 1280–1287.

[Feng et al. 2018] Feng, D., Rosenbaum, L., Dietmayer, K.: "Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection"; in 2018 21st International Conference on Intelligent Transportation Systems (ITSC). IEEE (2018), 3266–3273.

[Gal 2016] Gal, Y.: "Uncertainty in deep learning"; University Of Cambridge, 1, 3 (2016).

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

[Gong et al. 2019] Gong, T., Lee, T., Stephenson, C., Renduchintala, V., Padhy, S., Ndirango, A., Keskin, G., Elibol, O.: "A comparison of loss weighting strategies for multi task learning in deep neural networks"; IEEE Access, 7 (2019), 141627-141632.

[Graves 2011] Graves, A.: "Practical variational inference for neural networks"; in Advances in neural information processing systems (2011), 2348–2356.

[Gustafsson et al. 2020] Gustafsson, F.K., Danelljan, M., Schon, T.B.: "Evaluating scalable bayesian deep learning methods for robust computer vision"; Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition Workshops, (2020), 318-319.

[He et al. 2019] He, Y., Zhu, C., Wang, J., Savvides, M., Zhang, X.: "Bounding box regression with uncertainty for accurate object detection"; in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2019), 2888–2897.

[Jalali et al. 2010] Jalali, A., Sanghavi, S., Ruan, C., Ravikumar, P.K.: "A dirty model for multi-task learning"; Advances In Neural Information Processing Systems, (2010), 964-972.

[Kaiser et al. 2017] Kaiser, L., Gomez, A.N., Shazeer, N., Vaswani, A., Parmar, N., Jones, L., Uszkoreit, J.: "One model to learn them all"; ArXiv Preprint ArXiv:1706.05137, (2017).

[Kang et al. 2011] Kang, Z., Grauman, K., Sha, F.: "Learning with Whom to Share in Multi-task Feature Learning"; ICML, 2 (2011), 4.

[Kendall and Gal 2017] Kendall, A., Gal, Y.: "What uncertainties do we need in bayesian deep learning for computer vision?"; Advances In Neural Information Processing Systems, (2017), 5574-5584.

[Kendall et al. 2018] Kendall, A., Gal, Y., Cipolla, R.: "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics"; Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition, (2018), 7482-7491.

[Kiapour 2018] Kiapour, A.: "Bayes, E-Bayes and robust Bayes premium estimation and prediction under the squared log error loss function."; Journal of the Iranian Statistical Society 17.1 (2018), 33-47.

[Kirkpatrick et al. 2017] Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al,: "Overcoming catastrophic forgetting in neural networks"; Proceedings Of The National Academy Of Sciences, 114, 13 (2017), 3521-3526.

[Liao et al. 2016] Liao, Y., Kodagoda, S., Wang, Y., Shi, L., Liu, Y.: "Understand scene categories by objects: A semantic regularized scene classifier using convolutional neural networks"; 2016 IEEE International Conference On Robotics And Automation (ICRA), (2016), 2318-2325.

[Liebel and Körner 2018] Liebel, L., Körner, M.: "Auxiliary tasks in multi-task learning."; ArXiv Preprint ArXiv:1805.06334, (2018).

[Liu et al. 2019] Liu, S., Johns, E., Davison, A.: "End-to-end multi-task learning with attention"; Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition, (2019), 1871-1880.

[Liu et al. 2015] Liu, X., Gao, J., He, X., Deng, L., Duh, K., Wang, Y.Y.: "Representation learning using multi-task deep neural networks for semantic classification and information retrieval", (2015).

[Long et al. 2013] Long, M., Wang, J., Ding, G., Sun, J., Yu, P.: "Transfer feature learning with joint distribution adaptation"; Proceedings Of The IEEE International Conference On Computer Vision, (2013), 2200-2207.

[Meyer et al. 2019] Meyer, G. P., Laddha, A., Kee, E., Vallespi-Gonzalez, C., Wellington, C. K.: "Lasernet: An efficient probabilistic 3d object detector for autonomous driving"; in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2019), 12677–12686.

[Meyer and Thakurdesai 2019] Meyer, G. P., Thakurdesai, N.: "Learning an uncertainty-aware object detector for autonomous driving"; arXiv preprint arXiv:1910.11375 (2019).

[Miller et al. 2018] Miller, D., Nicholson, L., Dayoub, F., Sünderhauf, N.: "Dropout sampling for robust object detection in open-set conditions"; in 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE (2018), 1–7.

[Misra et al. 2016] Misra, I., Shrivastava, A., Gupta, A., Hebert, M.: "Cross-stitch networks for multi-task learning"; Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition, (2016), 3994-4003.

