

Quantifying Predictive Uncertainty in Medical Image Analysis with Deep Kernel Learning

1st Zhiliang Wu
Siemens AG
LMU Munich
Munich, Germany
zhiliang.wu@siemens.com

2nd Yinchong Yang
Siemens AG
Munich, Germany
yinchong.yang@siemens.com

3rd Jindong Gu
Siemens AG
LMU Munich
Munich, Germany
jindong.gu@siemens.com

4th Volker Tresp
Siemens AG
LMU Munich
Munich, Germany
volker.tresp@siemens.com

Abstract—Deep neural networks are increasingly being used for the analysis of medical images. However, most works neglect the uncertainty in the model’s prediction. We propose an uncertainty-aware deep kernel learning model which permits the estimation of the uncertainty in the prediction by a pipeline of a Convolutional Neural Network and a sparse Gaussian Process. Furthermore, we adapt different pre-training methods to investigate their impacts on the proposed model. We apply our approach to Bone Age Prediction and Lesion Localization. In most cases, the proposed model shows better performance compared to common architectures. More importantly, our model expresses systematically higher confidence in more accurate predictions and less confidence in less accurate ones. Our model can also be used to detect challenging and controversial test samples. Compared to related methods such as Monte-Carlo Dropout, our approach derives the uncertainty information in a purely analytical fashion and is thus computationally more efficient.

Index Terms—uncertainty quantification, medical imaging, sparse Gaussian Process approximation, deep Convolutional Neural Networks,

I. INTRODUCTION

Various machine learning methods have been developed to support patient care to deal with the exploding amount of healthcare data [1], [2]. An important example is medical imaging. In classical image analysis, the standard machine learning methods make predictions based on sophisticated handcrafted features extracted from the medical images. With the introduction of deep neural networks (DNNs), especially Convolutional Neural Networks (CNNs), manual feature engineering is replaced by self-organized supervised representation learning.

Although DNNs now define the state-of-the-art in many applications, an often encountered problem is that they fail to provide reasonable confidence estimates for their predictions [3]. In a classification task, this can result in over-confident predictions for misclassified samples [4]. This observation has encouraged the development of various calibration methods such as temperature scaling [5] and isotonic regression [6]. In a regression task, however, the modeling of input-dependent predictive uncertainty is rarely considered. Whereas in classification, a well-calibrated probabilistic prediction can sometimes be used to derive confidence values, in regression one needs

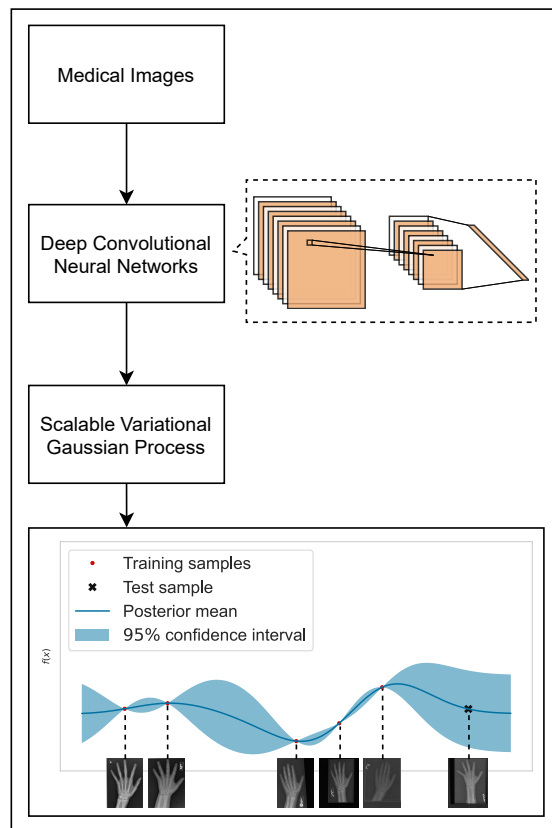


Fig. 1. An illustration of our proposed model: Combining a deep Convolutional Neural Network with the Scalable Variational Gaussian Process. Latent features of a medical image are extracted by the Convolutional Neural Network and then consumed by the Scalable Variational Gaussian Process. The proposed model outputs a predictive distribution of the Gaussian Process posterior, which can be interpreted as a mean estimate and a predictive variance. For instance, with a localized kernel the prediction of a rare test sample (the rightmost image) with few similar training samples demonstrates a larger variance.

to estimate confidence considering predictive distributions [7].

