MULTI-TASK NEURAL PROCESSES

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural processes have recently emerged as a class of powerful neural latent variable models that combine the strengths of neural networks and stochastic processes. As they can encode contextual data in the network's function space, they offer a new way to model task relatedness in multi-task learning. To study its potential, we develop *multi-task neural processes*, a new variant of neural processes for multi-task learning. In particular, we propose to explore transferable knowledge from related tasks in the function space to provide inductive bias for improving each individual task. To do so, we derive the function priors in a hierarchical Bayesian inference framework, which enables each task to incorporate the shared knowledge provided by related tasks into its context of the prediction function. Our multi-task neural processes methodologically expand the scope of vanilla neural processes and provide a new way of exploring task relatedness in function spaces for multi-task learning. The proposed multi-task neural processes are capable of learning multiple tasks with limited labeled data and in the presence of domain shift. We perform extensive experimental evaluations on several benchmarks for the multi-task regression and classification tasks. The results demonstrate the effectiveness of multi-task neural processes in transferring useful knowledge among tasks for multi-task learning and superior performance in multi-task classification and brain image segmentation[1].

1 INTRODUCTION

As deep neural networks are black-box function approximations, it is difficult to introduce prior domain or expert knowledge into a prediction function (Jakkala, 2021). In contrast, Gaussian processes (Rasmussen, 2003) explicitly define distributions over functions and perform inference over these functions given some training examples. This enables reliable and flexible decision-making. However, Gaussian processes can suffer from high computational complexity due to the manipulation of kernel matrices. Therefore, there has been continuous interest in bringing together neural networks and Gaussian processes (Damianou & Lawrence, 2013; Wilson et al., 2016; Garnelo et al., 2018a; Jakkala, 2021) into so-called neural processes.

Neural processes (Garnelo et al., 2018b) combine the computational efficiency of neural networks with the uncertainty quantification of stochastic processes. They are a class of neural latent variables model, which deploy a deep neural network to encode context observations into a latent stochastic variable to model prediction functions. Neural processes provide an elegant formalism to efficiently and effectively incorporate multiple datasets into learning distributions over functions. This formalism is also promising in multi-task learning to improve individual tasks by transferring useful contextual knowledge among related tasks. Their capability of estimating uncertainty over predictions also makes them well-suited for multi-task learning with limited data, where each task has only a few training samples. However, neural processes rely on the implicit assumption that the context and target sets are from the same distribution and can be aggregated by a simple average pooling operation (Kim et al., 2019; Volpp et al., 2020). This makes it non-trivial to directly apply neural processes to modeling multiple heterogeneous tasks from different domains, where the context data of different tasks are from distinctive distributions (Long et al., 2017).

In this paper, we develop *multi-task neural processes*, a new variant of neural processes for multi-task learning. Particularly, we propose to explore task relatedness in the function space by specifying

[1] Our code is available in https://anonymous.4open.science/r/Multi-Task-Neural-Processes/.

the function priors in a hierarchical Bayesian inference framework. The shared knowledge from related tasks is incorporated into the context of each individual task, which serves as the inductive bias for making predictions in this task. The hierarchical architecture allows us to design expressive data-dependent priors. This enables the model to capture the complex task relationships in multi-task learning. By leveraging hierarchical modeling, multi-task neural processes are capable of exploring shared knowledge among related tasks in a principled way by specifying the function prior.

We validate the effectiveness of the proposed multi-task neural processes by extensive experiments in both multi-task classification and regression. The results demonstrate that multi-task neural processes can effectively capture task relatedness in the function space and consistently improve the performance of each individual task, especially in the limited data regime.

2 Preliminaries: Neural Processes

In this section, we briefly review vanilla neural processes (Garnelo et al., 2018b) based on which we derive our multi-task neural processes.

Let a data set be given composed of training samples and corresponding labels. In order to better reflect the desired model behaviour at test time (Garnelo et al., 2018b), the training data is split into the context set $D = (\mathbf{X}, \mathbf{Y})$ and a target set $D^* = (\mathbf{X}^*, \mathbf{Y}^*)$, where $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ is a subset of the training data, $\mathbf{Y} = \{\mathbf{y}_1, \cdots, \mathbf{y}_n\}$ the corresponding set of labels and n the size of the context set. To ensure both sets have the same data distribution, the context set is split from the target set. Now, given the context set $(\mathbf{X}, \mathbf{Y})$, we would like to estimate a function that can make predict the labels $\mathbf{Y}^*$ for target samples $\mathbf{X}^*$.

In general, we define a stochastic process by a random function $f : \mathcal{X} \to \mathcal{Y}$. Given the context set, we define the joint distribution over the function values $\{f(\mathbf{x}_1), \cdots, f(\mathbf{x}_n)\}$, which in Gaussian processes is a multivariate Gaussian distribution parameterized by a kernel function. The rationale of neural processes is to extract knowledge from the context set to specify the prior over the prediction function. Instead of a kernel function, *neural processes* adopt a deep neural network to define the prior distribution. Specifically, the model introduces a latent variable $\mathbf{z}$ to account for uncertainty in the predictions of $\mathbf{Y}^*$. The observed context set is encoded into the latent variable which is conditioned on the context set, i.e. it follows the prior distribution $p(\mathbf{z}|\mathbf{X}, \mathbf{Y})$. The latent variable $\mathbf{z}$ is a high-dimensional random vector parameterising the stochastic process by $f(\mathbf{X}^*) = g(\mathbf{X}^*, \mathbf{z})$. The function $g(\cdot, \cdot)$ is an extra fixed and learnable decoder function, which is also implemented by a neural network. Thus, the neural process model can be formulated as follows:

$$p(\mathbf{Y}^*|\mathbf{X}^*, \mathbf{X}, \mathbf{Y}) = \int p(\mathbf{Y}^*|\mathbf{X}^*, \mathbf{z}) p(\mathbf{z}|\mathbf{X}, \mathbf{Y}) d\mathbf{z}. \tag{1}$$

The graphical model for neural processes is shown in Figure 1 (a).

The neural process model is optimized using amortized variational inference. Let $q(\mathbf{z}|\mathbf{X}^*, \mathbf{Y}^*)$ be a variational posterior of the latent variable $\mathbf{z}$. The evidence lower-bound (ELBO) for neural processes is given as follows:

$$\log p(\mathbf{Y}^*|\mathbf{X}^*, \mathbf{X}, \mathbf{Y}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{X}^*, \mathbf{Y}^*)}[p(\mathbf{Y}^*|\mathbf{X}^*, \mathbf{z})] - \mathbb{D}_{\mathrm{KL}}[q(\mathbf{z}|\mathbf{X}^*, \mathbf{Y}^*)||p(\mathbf{z}|\mathbf{X}, \mathbf{Y})]. \tag{2}$$

In neural processes, the function space defined by deep neural networks allows the model to extract deep features while retaining a probabilistic interpretation (Jakkala, 2021). In multi-task learning, usually different tasks can be from different domains and have their specific data distributions (Lawrence & Platt, 2004; Long et al., 2017). Due to the complex data structure of multi-task learning, it is not straightforward to explore task relatedness in such function spaces. In this paper, we aim to extend the methodology of the neural process to the scenarios of multi-task learning to learn the shared knowledge among tasks for improving individual tasks.