[Obozinski et al. 2006] Obozinski, G., Taskar, B., Jordan, M.: "Multi-task feature selection"; Statistics Department, UC Berkeley, Tech. Rep, 2, 2 (2006).

[Obozinski 2010] Obozinski, G., Taskar, B., Jordan, M.: "Joint covariate selection and joint subspace selection for multiple classification problems"; Statistics And Computing, 20, 2 (2010), 231-252.

[Oquab et al. 2014] Oquab, M., Bottou, L., Laptev, I., Sivic, J.: "Learning and transferring mid-level image representations using convolutional neural networks"; Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition, (2014), 1717-1724.

[Pan and Yang 2009] Pan, S.J., Yang, Q.: "A survey on transfer learning"; IEEE Transactions On Knowledge And Data Engineering, 22, 10 (2009), 1345-1359.

[Pan et al. 2020] Pan, H., Wang, Z., Zhan, W., Tomizuka, M.: "Towards better performance and more explainable uncertainty for 3d object detection of autonomous vehicles"; 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2020.

[Paredes et al. 2012] Paredes, B.R., Argyriou, A., Berthouze, N., Pontil, M.: "Exploiting unrelated tasks in multi-task learning"; Artificial Intelligence And Statistics, PMLR, (2012), 951-959.

[Ruder 2017] Ruder, S.: "An overview of multi-task learning in deep neural networks"; ArXiv Preprint ArXiv:1706.05098, (2017).

[Sanh et al. 2019] Sanh, V., Wolf, T., Ruder, S.: "A hierarchical multi-task approach for learning embeddings from semantic tasks"; Proceedings Of The AAAI Conference On Artificial Intelligence, 33 (2019), 6949-6956.

[Sermanet et al. 2013] Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: "Overfeat: Integrated recognition, localization and detection using convolutional networks."; ArXiv Preprint ArXiv:1312.6229, (2013).

[Stein 1956] Stein, C.: "Inadmissibility of the usual estimator for the mean of a multivariate normal distribution"; Stanford University Stanford United States, (1956).

[Teh et al. 2017] Teh, Y., Bapst, V., Czarnecki, W.M., Quan, J., Kirkpatrick, J., Hadsell, R., Heess, N., Pascanu, R.: "Distral: Robust multitask reinforcement learning"; Advances In Neural Information Processing Systems, (2017), 4496-4506.

[Teichmann et al. 2018] Teichmann, M., Weber, M., Zoellner, M., Cipolla, R., Urtasun, R.: "Multinet: Real-time joint semantic reasoning for autonomous driving"; 2018 IEEE Intelligent Vehicles Symposium (IV), (2018), 1013-1020.

[Thrun 1996] Thrun, S.: "Is learning the n-th thing any easier than learning the first?"; Advances In Neural Information Processing Systems, (1996), 640-646.

[Thrun and Pratt 2012] Thrun, S., Pratt, L.: "Learning to learn"; Springer Science, Business Media, (2012).

[Torkayesh and Deveci 2021] Torkayesh, AE., Deveci, M.: "A mulTi-noRmalization mUlti-distance aSsessmenT (TRUST) approach for locating a battery swapping station for electric scooters."; Sustainable Cities and Society 74 (2021): 103243.

[Van Amersfoort et al. 2020] Van Amersfoort, J., Smith, L., Teh, Y. W., Gal, Y.: "Uncertainty estimation using a single deep deterministic neural network"; International Conference on Machine Learning. PMLR (2020).

[Wang and Yeung 2020] Wang, H., Yeung, D.Y.: "A Survey on Bayesian Deep Learning"; ACM Computing Surveys (CSUR), 53, 5 (2020), 1-37.

[Wang et al. 2015] Wang, X., Fouhey, D., Gupta, A.: "Designing deep networks for surface normal estimation"; Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition, (2015), 539-547.

[Xue et al. 2007] Xue, Y., Liao, X., Carin, L., Krishnapuram, B.: "Multi-task learning for classification with dirichlet process priors"; Journal Of Machine Learning Research, 8(Jan) (2007), 35-63.

[Yang and Hospedales 2014] Yang, Y., Hospedales, T.M.: "A unified perspective on multi-domain and multi-task learning"; ArXiv Preprint ArXiv:1412.7489, (2014).

[Yosinski et al. 2014] Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: "How transferable are features in deep neural networks?"; Advances In Neural Information Processing Systems, (2014), 3320-3328.

[Zhang and Yang 2017] Zhang, Y., Yang, Q; "A survey on multi-task learning"; ArXiv Preprint ArXiv:1707.08114, (2017).

[Zhu and Shan 2021] Zhu, H., Shan Y.: "Intra-Class Uncertainty Loss Function for Classification"; 2021 IEEE International Conference on Multimedia and Expo (ICME). IEEE (2021).