The problem of quantifying the predictive uncertainty in deep learning models limits their applicability to safety-critical domains such as healthcare [8], [9]. In most cases, existing clinical decision support systems that rely on deep learning

can only provide a point estimate, e.g., for a continuous severity score, progression-free survival time, or length-of-stay. Physicians, who are supposed to interpret the output of a DNN, face the challenge of not knowing how much they could trust the prediction. The goal of this paper is to provide a quantified uncertainty estimate for each prediction in a principled, mathematically sound manner. An unusually high uncertainty estimate would encourage the physician to investigate a case more closely since it is more likely to deviate from the “normal” ones. A high uncertainty typically means that there are few similar cases in the training data.

In a frequentist analysis of linear regression models, the predicted variance is derived from that of the model parameters; due to its high-dimensional nonlinear nature, this cannot easily be applied to DNNs. Currently, leading approaches to reason on the uncertainty in DNNs include MC dropout methods [10], [11], Bayesian neural networks [12], [13] and deep ensembles [14]. These methods could become computationally expensive when large and deep neural networks are necessary. In this paper, we explore another direction to quantify predictive uncertainty with Gaussian Processes (GPs), a well-known class of Bayesian machine learning methods [15]. A GP implicitly ties the predictive uncertainty with the similarity between samples, which does not model ensembles and can produce a predictive distribution with only one forward pass.

More specifically, when localized kernels are being used, e.g., Radial Basis Function (RBF) kernels or the more general Matérn kernels, a GP model would be confident in its mean estimate if there have been training samples observed in the “neighborhood” of the test input. Otherwise, the model would tend to output high variances for the predictions. Thus, for RBF-kernels, the Euclidean distance of inputs must be meaningful in the application, which often is not the case in high-dimensional problems, a typical example being raw images described by their pixel values.

Another known challenge in GP is the fact that computational complexity scales as $\mathcal{O}(n^3)$ and storage complexity as $\mathcal{O}(n^2)$, where n denotes the number of data samples [15]. In recent years, significant progress has been made to address these scalability issues [16], which has motivated work on combining DNNs with GPs, a.k.a. *Deep Kernel Learning* [17]. The pipeline architecture we propose in this paper is shown in Fig. 1, where we apply a state-of-the-art sparse GP on top of a CNN for predictions on medical images.

Our contribution can be summarized as follows:

- We present a novel deep kernel learning model for regression on medical images based on the latest developments in sparse GPs and CNNs.
- We enhance the proposed model by introducing different pre-training methods for the CNNs and initialization methods to optimize the inducing points in sparse GPs.
- We apply the proposed model to the tasks of univariate bone age prediction and multivariate lesion localization, and provide a thorough comparison of different pre-training and initialization methods in terms of both point estimate and predictive uncertainty.

II. RELATED WORK

Deep Convolutional Neural Networks for Medical Image Analysis The analysis of medical images is arguably one of the areas where deep learning methods have been demonstrating the most promising performances, including diagnostics, dermatology, radiology, ophthalmology, and pathology [18]. Deep learning-based solutions can offer physicians second opinions by, e.g., annotating the regions of interest. More specifically, CNN-based solutions have achieved physician-level accuracy, e.g., with CheXNet [19] for pneumonia detection, which is a 121-layer Dense Convolutional Network (DenseNet) [20] trained on the ChestX-ray 14 dataset [21]. One factor limiting the progress is the relatively small size of labeled datasets available for specific clinical tasks when compared to large visual databases on nonmedical images, like the ImageNet dataset [22]. Therefore, methods like transfer learning are commonly used to take advantage of models trained on more or less unrelated datasets. However, many of these solutions focus only on improving the point estimate performance, ignoring the importance of the predictions’ uncertainty. In this work, we focus on addressing the problem of providing meaningful uncertainty estimates.