3 Multi-Task Neural Processes

The common setup of multi-task learning is that there are multiple related tasks for which we would like to improve the learning of each individual one by sharing information across the different

tasks (Williams et al., 2007). Multi-task learning has been studied under different settings (Requeima et al., 2019; Williams et al., 2007; Lawrence & Platt, 2004; Yu et al., 2005; Long et al., 2017). In this paper, we tackle the multi-input multi-output setting, where each task has a different distribution while different tasks share the same target space (Lawrence & Platt, 2004; Yu et al., 2005; Long et al., 2017; Zhang et al., 2020). It aims to improve the overall performance of all multiple tasks simultaneously different from the sequential multi-task leaning (Requeima et al., 2019; Garnelo et al., 2018a). This is a challenging scenario due to the domain shift between tasks, which makes it sub-optimal to directly apply neural processes by incorporating the data from other related tasks into the context of each individual task.

3.1 Hierarchical Context modeling

Multi-task learning considers the estimation of random functions f_l, $l = 1, ..., L$ for each of the L related tasks. Each task l has its own training data, which is split into a context set $\mathcal{D}_l = (\mathbf{X}_l, \mathbf{Y}_l)$ and a target set $\mathcal{D}_l^* = (\mathbf{X}_l^*, \mathbf{Y}_l^*)$. $\mathbf{X}_l^* \in \mathbb{R}^{n_l^* \times d}$ and $\mathbf{X}_l \in \mathbb{R}^{n_l \times d}$ are inputs while $\mathbf{Y}_l^* \in \mathbb{R}^{n_l^* \times C}$ and $\mathbf{Y}_l \in \mathbb{R}^{n_l \times C}$ are outputs. d and C are the sizes of the input space and output space, respectively. n_l^* and n_l are the sizes of respectively the target and context set for the l-th task. Thus, we obtain the task-specific latent variables $\mathbf{f}_l = f_l(\mathbf{X}_l) \in \mathbb{R}^{n_l \times C}$. We use $\{D_l\}_{l=1}^L$ to denote all context sets in the dataset, which for brevity we represent as $\{\mathcal{D}_l\}$ and likewise will do for other sets.

Formulated this way the goal of multi-task learning is to predict $\{\mathbf{Y}_l^*\}$ for given $\{\mathbf{X}_l^*\}$ simultaneously with the assistance information $\{D_l\}$ from all tasks. To this end, we construct a joint prediction distribution with respect to the latent random functions $\{f_l\}$ as:

$$p(\{\mathbf{Y}_l^*\}|\{\mathbf{X}_l^*\}, \{D_l\}) = \prod_{l=1}^{L} p(\mathbf{Y}_l^*|\mathbf{X}_l^*, \{D_l\}) = \prod_{l=1}^{L} \int p(\mathbf{Y}_l^*|\mathbf{X}_l^*, f_l) p(f_l|\mathbf{M}) df_l. \tag{3}$$

Here, to enable shared knowledge to be transferred among tasks, we introduce a global variable $\mathbf{M}$ which works as a container to collect the useful information from $\{D_l\}$ of all tasks. In contrast to neural processes for single tasks, the global variable $\mathbf{M}$ provides the contextual information from all tasks for each individual task. The concrete formation of $\mathbf{M}$ depends on the learning scenarios. In practice, $\mathbf{M}$ contains the aggregated samples from different classes in classification tasks or feature vectors in regression tasks.

Similar to Gaussian processes, we assume that the function value is $\mathbf{Y}_l^* = f_l(\mathbf{X}_l^*) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ is the observation noise. Thus, we can define the predictive likelihood on the target set as follows:

$$p(\mathbf{Y}_l^*|\mathbf{X}_l^*, f_l) = \mathcal{N}(\mathbf{Y}_l^*|f_l(\mathbf{X}_l^*), \sigma^2). \tag{4}$$

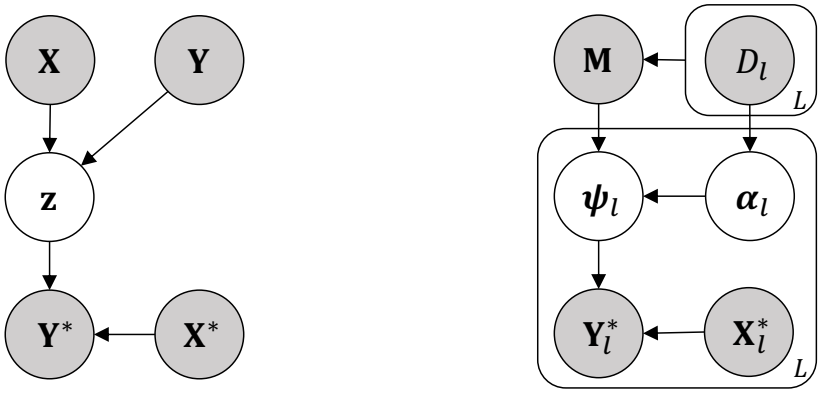
We can insert this function into (3), and then the model for each task l is further specified by:

$$p(\mathbf{Y}_l^*|\mathbf{X}_l^*, \{D_l\}) = \int \mathcal{N}(\mathbf{Y}_l^*|f_l(\mathbf{X}_l^*), \sigma^2) p_\theta(f_l|\mathbf{M}) df_l. \tag{5}$$

In order to combine Gaussian processes and neural networks in the context of multi-task learning, we define $p_\theta(f_l|\mathbf{M})$ as a deep neural network in place of a Gaussian distribution parameterized by a kernel function. To be more specific, we assume that f_l is parameterized by a random variable $\boldsymbol{\psi}_l$ by defining $f_l(\mathbf{X}) = \boldsymbol{\psi}_l^\top \mathbf{X}$. We specify a data dependent prior by conditioning $\boldsymbol{\psi}_l$ on the global variable $\mathbf{M}$. In this way, we incorporate the transferable knowledge into the learning of the prediction function of the current tasks. Thus, the predictive distribution for the l-th task over its target set can be formulated as follows:

$$p(\mathbf{Y}_l^*|\mathbf{X}_l^*, \{D_l\}) = \int \mathcal{N}(\mathbf{Y}_l^*|\boldsymbol{\psi}_l^\top \mathbf{X}_l^*, \sigma^2) p_\theta(\boldsymbol{\psi}_l|\mathbf{M}) d\boldsymbol{\psi}_l. \tag{6}$$

In effect, $\boldsymbol{\psi}_l$ denotes the parameters of the classifier for classification tasks or parameters of regressors for regression tasks. Particularly, for classification tasks, the task specific classifier is denoted as $\boldsymbol{\psi}_l \in \mathbb{R}^{d \times C}$, where d is the dimension of the input feature and C is the number of classes in the dataset. As done in (Requeima et al., 2019), we generate each column of $\boldsymbol{\psi}_l$ independently from the context samples of the corresponding class. In our case, each column of $\boldsymbol{\psi}_l$ encodes the context information of its class from all tasks $p_\theta(\boldsymbol{\psi}_l|\mathbf{M}) = \prod_{c=1}^{C} p_\theta(\boldsymbol{\psi}_{l,c}|\mathbf{M}_c)$.



(a) Neural processes (b) Multi-task neural processes

Figure 1: Graphical illustration of neural processes and multi-task neural processes. Shaded nodes indicate observed variables, and white nodes indicate unobserved latent variable.