Scalable Variational Gaussian Process with Neural Networks Efforts from earlier times include the Bayesian committee machine [23], Nyström methods [24], [25], the Fully Independent Training Conditional (FITC) Approximation [26], and Variational Free Energy (VFE) [27]. Recently, [28] proposed the Scalable Variational Gaussian Process (SVGP), which reduces the computational complexity to $\mathcal{O}(m^3)$, where m denotes the number of *inducing points* (more details see Sec. III-A). In addition, SVGP enables the training with stochastic gradient descent (SGD)-based methods. Afterward, [29] combined SVGP with DNNs for classification tasks. The approach is called Gaussian Process hybrid deep networks (GPDNN). [17] combined neural networks with a KISS-GP covariance matrix, which takes advantage of a sparse matrix with inducing points lying on some grid structure [30]. The proposed method is called Deep Kernel Learning (DKL). More recently, [31] proposed the Parametric Predictive Gaussian Process (PPGP) regressor to improve the predictive variances in SVGP-based models, which shows promising performance in various applications. Inspired by the idea of DKL, we propose in this paper a model to train a state-of-the-art sparse GP model with deep CNNs in a more cohesive way.

Pre-training Techniques for Deep Convolutional Neural networks Pre-training techniques have been developed for Deep Belief Networks [32] and stacked auto-encoders [33], where unsupervised pre-training was used for initialization, followed by supervised fine-tuning. Meanwhile, transfer learning techniques received increasing attention due to their ability to derive good representations for instances also from domains not considered in training [34]. After the introduction of the ImageNet challenge [22], it has been common practice to pre-train models on the ImageNet dataset as an initialization for other downstream tasks. More recently, self-supervised

learning methods, e.g., contrastive learning [35], have gained much attention as a powerful learning paradigm, which bridges the performance gap between supervised learning methods and unsupervised ones significantly. From a pre-training perspective, self-supervised learning can be considered as an example in deep metric learning (DML). The goal of DML is to map data to a latent space where data points with similar labels are located close together, and data with dissimilar labels are far apart [36]. In this paper, we adapt different pre-training methods under the setting of DKL.

III. METHOD

In this section, we provide a detailed introduction to our method. Our proposed model consists of two consecutive parts: a trainable feature extractor based on deep CNNs and an uncertainty-aware prediction model in the form of sparse GPs. The feature extractor is also commonly known as the *backbone*, because it refers to the parts of the network excluding the final classification layers in, e.g., DenseNets [20] or ResNets [37]. The output of such backbones, namely the feature maps, a.k.a. latent representations of the raw input, serves as input to the predictive GP regression. In the following, we first discuss how sparse GP models can scale to large datasets. Afterward, we introduce our initialization and pre-training techniques. Finally, we summarize the complete algorithm from the network initialization to the GP fine-tuning.

Notations: We denote the training dataset as $\{\mathbf{X}_i, y_i\}_{i=1}^n$, where $\mathbf{X}_i \in \mathbb{R}^{n_H \times n_W \times n_C}$ is an image of size $n_H \times n_W$ with n_C color dimensions, $y_i \in \mathbb{R}$ is the target variable in a univariate regression task, and n is the number of data samples. With the CNN backbones, we extract a latent representation from \mathbf{X}_i and denote it as \mathbf{h}_i .

A. Scalable Variational Gaussian Processes as Output Layers

A Gaussian process (GP) is a collection of random variables, any finite number of which have a joint zero-mean Gaussian distribution [15]. Formally, if we denote all target variables y_i in the column vector as $\mathbf{y} \in \mathbb{R}^n$ in a univariate regression problem, it follows

$$\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \mathbf{K} + \sigma_{\text{obs}}^2 \mathbf{I}),$$

where the covariance matrix $\mathbf{K} \in \mathbb{R}^{n \times n}$ is parametrized by the respective inputs as

$$k_{ij} := k(\mathbf{h}_i, \mathbf{h}_j)$$

and σ_{obs}^2 is the noise variance. Note that in a standard setup of GP, the input to the kernel function $k(\cdot, \cdot)$ is a pair of feature vectors. In the scope of our work, we feed the latent representations generated by CNN backbones to the kernel function.

To find optimal hyperparameters in the kernel function (e.g., the scaling parameter in an RBF-kernel), the training of the GP involves maximizing the log marginal likelihood

$$\mathcal{L}_{\text{GP}} = -\frac{1}{2} \mathbf{y}^\top (\mathbf{K} + \sigma_{\text{obs}}^2 \mathbf{I})^{-1} \mathbf{y} - \frac{1}{2} \log |\mathbf{K} + \sigma_{\text{obs}}^2 \mathbf{I}| - \frac{n}{2} \log 2\pi.$$

Given a new input sample \mathbf{h}_* , the GP model provides a predictive distribution as

$$f_* \sim \mathcal{N}(\mathbf{k}_*^\top (\mathbf{K} + \sigma_{\text{obs}}^2 \mathbf{I})^{-1} \mathbf{y}, \mathbf{k}_{**} - \mathbf{k}_*^\top (\mathbf{K} + \sigma_{\text{obs}}^2 \mathbf{I})^{-1} \mathbf{k}_*),$$

where $\mathbf{k}_* = [k(\mathbf{h}_1, \mathbf{h}_*), \dots, k(\mathbf{h}_n, \mathbf{h}_*)]^\top \in \mathbb{R}^n$.

However, the complexity from the inverse operation of the large matrices in \mathcal{L}_{GP} and f_* hinders the application of GP models to large-scale datasets, which was the motivation for works on different approximation methods.

The key idea of [26]–[28] is to learn a number of so-called inducing points by variational methods, which can be viewed as a learnable pseudo dataset $\{\mathbf{z}_i, u_i\}_{i=1}^m =: (\mathbf{Z}, \mathbf{u})$ to summarize the original large dataset, where $m \ll n$. The approximation follows these steps: 1) The original dataset is augmented with the inducing points; 2) Based on different assumptions, the log marginal likelihood $\log p(\mathbf{y})$ is approximated as a function only of inducing points; 3) Optimization is done by maximizing either the approximated log marginal likelihood [26], or the lower bound of it, which is also known as the Evidence Lower Bound (ELBO) [27], [28]; 4) The predictions for new samples are based on the optimized inducing points instead of the original dataset. Among all approximation methods, SVGP turns out to be the most popular one, possibly thanks to its largely reduced computational and storage complexity as well as the natural integration of SGD-based methods [28], [38]. Therefore, we take advantage of SVGP as one of the sparse GP models in this paper.

In SVGP [28], a multivariate Normal distribution $\mathcal{N}(\mathbf{m}, \mathbf{S})$ is introduced to the variational distribution $q(\mathbf{u})$. [31] maximizes the ELBO

$$\mathcal{L}_{\text{SVGP}} = \sum_{i=1}^n \left\{ \log \mathcal{N}(y_i | \mu_f(\mathbf{h}_i), \sigma_{\text{obs}}^2) - \frac{\sigma_f^2(\mathbf{h}_i)}{2\sigma_{\text{obs}}^2} \right\} - \text{KL}(q(\mathbf{u}) \| p(\mathbf{u})), \quad (1)$$