Directly aggregating $\mathbf{M}$ as done in neural processes for single tasks is not applicable for multi-task learning due to the distribution shift between tasks. The data from related tasks should be processed and adapted to the current task as the contextual information. To this end, we introduce a higher-level latent variable $\boldsymbol{\alpha}_l$ to extract the shared knowledge from $\mathbf{M}$, which is conditioned on the data D_l of each task. This results in a hierarchical Bayesian modeling of functions in the neural process:

$$p_\theta(\boldsymbol{\psi}_l|\mathbf{M}) = \int p_{\theta_1}(\boldsymbol{\psi}_l|\boldsymbol{\alpha}_l, \mathbf{M}) p_{\theta_2}(\boldsymbol{\alpha}_l|D_l) d\boldsymbol{\alpha}_l, \tag{7}$$

where $\boldsymbol{\alpha}_l$ is the latent variable to control the access to shared knowledge for each task, which is used to explore the relevant knowledge to the task l. $p_{\theta_1}(\boldsymbol{\psi}_l|\boldsymbol{\alpha}_l, \mathbf{M})$ and $p_{\theta_2}(\boldsymbol{\alpha}_l|D_l)$ are prior distributions of the latent variable $\boldsymbol{\psi}_l$ and $\boldsymbol{\alpha}_l$, respectively, which are parameterized with neural networks. To be specific, we define $p_{\theta_1}(\boldsymbol{\psi}_l|\boldsymbol{\alpha}_l, \mathbf{M}) = \mathcal{N}(\boldsymbol{\psi}_l|\mu(\mathbf{m}_l), \Sigma(\mathbf{m}_l))$. Here $\mathbf{m}_l$ contains the relevant knowledge to the task l, which is adapted from the global variable $\mathbf{M}$ by a deterministic function $\mathbf{m}_l = h(\boldsymbol{\alpha}_l, \mathbf{M})$, where $h(\cdot)$ is a learnable function implemented with a neural network.

By substituting (7) into (6), we obtain the model of multi-task neural processes as follows:

$$p(\{\mathbf{Y}_l^*\}|\{\mathbf{X}_l^*\}, \{D_l\}) = \prod_{l=1}^{L} \int \int \mathcal{N}(\mathbf{Y}_l^*|\boldsymbol{\psi}_l^\top \mathbf{X}_l^*, \sigma^2) p_{\theta_1}(\boldsymbol{\psi}_l|\boldsymbol{\alpha}_l, \mathbf{M}) p_{\theta_2}(\boldsymbol{\alpha}_l|D_l) d\boldsymbol{\psi}_l d\boldsymbol{\alpha}_l. \tag{8}$$

This hierarchical Bayesian model provides a principled way to explore shared knowledge among tasks in the function space and incorporate it as inductive bias into the learning of each individual task by specifying a data dependent prior. The graphical model of the multi-task neural processes is shown in Figure 1 (b).

3.2 Variational Hierarchical Inference

The previous section developed the model of multi-tasks neural processes with a hierarchical context model. We now describe how to optimize the model and obtain the predictions by leveraging a variational Bayesian inference framework. To that end, based on a conditional independence assumption, we introduce the variational joint posterior distribution factorized as follows:

$$q_\varphi(\{\boldsymbol{\psi}_l\}, \{\boldsymbol{\alpha}_l\}|\{\mathcal{D}_l^*\}) = \prod_{l=1}^{L} q_\varphi(\boldsymbol{\psi}_l, \boldsymbol{\alpha}_l|\mathcal{D}_l^*) = \prod_{l=1}^{L} q_{\varphi_1}(\boldsymbol{\psi}_l|\mathcal{D}_l^*) q_{\varphi_2}(\boldsymbol{\alpha}_l|\mathcal{D}_l^*), \tag{9}$$

where $q_{\varphi_1}(\boldsymbol{\psi}_l|\mathcal{D}_l^*)$ and $q_{\varphi_2}(\boldsymbol{\alpha}_l|\mathcal{D}_l^*)$ are variational posteriors of the latent variables $\boldsymbol{\psi}_l$ and $\boldsymbol{\alpha}_l$ for the task l, respectively. Both variational posteriors are parameterized as Gaussian distributions and inferred by amortized inference networks parameterized by φ_1 and , φ_2, respectively. We make use of the amortized variational inference technique (Kingma & Welling, 2013) to learn such distributions over latent variables.

Learning By incorporating the variational posteriors into (9), we derive the ELBO for the multi-task neural processes as follows:

$$\log p(\{\mathbf{Y}_l^*\}|\{\mathbf{X}_l^*\},\{D_l\}) \geq \sum_{l=1}^{L} \Big\{ \mathbb{E}_{q_{\varphi_2}(\boldsymbol{\alpha}_l|\mathcal{D}_l^*)} \{ \mathbb{E}_{q_{\varphi_1}(\boldsymbol{\psi}_l|\mathcal{D}_l^*)} [p(\mathbf{Y}_l^*|\mathbf{X}_l^*, \boldsymbol{\psi}_l)] - \mathbb{D}_{\mathrm{KL}}[q_{\varphi_1}(\boldsymbol{\psi}_l|\mathcal{D}_l^*)||p_{\theta_1}(\boldsymbol{\psi}_l|\boldsymbol{\alpha}_l, \mathbf{M})] \} - \mathbb{D}_{\mathrm{KL}}[q_{\varphi_2}(\boldsymbol{\alpha}_l|\mathcal{D}_l^*)||p_{\theta_2}(\boldsymbol{\alpha}_l|D_l)] \Big\}. \tag{10}$$

The detailed derivation is provided in Appendix A. To maximize the lower bound, we sample $\boldsymbol{\alpha}_l$ and $\boldsymbol{\psi}_l$ from the variational posteriors to approximate the inner expectation by adopting the reparameterization trick (Kingma & Welling, 2013). In doing so, we obtain the following empirical objective:

$$\hat{L}_{\mathrm{MTNP}}(\theta, \varphi) = \sum_{l=1}^{L} \Big\{ \frac{1}{N_a} \sum_{i=1}^{N_a} \{ \frac{1}{N_f} \sum_{j=1}^{N_f} [-\log p(\mathbf{Y}_l^*|\mathbf{X}_l^*, \boldsymbol{\psi}_l^{(j)})] + \lambda_f \mathbb{D}_{\mathrm{KL}}[q_{\varphi_1}(\boldsymbol{\psi}_l|\mathcal{D}_l^*)||p_{\theta_1}(\boldsymbol{\psi}_l|\boldsymbol{\alpha}_l^{(i)}, \mathbf{M})\} + \lambda_a \mathbb{D}_{\mathrm{KL}}[q_{\varphi_2}(\boldsymbol{\alpha}_l|\mathcal{D}_l^*)||p_{\theta_2}(\boldsymbol{\alpha}_l|D_l)] \Big\}, \tag{11}$$

where $\boldsymbol{\psi}_l^{(j)} \sim q_{\varphi_1}(\boldsymbol{\psi}_l|\mathcal{D}_l^*)$ and $\boldsymbol{\alpha}_l^{(i)} \sim q_{\varphi_2}(\boldsymbol{\alpha}_l|\mathcal{D}_l^*)$. N_f and N_a are the number of Monte Carlo samples for the variational posteriors of $\boldsymbol{\psi}$ and $\boldsymbol{\alpha}$, respectively. λ_f and λ_a are the hyperparameters to help stably train the KL-divergence terms, which are set following the annealing scheme of (Bowman et al., 2015).