where we have the predictive mean $\mu_f(\mathbf{h}_i) = \mathbf{k}_i^\top \mathbf{K}_{uu}^{-1} \mathbf{m}$, the predictive variance $\sigma_f^2(\mathbf{h}_i) = k_{ii} - \mathbf{k}_i^\top \mathbf{K}_{uu}^{-1} \mathbf{k}_i + \mathbf{k}_i^\top \mathbf{K}_{uu}^{-1} \mathbf{S} \mathbf{K}_{uu}^{-1} \mathbf{k}_i$, $p(\mathbf{u}) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{uu})$, $\mathbf{k}_i \in \mathbb{R}^m$, $\mathbf{K}_{uu} \in \mathbb{R}^{m \times m}$, and $\text{KL}(\cdot \| \cdot)$ denotes the Kullback–Leibler divergence between two distributions. We use Θ to denote all trainable parameters and adapt them with SGD-based methods, including \mathbf{m}, \mathbf{S} for the variational distribution $q(\mathbf{u})$, inducing points \mathbf{Z}, \mathbf{u} for the covariance matrices like \mathbf{K}_{uu} or \mathbf{k}_i , σ_{obs} in the likelihood model and various hyperparameters in the kernel function, e.g., length scale in an RBF kernel.

[31] points out that the predictive distribution in SVGP tends to be dominated by the observational noise and underestimates the input-dependent uncertainty. As a solution, they proposed the PPGP Regressor, which takes advantage of the formulation of the predictive distributions in SVGP but restores the symmetry of the function variance $\mu_f(\mathbf{h}_i)$ in the training objective through the maximum likelihood estimation (MLE) methods. Formally, the objective in PPGP is

$$\mathcal{L}_{\text{PPGP}} = \sum_{i=1}^n \log \mathcal{N}(y_i | \mu_f(\mathbf{h}_i), \sigma_{\text{obs}}^2 + \sigma_f^2(\mathbf{h}_i)) - \text{KL}(q(\mathbf{u}) \| p(\mathbf{u})). \quad (2)$$

In our experiments, we report results of both SVGP and PPGP methods, which only differ in their respective ELBO objectives.

Although the scalability problem in large datasets is nicely addressed in the SVGP-based models, the commonly used GP kernels, such as RBF and Matérn, cannot directly handle high dimensional data such as images. Therefore, there are many efforts to combine the inductive bias in neural networks and the non-parametric nature of GP-based models, including GPDNN [29] and DKL [17]. In [29] and [17], the training is initiated with a standard neural network with a linear predictive model fit on the target variable. Afterward, the linear model is replaced with a GP to enable uncertainty-aware prediction. In our experiments with these approaches, we do not observe performance improvement in terms of point estimates. Therefore, we explore other pre-training methods that do not directly require the target variables, including Convolutional Autoencoders and Deep Metric Learning.

Fig. 2 illustrates the basic idea of integrating DKL in our proposed model. The image sample \mathbf{X}_i is embedded in some latent space defined by the backbone as \mathbf{h}_i . The SVGP-based model then consumes \mathbf{h}_i as the input to produce the predictive distribution $\mathcal{N}(\mu_i, \sigma_i^2)$ for the target variable y_i . In other words, we propose to use SVGP-based models as output layers to replace the final linear layers found in common CNN architectures. The trainable parameters Φ in the backbone and Θ in the SVGP-based output layers are optimized together w.r.t. the ELBO objective.

Based on our observations, we realized that two modifications turn out to be critical for training the proposed model. First, we find it necessary for the model architecture to add one more linear layer after the backbone to *further* reduce the dimension of the latent space. In ResNet18 and DenseNet121, the dimensions of the extracted latent spaces are 512 and 1024, respectively. These dimensions prove to be too large for RBF kernel functions that rely on l^2 norm, presumably due to the fact that in high dimensional space, the Euclidean norm becomes irrelevant as a distance measure [39], [40]. With a thorough hyper-parameter search, we find that a dimension reduction to 50 always shows a stable performance.

Second, the initialization of the inducing points plays an important role. If the inducing inputs are initialized from

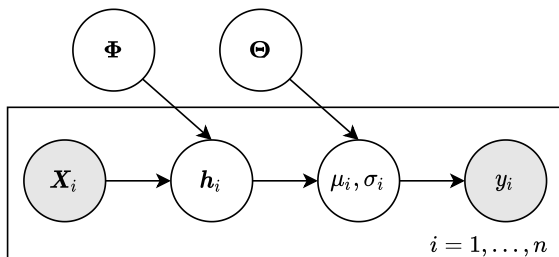


Fig. 2. Graphical model of deep kernel learning model in plate notation. Nodes are variables where shaded ones are observed, and non-shaded ones are latent variables. Plates indicate the repetition of the subgraph.

random vectors, the training never converges to meaningful results in our experiments. One explanation would be that with purely random initialization of inducing points, the covariance function value between each pair of samples is also random since it is defined via all inducing points (cf. Equation (1) in [27]). To this end, the GP has no chance of modeling the target based on these random distances with a multivariate Gaussian. As a solution, we initialize inducing inputs by the latent representations produced by the backbone from the image samples. Formally, we initialize the inducing inputs by

$$\mathbf{z}_i^{\text{init}} = \mathbf{h}_i = f_{\Phi}(\mathbf{X}_i),$$

where the parameters Φ in the backbone can be initialized from scratch, transferred from other models, or pre-trained in auxiliary tasks. Such a procedure is similar to using a subset of the dataset as the initial inducing inputs in a vanilla SVGP model, where raw features are fed to the model directly.

So far, we focused on models for univariate regression problems. For the multivariate case, we propose to use the same backbone to generate the latent representations but feed it as input to multiple independent SVGP-based models. The number of involved SVGPs equals the dimension of the target variables.¹ For the cases where there are correlations between the target variables, more advanced methods like Linear Model of Coregionalization (LMC) can be used [41], which we leave to our future work.

B. Pre-training Convolutional Neural Networks

In our proposed architecture, the sparse GP model is defined in a latent space learned by the CNN backbone. The optimization task is correspondingly twofold: the GP is supposed to learn the parameters like the inducing points, and the CNN backbone should adapt its parameters to generate representative latent features. However, in the early phase of training, the CNN backbone may not have learned to extract representative features. In other words, the training samples could be mapped somewhat randomly in the latent space. This could pose a challenging task for the downstream GP model such that – based on our observation of the experiments – its parameters might converge to unfavorable values that are difficult to correct later on. To address this issue, we anticipate that the training quality could be improved by two strategies:

- Initialization with transfer learning in Sec. III-B1
- Pre-training with auxiliary tasks:
 - Convolutional Autoencoder in Sec. III-B2
 - Deep Metric Learning in Sec. III-B3

1) *Transfer learning*: We regard transfer learning as reusing the knowledge from the models trained on different datasets. In the simplest case, we consider reusing CNN layers that have been trained on the classification task on the ImageNet dataset. It has been shown that in a CNN architecture, the early layers that are close to the input can learn to extract generic, low-level features that may apply across different types of image

¹This is also the configuration for the model with (multivariate) linear layers, which facilitates a fair comparison in experiments.

data [34], [42]. These generic features typically include edges, basic patterns, and color gradients. We anticipate that such an initialization of the CNN would produce a latent space where the distance between samples better represents the distance in the original feature space, thus providing an advantageous starting point of learning the GP kernel. In the following, the pre-training methods are adapted to be used in settings either with or without transfer learning. With the proposed adaptations, we would be able to investigate the effectiveness of various components through experiments.

2) *Convolutional Autoencoder*: The autoencoder (AE) is a well-known unsupervised representation learning method for dimensionality reduction. It consists of an encoder and a decoder. The encoder maps the inputs to some lower-dimensional latent space, whereas the decoder reconstructs the inputs from the latent representations, which ensures that the encoder has learned the most relevant features. Convolutional Autoencoders (CAEs) are a special case of AEs in that the convolutional filters are reused among different locations of the input to preserve the spatial locality [43].