Prediction Having the learned model, we can make prediction on the target set. Given a test sample $\mathbf{x}_l$ from the l-th task, we can produce the predictive distribution which involves the prior distributions $p_{\theta_1}(\boldsymbol{\psi}_l|\boldsymbol{\alpha}_l, \mathbf{M})$ and $p_{\theta_2}(\boldsymbol{\alpha}_l|D_l)$. The predictive distribution with the integration for the introduced latent variables is formulated as:

$$p(\mathbf{y}_l|\mathbf{x}_l, \{D_l\}) = \int \int p(\mathbf{y}_l|\mathbf{x}_l, \boldsymbol{\psi}_l) p_{\theta_1}(\boldsymbol{\psi}_l|\boldsymbol{\alpha}_l, \mathbf{M}) p_{\theta_2}(\boldsymbol{\alpha}_l|D_l) d\boldsymbol{\alpha}_l d\boldsymbol{\psi}_l \tag{12}$$

Here we also need to apply the Monte Carlo estimation over (12) and obtain the predictions as follows:

$$p(\mathbf{y}_l|\mathbf{x}_l) \approx \frac{1}{N_a} \sum_{i=1}^{N_a} \frac{1}{N_f} \sum_{j=1}^{N_f} p(\mathbf{y}_l|\mathbf{x}_l, \boldsymbol{\psi}_l^{(j)})], \tag{13}$$

where $\boldsymbol{\psi}_l^{(j)} \sim p_{\theta_1}(\boldsymbol{\psi}_l|\boldsymbol{\alpha}_l^{(i)}, \mathbf{M})$ and $\boldsymbol{\alpha}_l^{(i)} \sim p_{\theta_2}(\boldsymbol{\alpha}_l|D_l)$.

4 Related Work

Neural network based models have achieved impressive results on various applications (LeCun et al., 2015). However, due to the large number of parameters these neural models need large-scale data with annotation. It is challenging to train a deep neural model that generalizes well with number-limited labeled data. To reduce such labeling consumption, recent works (Long et al., 2017; Liu et al., 2016) follow a multi-task learning strategy to fully leverage information from relevant tasks as inductive bias (Caruana, 1997) to improve each task's performance. The main challenge of multi-task learning is that each task is provided with a limited amount of labeled data, which is insufficient to build reliable classifiers without overfitting (Long et al., 2017).

Multi-task learning (MTL) aims to learn several tasks simultaneously and improve their overall performance. The crux of MTL is how to explore task relatedness from different tasks, which could be particularly significant when limited data for each task is available (Long et al., 2017; Zhang et al., 2020). Recently, the task relatedness is learned in many different aspects of the model, e.g., loss functions (Liu et al., 2020; Qian et al., 2020; Kendall et al., 2018), parameter space (Long et al., 2017; Bakker & Heskes, 2003), or representation space (Misra et al., 2016). In this paper, we focus on the data-insufficient problem for multi-task learning. Different from other branches of MTL models (Huang et al., 2021; Fu et al., 2021; Phillips et al., 2021) which leverage several channels of supervision information simultaneously included by the same input, our data setting allows each

task to have its own individual data. This is much more challenging due to the distribution shift between inputs of different tasks.

In previous works (Lawrence & Platt, 2004; Yu et al., 2005; Yousefi et al., 2019), multi-task learning benefits from Gaussian Processes by generally incorporating the shared information in the Gaussian Processes prior to synergize several random functions of different tasks simultaneously. For each task, the Gaussian prior of its individual function is $p(f_l|\mathbf{X}_l, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, \mathbf{K}_l)$. $\boldsymbol{\theta}$ is the shared kernel parameter vector, which allows task specific random functions to share some common structure (Lawrence & Platt, 2004; Yu et al., 2005). In order to consider the negative transfer from outlier tasks, some works (Williams et al., 2007; Yu et al., 2007; Zhang & Yeung, 2010) specify the task covariance matrix to model the positive and negative transfer. In general, Gaussian processes have a few key limitations which limit the models' applicability, in particular computation cost, storage cost, and the lack of kernel functions capable of handling structured data (Jakkala, 2021). In the context of multi-task learning, such drawbacks are further complicated due to the more varied data distributions and more available training data. As an alternative to using Gaussian processes for multi-task learning, nowadays deep neural networks have become popular as parameterized functions, which construct an information sharing architecture between tasks.

Recently, neural processes (Garnelo et al., 2018b) and the related works (Garnelo et al., 2018a; Kim et al., 2019; Wang & Van Hoof, 2020; Requeima et al., 2019; Gordon et al., 2019) combine Gaussian Processes and neural networks, which are not only computationally efficient but also retain a probabilistic interpretation of the model. Previous works related to Neural Processes offer an efficient and flexible way to model the context data in the function space for prediction. Similar with Deep Kernel Learning (Wilson et al., 2016) where a Gaussian Process is applied to learned representations of data, one branch (Garnelo et al., 2018b; Kim et al., 2019; Wang & Van Hoof, 2020) models the predictive distribution $p(\mathbf{Y}^*|\ g(\mathbf{X}^*, \mathbf{z}))$ by introducing a latent representation variable $\mathbf{z}$. This requires an extra neural network $g(\cdot, \cdot)$ to obtain the prediction value. Another branch (Requeima et al., 2019; Gordon et al., 2019) is directly modeling the predictive distribution $p(\mathbf{Y}^*|\boldsymbol{\psi}(\mathbf{X}^*))$ by introducing a latent parameter variable $\boldsymbol{\psi}$, acting as the parameters of the neural network. Whereas those previous works focused on the context information from its own task to generate the task-specific latent variables, here we extend the methodology and study how to incorporate the shared knowledge provided by related tasks into the context to explore task relatedness in the function space for multi-task learning.

Hierarchical modeling in the Bayesian framework has been successful to design the form of the prior (Daumé III, 2009; Zhao et al., 2017; Klushyn et al., 2019; Wang & Van Hoof, 2020) and posterior distributions (Ranganath et al., 2016; Krueger et al., 2017; Zhen et al., 2020) based on many observations. It allows the latent variable to follow a complicated distribution and forms a highly flexible approximation (Krueger et al., 2017). In multi-task learning, hierarchical modeling helps our model to fit the context sets from related tasks in a specific task, fully utilizing the shared knowledge provided by related tasks.

5 Experiments and Results

We conduct extensive experiments to evaluate the proposed multi-task neural processes on benchmark datasets for regression, classification and segmentation tasks to demonstrate their effectiveness in exploring task relatedness for multi-task learning.

5.1 Experimental setup

To show the effectiveness of the proposed multi-tasks neural processes, we conduct a thorough comparison implementing multiple different baseline models. Single task learning is implemented by task-specific feature extractors and classifiers without knowledge sharing among tasks. Basic multi-task learning shares feature extractors and adds task specific classifiers. We also define variational extensions of the single task learning and basic multi-task learning, which cast models as variational Bayesian problems and treat classifiers as latent variables (Shen et al., 2021). Since the major contributions of this work are the extension of neural processes for multi-task learning and hierarchical context modeling, we implement neural processes and its variant, neural processes with all task context for comparison. For each task, neural processes encode only its specific context observations

Table 1: Performance comparison (average accuracy) on `Office-Home` training with 5%, 10%, 20% labeled data. The higher the better.

Methods	5%	10%	20%
Single task learning	49.2±0.2	58.3±0.1	64.9±0.1
Variational single task learning	51.1±0.1	60.2±0.2	65.8±0.2
Basic multi-task learning	50.4±0.1	59.5±0.1	65.6±0.1
Variational basic multi-task learning	51.3±0.1	60.9±0.1	67.0±0.2
Neural processes	54.7±0.1	59.4±0.2	69.3±0.2
Neural processes with all task context	59.1±0.2	62.4±0.1	69.1±0.1
Multi-task neural processes	**60.0±0.1**	**63.3±0.1**	**69.9±0.3**

into the latent variables to model prediction functions by a deep neural network. Different tasks share the parameters of deep neural networks. Neural processes with all task context is a straightforward extension of neural processes for multi-task learning, which treats context sets from all tasks equally without hierarchical context modeling. For a fair comparison, all the above-mentioned methods share the same architecture of the feature extractor.

5.2 Multi-Task Classification

We evaluate the performance of our multi-task neural processes on real-world multi-task classification, where different tasks are defined as image classification problems in different domains. The tasks are related to each other since they share the same label space. We conduct experiments on `Office-Home`, `Office-Caltech` and `ImageCLEF`. `Office-Home` (Venkateswara et al., 2017) contains images from four domains/tasks: Artistic (A), Clipart (C), Product (P) and Real-world (R). Each task contains images from 65 object categories collected under office and home settings. There are about $15,500$ images in total. `Office-Caltech` (Gong et al., 2012) contains the ten categories shared between Office-31 (Saenko et al., 2010) and Caltech-256 (Griffin et al., 2007). One task uses data from Caltech-256 (C), and the other three tasks use data from Office-31, whose images were collected from three distinct domains/tasks, namely Amazon (A), Webcam (W) and DSLR (D). There are $8 \sim 151$ samples per category per task, and $2,533$ images in total. `ImageCLEF` (Long et al., 2017), the benchmark for the ImageCLEF domain adaptation challenge, contains 12 common categories shared by four public datasets/tasks: Caltech-256 (C), ImageNet ILSVRC 2012 (I), Pascal VOC 2012 (P), and Bing (B). There are $2,400$ images in total.

Table 2: Performance comparison of different methods on the `Office-Home` dataset.

Methods	5%					10%				
	A	C	P	R	Avg.	A	C	P	R	Avg.
Single task learning	36.7±0.4	30.8±0.5	67.5±0.3	61.7±0.3	49.2±0.2	50.4±0.3	40.8±0.3	74.4±0.4	67.5±0.4	58.3±0.1
Bakker & Heskes (2003)	40.0±0.1	33.6±0.3	69.8±0.4	63.6±0.3	52.8±0.1	52.5±0.3	42.3±0.4	75.7±0.5	69.5±0.5	60.0±0.2
Long et al. (2017)	47.8±0.4	37.9±0.2	73.6±0.3	70.4±0.2	57.4±0.1	57.2±0.3	43.3±0.1	78.7±0.3	74.4±0.1	**63.4±0.2**
Kendall et al. (2018)	40.2±0.2	33.6±0.4	69.5±0.2	63.7±0.1	51.8±0.1	49.1±0.1	38.7±0.3	73.4±0.2	67.4±0.3	57.2±0.2
Qian et al. (2020)	37.9±0.3	31.4±0.2	67.7±0.3	62.4±0.2	49.9±0.2	47.1±0.2	37.2±0.1	70.5±0.2	66.3±0.3	55.3±0.1
Multi-task neural processes	55.0±0.2	40.8±0.3	74.2±0.2	69.9±0.2	**60.0±0.1**	59.5±0.3	44.4±0.5	77.3±0.3	72.0±0.3	63.3±0.1

Table 3: Performance comparison of different methods on the `Office-Caltech` dataset.

Methods	5%					10%				
	A	W	D	C	Avg.	A	W	D	C	Avg.
Single task learning	87.4±0.4	87.9±0.3	96.4±0.5	82.8±0.2	88.6±0.3	92.8±0.5	97.7±0.3	87.8±0.2	84.3±0.4	90.7±0.2
Bakker & Heskes (2003)	93.2±0.2	94.0±0.3	94.7±0.3	85.4±0.4	91.8±0.1	94.9±0.4	97.6±0.5	96.6±0.5	90.9±0.4	95.0±0.2
Long et al. (2017)	92.7±0.2	94.3±0.2	97.1±0.2	89.2±0.6	93.4±0.2	95.0±0.3	98.1±0.4	95.0±0.5	91.3±0.2	94.8±0.3
Kendall et al. (2018)	93.6±0.4	92.5±0.2	95.0±0.5	83.9±0.5	91.2±0.3	94.9±0.5	96.2±0.4	93.6±0.3	90.4±0.2	93.8±0.2
Qian et al. (2020)	92.6±0.3	90.9±0.2	95.7±0.4	85.2±0.6	91.1±0.3	94.2±0.4	97.0±0.4	95.0±0.3	90.2±0.3	94.1±0.3
Multi-task neural processes	94.6±0.1	95.8±0.2	97.9±0.0	90.2±0.1	**94.6±0.1**	95.1±0.1	97.7±0.1	97.1±0.3	91.6±0.3	**95.4±0.1**

Table 4: Performance comparison of different methods on the `ImageCLEF` dataset.

Methods	5%					10%				
	C	I	P	B	Avg.	C	I	P	B	Avg.
Single task learning	85.4±0.6	71.4±0.4	57.7±0.2	36.0±0.2	62.6±0.2	88.9±0.5	77.8±0.3	64.3±0.2	47.6±0.5	69.7±0.3
Bakker & Heskes (2003)	90.9±0.4	85.4±0.6	68.1±0.3	51.4±0.5	73.9±0.3	91.0±0.5	87.1±0.3	73.4±0.4	54.5±0.2	76.5±0.4
Long et al. (2017)	90.1±0.5	76.5±0.5	72.8±0.3	54.9±0.4	73.7±0.4	93.3±0.4	83.2±0.6	70.4±0.4	56.3±0.4	75.8±0.2
Kendall et al. (2018)	93.2±0.6	86.1±0.4	68.6±0.3	50.4±0.4	74.6±0.2	91.9±0.3	88.9±0.5	74.3±0.3	52.4±0.2	76.9±0.3
Qian et al. (2020)	91.6±0.3	85.8±0.4	68.4±0.3	50.2±0.4	74.0±0.4	90.7±0.4	88.1±0.6	75.6±0.4	54.6±0.3	77.3±0.3
Multi-task neural processes	90.5±0.3	84.9±0.2	70.2±0.2	58.9±0.4	**76.1±0.1**	93.5±0.4	88.5±0.3	74.6±0.4	61.7±0.3	**79.6±0.1**

Table 5: Performance of multi-task regression (normalized mean squared errors) for rotation angle estimation. The lower the better.

Methods	0	1	2	3	4	5	6	7	8	9	Mean
Single task learning	.138	.158	.245	.216	.327	.150	.229	.286	.209	.191	.215
Variational single task learning	.144	.161	.330	.197	.382	.208	.189	.296	.135	.194	.224
Basic multi-task learning	.114	.124	.115	.124	.115	.114	.114	.125	.115	.115	.118
Variational basic multi-task learning	.121	.124	.121	.123	.121	.121	.121	.124	.121	.121	.121
Neural processes	.193	.058	.105	.067	.101	.120	.158	.107	.083	.126	.112
Neural processes with all task context	.188	.064	.114	.063	.103	.116	.167	.095	.067	.111	.109
Multi-task neural processes	.183	.060	.098	.067	.109	.109	.160	.092	.077	.113	**.106**

We adopt the standard evaluation protocols (Zhang & Yang, 2021) for the `Office-Home`, `Office-Caltech` and `ImageCLEF` datasets, randomly selecting 5% and 10% of samples from each task in the dataset as the training set, using the remaining samples as the test set (Long et al., 2017). Note that when we use 5%, 10% and 20% of labeled data for training, there are on average about 3, 6 and 12 samples per category per task, respectively. In this setting, each task has a limited amount of data, which is insufficient for building reliable classifiers without overfitting. For all three benchmarks, we pre-extract the input feature by adopting the VGGnet without the final classifier layer, as the same setting of (Long et al., 2017). The architectures of inference networks used in our method are provided in Appendix B. In this section, all the results with error bars are obtained based on a 95% confidence interval from five runs.