Normally, CAEs have several convolutional layers in the encoder and transposed convolutional layers in the decoder. To enable transfer learning in CAEs, we propose to use the CNN backbone as the encoder. And we construct a symmetric decoder using transposed convolutional layers. In such a way, the decoder can have a similar model capacity to that of the encoder. Formally, after getting the latent representations \mathbf{h}_i from the encoder $f_{\Phi}(\cdot)$, we feed it into the decoder $g_{\Psi}(\cdot)$ to reconstruct the original images

$$g_{\Psi} : \mathbb{R}^h \rightarrow \mathbb{R}^{n_H \times n_W \times n_C}$$

$$\mathbf{h}_i \mapsto g_{\Psi}(\mathbf{h}_i) =: \hat{\mathbf{X}}_i,$$

where Ψ denotes the trainable parameters in the decoder network. The parameters of the encoder Φ can be initialized from scratch or from models using transfer learning. The training of the CAE involves minimizing the Mean Squared Error (MSE) between the original image sample \mathbf{X}_i and the reconstructed image $\hat{\mathbf{X}}_i$. Formally, the loss function is defined as

$$\mathcal{J}_{\text{CAE}} = \frac{1}{n} \sum_{i=1}^n \|\mathbf{X}_i - \hat{\mathbf{X}}_i\|_2^2, \quad (3)$$

where $\|\cdot\|_2$ denotes the Euclidean norm.

3) *Deep Metric Learning*: Taking advantage of the structure of twin neural networks (replications of the same NN), CNN backbones are applied in DML for learning the latent representations so that samples with similar labels would be mapped closer to each other in the latent space. The latent representations from the trained backbones turn out to be effective for tasks like face verification [44], [45] or person re-identification [46].

Given an image sample \mathbf{X}_i from the dataset, the backbone defines a function $f_{\Phi}(\cdot)$ to embed it in some latent space.

Formally, we have

$$f_{\Phi} : \mathbb{R}^{n_H \times n_W \times n_C} \rightarrow \mathbb{R}^h$$

$$\mathbf{X}_i \mapsto f_{\Phi}(\mathbf{X}_i) =: \mathbf{h}_i,$$

where Φ denotes the trainable parameters in the network and h is the dimension of the latent space. The trainable parameters Φ can be either initialized from scratch or be transferred from models trained on other large-scaled datasets, e.g., the ImageNet dataset.

With the class label information, a triplet is defined to consist of an anchor sample \mathbf{X}_i^A , a positive sample \mathbf{X}_i^P , and a negative sample \mathbf{X}_i^N , where the anchor is of the same class as the positive and the negative is not. However, in a regression problem, the target variables cannot define the triplets directly since they are continuous values. To mitigate this, we propose to categorize the target variables into classes to generate triplets in DML. It can also be viewed as a coarse pre-training step before the final fine-tuning from a learning perspective. Concretely speaking, we categorize target variables according to their binning in the histogram for univariate regression tasks and apply K-means clustering to find a class label for the target variables in multivariate regression tasks.

During training, minimizing the triplet margin loss makes the anchor-positive distance smaller than the anchor-negative distances by a certain margin [45]. Formally, the loss function is defined as

$$\mathcal{J}_{\text{triplet}} = \sum_{i=1}^n [d(\mathbf{h}_i^A, \mathbf{h}_i^P) - d(\mathbf{h}_i^A, \mathbf{h}_i^N) + \alpha]_+, \quad (4)$$

where $d(\cdot, \cdot)$ is a distance metric, e.g., the Euclidean distance, α is the value of a pre-defined margin, and $[\cdot]_+$ only takes the positive part of the variable. Also, triplet selection is an important step to get fast convergence of the training since the network only gets gradients from the triplets having positive values in Equation (4). Within each mini-batch, we pick negative samples whose distance to the anchor is larger than the anchor-positive distance (within a margin of α). That means we have

$$0 < d(\mathbf{h}_i^A, \mathbf{h}_i^N) - d(\mathbf{h}_i^A, \mathbf{h}_i^P) < \alpha,$$

which are regarded as *semi-hard* examples in [45].