We provide comprehensive comparisons on `Office-Home` in Table 1, which is a more challenging multi-task classification dataset with 65 categories. The results show that the proposed multi-task neural processes outperform other counterpart methods. The variational version of single task learning and basic multi-task learning show better performance than themselves, which demonstrates the benefits of the variational Bayesian framework. Training with all task context improves the performance of neural processes by a large margin when 5% labeled data is available, showing the benefit of exploring shared knowledge from related tasks, especially with limited data. Noticeably compared to neural processes with all task context, the proposed multi-task neural processes benefit from hierarchical context modeling and show even better performance.

More comparison results on the `Office-Home`, `Office-Caltech`, `ImageCLEF` datasets are shown in Tables 2, 3 and 4, respectively. Experimental results on all three benchmarks with 20% labeled data are provided in Appendix C. In multi-task learning, the average accuracy of all tasks is used for overall performance measurement. The best results are marked in bold. The proposed multi-task neural processes achieve competitive and even better performance on such multi-task classification datasets with different train-test splits. It is worth highlighting that in the more challenging setting, training with fewer training data of each task, the proposed multi-task neural processes shows a larger performance advantage over compared methods. This demonstrates the effectiveness of our method in handling context information from related tasks to improve the performance of each task from limited data.

5.3 Multi-Task Regression

In order to show the effectiveness of multi-task neural processes for multi-task regression, we apply our multi-task neural processes to rotation angle estimation on the `Rotated MNIST` dataset (LeCun et al., 1998). We adopt this dataset to study multi-task regression, where each task is an angle estimation problem for each digit and different tasks corresponding to different digits are related because they share the same rotation angle space. Each image is rotated by 0° through 90° in intervals of 10°, where the rotation angle is the regression target. We randomly choose six samples per task per angle as the training set. For a fair comparison, all methods share the same split settings. For end-to-end training, we design a deep neural network with three convolutional layers and two full-connected layers to extract feature representations of 2048 dimensions.

The comparison results are shown in Table 5. Since we would like to improve the overall performance of all regression tasks, we use the Mean of normalized mean squared errors of all tasks as the measurement. The proposed multi-task neural processes outperform other counterpart methods by yielding an overall lower mean error. It is interesting to mention that for some digits, e.g., `0` and `6`, their estimation errors are higher than other digits. The reason could be that these digits have some degree of rotational invariance, which makes it harder for them to regress their target angles.

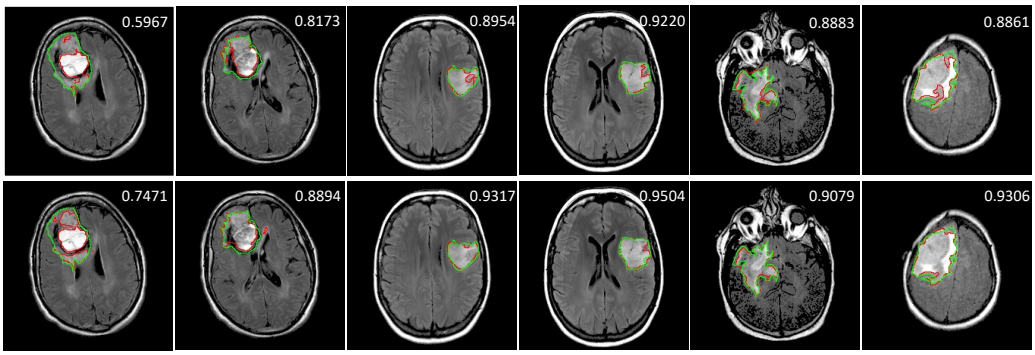


Figure 2: The segmentation results by the proposed multi-task neural processes (bottom row) and the U-Net (upper row). Green outline corresponds to the ground truth and red to the segmentation output. The numbers are the DSC scores compared against ground truth. Our multi-task neural processes can predict contours closer to the ground truth ones with higher DSC scores than U-Net, indicating the advantages of exploring spatial context information for brain image segmentation.

The consistent better performance over vanilla neural processes demonstrates the effectiveness of our multi-task neural processes in capturing task relatedness for improving individual tasks.

5.4 Brain Image Segmentation

In this section, we demonstrate that multi-task neural processes are also able to explore spatial context information to improve image segmentation. To this end, we adopt a brain image dataset (Buda et al., 2019) with lower-grade gliomas collected from 110 patients. The number of images varies among patients from 20 to 88. The goal is to segment the tumor in each brain image.

To apply our multi-task neural processes, we cast the pixel-wise segmentation as a multi-task regression problem. To be specific, we consider the prediction of each pixel to be a regression task. For the task l, we define Ω_l as a local region centered at the spatial position l, which provides the local context information. We fit our model in this multi-task regression setting to produce pixel-wise predictions. The function prior in our model here can be formulated as $p(\boldsymbol{\psi}_l|\{\Omega_l\})$. Each task incorporates the shared knowledge provided by related tasks into its context of the prediction function. This offers an effective way to model the long-range interdependence of pixels in one image. For implementation, we use the U-Net architecture (Ronneberger et al., 2015) as the backbone and add our model as the final layer. We evaluate our method and U-Net on the brain segmentation dataset. The results show that the proposed multi-task neural processes surpass the baseline U-Net by 0.5% in terms of dice similarity coefficients (DSC) for the overall validation set. Figure 2 shows segmentation results of the proposed multi-task neural processes (bottom row) and the U-Net (upper row), where the green outline corresponds to the ground truth and the red to the segmentation output. Our multi-task neural processes predict contours closer to the ground truth. This demonstrates the advantages of exploring context information by multi-task neural processes for segmentation.

6 Conclusion

In this paper, we develop multi-task neural processes, a new variant of neural processes for multi-task learning. We propose to explore the task relatedness in the function space by specifying the function priors in a hierarchical Bayesian inference framework. The shared knowledge from related tasks is incorporated into the context of each individual task, which serves as the inductive bias for making predictions of this task. The hierarchical architecture allows us to design expressive data-dependent prior, enabling the model to explore the complex task relationships in multi-task learning. By leveraging the hierarchical modeling, multi-tasks neural processes are capable of capturing the shared knowledge from other tasks in a principled way by specifying the function prior. We evaluate multi-task neural processes on multi-task regression and classification datasets. Results demonstrate the effectiveness of multi-task neural processes in transferring useful knowledge among tasks for multi-task learning.

ETHICS STATEMENT

Since our method is a general contribution to the multi-task learning field, it has the potential to benefit any societal problem where multiple tasks are performed simultaneously, e.g., medical diagnosis. Accordingly, the method would also potentially face some negative social impacts accompanying with these applications, e.g., lack of fairness with the model trained by incomplete data due to the privacy of patients in medical imaging tasks.

REPRODUCIBILITY STATEMENT

To make our method reproducible, we provide the details of the method in Section 3. We also give the derivations of the evidence lower bound proposed in Section 3 in Appendix A. The datasets used in this paper can be obtained by referencing Section 5. There are also the experimental settings and implementation details for each dataset in Section 5. We also provide an anonymous link for the code of our method, which is in the abstract of this paper.

REFERENCES

BJ Bakker and TM Heskes. Task clustering and gating for bayesian multitask learning. 2003.

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

Mateusz Buda, Ashirbani Saha, and Maciej A Mazurowski. Association of genomic subtypes of lower-grade gliomas with shape features automatically extracted by a deep learning algorithm. *Computers in biology and medicine*, 109:218–225, 2019.

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

Andreas Damianou and Neil D Lawrence. Deep gaussian processes. In *Artificial intelligence and statistics*, pp. 207–215. PMLR, 2013.

Hal Daumé III. Bayesian multitask learning with latent hierarchies. *arXiv preprint arXiv:0907.0783*, 2009.

Gang Fu, Qing Zhang, Lei Zhu, Ping Li, and Chunxia Xiao. A multi-task network for joint specular highlight detection and removal. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7752–7761, 2021.

Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Rezende, and SM Ali Eslami. Conditional neural processes. In *International Conference on Machine Learning*, pp. 1704–1713. PMLR, 2018a.

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J Rezende, SM Eslami, and Yee Whye Teh. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018b.

Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2066–2073. IEEE, 2012.

Jonathan Gordon, Wessel P Bruinsma, Andrew YK Foong, James Requeima, Yann Dubois, and Richard E Turner. Convolutional conditional neural processes. *arXiv preprint arXiv:1910.13556*, 2019.

Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007.

Zhizhong Huang, Junping Zhang, and Hongming Shan. When age-invariant face recognition meets face age synthesis: A multi-task learning framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7282–7291, 2021.

Kalvik Jakkala. Deep gaussian processes: A survey. *arXiv preprint arXiv:2106.12135*, 2021.

Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7482–7491, 2018.

Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. *arXiv preprint arXiv:1901.05761*, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Alexej Klushyn, Nutan Chen, Richard Kurle, Botond Cseke, and Patrick van der Smagt. Learning hierarchical priors in vaes. *arXiv preprint arXiv:1905.04982*, 2019.

David Krueger, Chin-Wei Huang, Riashat Islam, Ryan Turner, Alexandre Lacoste, and Aaron Courville. Bayesian hypernetworks. *arXiv preprint arXiv:1710.04759*, 2017.

Neil D Lawrence and John C Platt. Learning to learn with the informative vector machine. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 65, 2004.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Liyang Liu, Yi Li, Zhanghui Kuang, Jing-Hao Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. Towards impartial multi-task learning. In *International Conference on Learning Representations*, 2020.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Deep multi-task learning with shared memory. *arXiv preprint arXiv:1609.07222*, 2016.

Mingsheng Long, Zhangjie Cao, Jianmin Wang, and S Yu Philip. Learning multiple tasks with multilinear relationship networks. In *Advances in neural information processing systems*, pp. 1594–1603, 2017.

Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3994–4003, 2016.

John Phillips, Julieta Martinez, Ioan Andrei Bârsan, Sergio Casas, Abbas Sadat, and Raquel Urtasun. Deep multi-task learning for joint localization, perception, and prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4679–4689, 2021.

Weizhu Qian, Bowei Chen, Yichao Zhang, Guanghui Wen, and Franck Gechter. Multi-task variational information bottleneck. *arXiv preprint arXiv:2007.00339*, 2020.

Rajesh Ranganath, Dustin Tran, and David Blei. Hierarchical variational models. In *International Conference on Machine Learning*, pp. 324–333. PMLR, 2016.

Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pp. 63–71. Springer, 2003.

James Requeima, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard E Turner. Fast and flexible multi-task classification using conditional neural adaptive processes. *Advances in Neural Information Processing Systems*, 32:7959–7970, 2019.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European conference on computer vision*, pp. 213–226. Springer, 2010.

Jiayi Shen, Xiantong Zhen, Marcel Worring, and Ling Shao. Variational multi-task learning, 2021. URL https://openreview.net/forum?id=kPheYCFm0Od.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *(IEEE) Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Michael Volpp, Fabian Flürenbrock, Lukas Grossberger, Christian Daniel, and Gerhard Neumann. Bayesian context aggregation for neural processes. In *International Conference on Learning Representations*, 2020.

Qi Wang and Herke Van Hoof. Doubly stochastic variational inference for neural processes with hierarchical latent variables. In *International Conference on Machine Learning*, pp. 10018–10028. PMLR, 2020.

Chris Williams, Edwin V Bonilla, and Kian M Chai. Multi-task gaussian process prediction. *Advances in neural information processing systems*, pp. 153–160, 2007.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *Artificial intelligence and statistics*, pp. 370–378. PMLR, 2016.

Fariba Yousefi, Michael Thomas Smith, and Mauricio A Álvarez. Multi-task learning for aggregated data using gaussian processes. *arXiv preprint arXiv:1906.09412*, 2019.

Kai Yu, Volker Tresp, and Anton Schwaighofer. Learning gaussian processes from multiple tasks. In *Proceedings of the 22nd international conference on Machine learning*, pp. 1012–1019, 2005.

Shipeng Yu, Volker Tresp, and Kai Yu. Robust multi-task learning with t-processes. In *Proceedings of the 24th international conference on Machine learning*, pp. 1103–1110, 2007.

Yi Zhang, Yu Zhang, and Wei Wang. Deep multi-task learning via generalized tensor trace norm. *arXiv preprint arXiv:2002.04799*, 2020.

Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

Yu Zhang and Dit-Yan Yeung. Multi-task learning using generalized t process. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 964–971. JMLR Workshop and Conference Proceedings, 2010.

Shengjia Zhao, Jiaming Song, and Stefano Ermon. Learning hierarchical features from generative models. *arXiv preprint arXiv:1702.08396*, 2017.

Xiantong Zhen, Yingjun Du, Huan Xiong, Qiang Qiu, Cees GM Snoek, and Ling Shao. Learning to learn variational semantic memory. *arXiv preprint arXiv:2010.10341*, 2020.

A DERIVATION OF THE ELBO FOR MULTI-TASK NEURAL PROCESSES

We provide a derivation of ELBO of the proposed multi-task processed with hierarchical context modeling. The likelihood of multi-task learning is as follows:

$$
\begin{aligned}
p(\{\mathbf{Y}_l^*\}|\{\mathbf{X}_l^*\},\{D_l\}) &= \prod_{l=1}^{L} p(\mathbf{Y}_l^*|\mathbf{X}_l^*,\{D_l\}) \\
&= \prod_{l=1}^{L} \int\int p(\mathbf{Y}_l^*|\mathbf{X}_l^*,\boldsymbol{\psi}_l) p_{\theta_1}(\boldsymbol{\psi}_l|\boldsymbol{\alpha}_l,\mathbf{M}) p_{\theta_2}(\boldsymbol{\alpha}_l|D_l) d\boldsymbol{\alpha}_l d\boldsymbol{\psi}_l
\end{aligned}
\tag{14}
$$

Based on conditional independence assumption, we introduce the variational joint posterior distribution factorized as (9). By incorporating the variational posteriors in the log likelihood, we can obtain the ELBO as follows:

$$
\begin{aligned}
&\log p(\{\mathbf{Y}_l^*\}|\{\mathbf{X}_l^*\},\{D_l\}) \\
&\quad= \sum_{l=1}^{L} \log \int\int p(\mathbf{Y}_l^*|\mathbf{X}_l^*,\boldsymbol{\psi}_l) p_{\theta_1}(\boldsymbol{\psi}_l|\boldsymbol{\alpha}_l,\mathbf{M}) p_{\theta_2}(\boldsymbol{\alpha}_l|D_l) d\boldsymbol{\alpha}_l d\boldsymbol{\psi}_l \\
&\quad= \sum_{l=1}^{L} \log \int \{ \int p(\mathbf{Y}_l^*|\mathbf{X}_l^*,\boldsymbol{\psi}_l) p_{\theta_1}(\boldsymbol{\psi}_l|\boldsymbol{\alpha}_l,\mathbf{M}) \frac{q_{\varphi_1}(\boldsymbol{\psi}_l|\mathcal{D}_l^*)}{q_{\varphi_1}(\boldsymbol{\psi}_l|\mathcal{D}_l^*)} d\boldsymbol{\psi}_l \} p_{\theta_2}(\boldsymbol{\alpha}_l|D_l) \frac{q_{\varphi_2}(\boldsymbol{\alpha}_l|\mathcal{D}_l^*)}{q_{\varphi_2}(\boldsymbol{\alpha}_l|\mathcal{D}_l^*)} d\boldsymbol{\alpha}_l \\
&\quad\geq \sum_{l=1}^{L} \Big\{ \mathbb{E}_{q_{\varphi_2}(\boldsymbol{\alpha}_l|\mathcal{D}_l^*)} \{ \int p(\mathbf{Y}_l^*|\mathbf{X}_l^*,\boldsymbol{\psi}_l) p_{\theta_1}(\boldsymbol{\psi}_l|\boldsymbol{\alpha}_l,\mathbf{M}) \frac{q_{\varphi_1}(\boldsymbol{\psi}_l|\mathcal{D}_l^*)}{q_{\varphi_1}(\boldsymbol{\psi}_l|\mathcal{D}_l^*)} d\boldsymbol{\psi}_l \} \\
&\qquad\qquad - \mathbb{D}_{\mathrm{KL}}[q_{\varphi_2}(\boldsymbol{\alpha}_l|\mathcal{D}_l^*)||p_{\theta_2}(\boldsymbol{\alpha}_l|D_l)] \Big\} \\
&\quad\geq \sum_{l=1}^{L} \Big\{ \mathbb{E}_{q_{\varphi_2}(\boldsymbol{\alpha}_l|\mathcal{D}_l^*)} \{ \mathbb{E}_{q_{\varphi_1}(\boldsymbol{\psi}_l|\mathcal{D}_l^*)} [p(\mathbf{Y}_l^*|\mathbf{X}_l^*,\boldsymbol{\psi}_l)] - \mathbb{D}_{\mathrm{KL}}[q_{\varphi_1}(\boldsymbol{\psi}_l|\mathcal{D}_l^*)||p_{\theta_1}(\boldsymbol{\psi}_l|\boldsymbol{\alpha}_l,\mathbf{M})] \} \\
&\qquad\qquad - \mathbb{D}_{\mathrm{KL}}[q_{\varphi_2}(\boldsymbol{\alpha}_l|\mathcal{D}_l^*)||p_{\theta_2}(\boldsymbol{\alpha}_l|D_l)] \Big\}.
\end{aligned}
\tag{15}
$$

B MORE EXPERIMENTAL DETAILS

Details of iteration numbers and batch sizes for different benchmarks are provided in Table 6. In each batch, the number of training samples from each task and category is identical. We train all models and parameters by the Adam optimizer Kingma & Ba (2014) using an NVIDIA Tesla V100 GPU. The learning rate is initially set as $1e-4$ and decreases with a factor of 0.5 every $3,000$ iterations. The network architectures of the proposed multi-task neural processes for multi-task classification are provided as follows.

Table 6: The iteration numbers and batch sizes on different datasets, where C and L denotes the number of classes and tasks in the specific dataset, respectively.

Dataset	**Iteration**	**Batch size**
`Office-Home`	$15,000$	$8*C*L$
`Office-Caltech`	$15,000$	$8*C*L$
`ImageCLEF`	$15,000$	$8*C*L$

Table 7: The architecture of inference networks $\varphi_1(\cdot)$ for latent variable $\boldsymbol{\psi}$ of multi-task neural processes.

Output size	**Layers**
4096	Input feature
4096	Dropout (p=0.7)
4096	Fully connected, ELU
4096	Fully connected, ELU
4096	Reparameterization to μ_ψ, σ_ψ^2

Table 8: The architecture of inference networks $\varphi_2(\cdot)$ for the latent variable $\boldsymbol{\alpha}$ of multi-task neural processes.

Output size	**Layers**
4096	Input feature
4096	Dropout (p=0.7)
2048	Fully connected, ELU
2048	Fully connected, ELU
2048	Reparameterization to μ_α, σ_α^2

Table 9: The architecture of the neural network $h(\cdot)$ of multi-task neural processes.

Output size	**Layers**
2048	Input feature
1024	Fully connected, ELU
512	Fully connected, ELU
L	Fully connected
L	Normalization

The architecture of the inference network φ_1 is provided in Table 7. The architecture of the inference network φ_2 is provided in Table 8. We note that the inference network θ_1 and θ_2 only share the same architectures with φ_1 and φ_2, respectively. They are different inference networks without parameter sharing. The architecture of the neural network h is provided in Table 9. During inference, we apply the reparameterization trick to generate the samples for the latent variables (Kingma & Welling, 2013).

C More Experimental Results

We provide experiments results on the three multi-task classification datasets with 20% training samples in Table 10, 11 and 12. The proposed multi-task neural processes consistently achieve the best performance on all three benchmarks.

Table 10: Performance comparison of different methods on the `Office-Home` dataset with 20% training samples.

Methods	A	C	P	R	Avg.
Single task learning	54.6±0.4	50.6±0.4	81.3±0.2	73.1±0.3	64.9±0.1
Bakker & Heskes (2003)	61.3±0.2	56.5±0.2	81.7±0.3	75.4±0.2	68.7±0.2
Long et al. (2017)	65.1±0.3	46.7±0.2	79.9±0.3	76.6±0.3	67.1±0.1
Kendall et al. (2018)	59.5±0.3	53.8±0.3	80.1±0.1	73.6±0.4	66.8±0.2
Qian et al. (2020)	58.3±0.2	53.5±0.3	79.8±0.2	73.1±0.3	66.2±0.1
Multi-task neural processes	64.2±0.1	55.7±0.3	82.6±0.2	77.2±0.3	**69.9±0.3**

Table 11: Performance comparison of different methods on the `Office-Caltech` dataset with 20% training samples.

Methods	A	W	D	C	Avg.
Single task learning	94.9±0.2	92.8±0.4	95.2±0.6	86.7±0.6	92.4±0.3
Bakker & Heskes (2003)	95.2±0.2	94.4±0.4	99.5±0.3	91.3±0.1	95.1±0.1
Long et al. (2017)	95.5±0.3	94.9±0.1	99.2±0.3	91.0±0.4	95.1±0.1
Kendall et al. (2018)	95.4±0.7	93.2±0.4	99.2±0.4	91.2±0.3	94.7±0.3
Qian et al. (2020)	95.7±0.4	94.1±0.2	99.2±0.5	91.1±0.4	95.0±0.2
Multi-task neural processes	94.9±0.3	96.6±0.2	99.2±0.4	92.3±0.2	**95.7±0.1**

Table 12: Performance comparison of different methods on the `ImageCLEF` dataset with 20% training samples.

Methods	C	I	P	B	Avg.
Single task learning	92.9±0.6	84.6±0.3	72.5±0.4	54.6±0.6	76.2±0.3
Bakker & Heskes (2003)	94.4±0.5	90.6±0.4	74.2±0.4	57.9±0.3	79.3±0.4
Long et al. (2017)	94.4±0.4	89.2±0.5	75.8±0.5	59.4±0.3	79.7±0.3
Kendall et al. (2018)	93.3±0.4	91.0±0.2	75.6±0.2	56.9±0.4	79.2±0.3
Qian et al. (2020)	93.1±0.3	92.1±0.5	74.4±0.7	55.8±0.6	78.9±0.5
Multi-task neural processes	92.1±0.4	91.5±0.5	79.2±0.4	60.6±0.3	**80.8±0.1**