To find an appropriate number of epochs for the pre-training, we take advantage of early stopping methods, which terminate the training automatically by monitoring specific metrics derived from the validation set. Here, we use the metric Mean Average Precision at R (MAP@R), a more informative evaluation metric since it combines the ideas of Mean Average Precision and R-precision [36].

We argue that the training objective of the DML agrees with the paradigm of a GP regression using localized kernels, which is to interconnect the similarity of data samples in the target space to the similarity in their input spaces. Since GP cannot directly operate in the raw pixel space, a mapping function that preserves the similarity from the target space to the latent space would provide the GP with an ideal input space.

C. End-to-end Fine-tuning Deep Kernel Learning

In this section, we elaborate our proposed method in Algorithm 1 by inversely joining the modules that have been introduced in the last two sections.

Algorithm 1: Fine-tuning Deep Kernel Learning

Input: An image dataset of the form $\{\mathbf{X}_i, y_i\}_{i=1}^n$.
Output: A fine-tuned DKL model for regression

```

1 if Transfer is True then
2   |  $\Phi \leftarrow \Phi^{\text{ImageNet}}$ 
3 end
4 switch Pre-training is DML do
5   | Generate triplets  $\{\mathbf{X}_i^A, \mathbf{X}_i^P, \mathbf{X}_i^N\}$ 
6   |  $\Phi \leftarrow \arg \min_{\Phi} \mathcal{J}_{\text{triplet}}(\Phi)$ 
7 end
8 switch Pre-training is CAE do
9   |  $\Phi, \Psi \leftarrow \arg \min_{\Phi, \Psi} \mathcal{J}_{\text{CAE}}(\Phi, \Psi)$ 
10 end
11 Initialize the inducing points  $\{\mathbf{Z} | z_i^{\text{init}} = f_{\Phi}(\mathbf{X}_i)\}$ 
12  $\Phi, \Theta \leftarrow \arg \max_{\Phi, \Theta} \mathcal{L}_{\text{PPGP}}(\Phi, \Theta)$ 
13 return  $\Phi, \Theta$ 

```

Depending on whether we want to reuse the model trained on the ImageNet dataset, we will initialize the parameters in the backbones from the transferred model or from scratch (line 2). If we use DML as pre-training, we first categorize the target variables to generate triplets with the class information (line 5). Then the backbones are trained with triplet margin loss in Equation (4) (line 6). On the other hand, if we use CAE as pre-training, the parameters in the encoder and decoder are trained jointly against the CAE loss in Equation (3) (line 9), where the encoder will be used as the backbone in later steps. After the pre-training, the backbone is used to initialize the inducing points with a subset of the image samples (line 11). Therefore, it is worth highlighting that the pre-training step affects the parameters in the neural network and the parameters in the SVGP-based output layer. Finally, the fine-tuning step is done w.r.t. the respective ELBO objective, where we use the PPGP objective in Equation (2) as an example (line 12).

IV. EXPERIMENTS

A. Datasets and Implementation Details

We have conducted experiments with two different datasets to validate our proposed methods. As an example for univariate regression tasks, we included the Bone Age Prediction (BAP) task from the Radiological Society of North America Pediatric Bone Age Machine Learning Challenge [47], [48]. In this dataset, there are 14,236 hand radiographs, where the target variable is defined as the bone age of pediatric patients under five years old. In addition, we included the lesion localization (LL) task from the DeepLesion dataset [49] as an example for the multivariate regression problem. In the original dataset, there are 32,120 axial computed tomography (CT) slices from 4,427 unique patients. Together with the tag information from LesaNet [50], we retrieve 7,310 slices for the lesion type

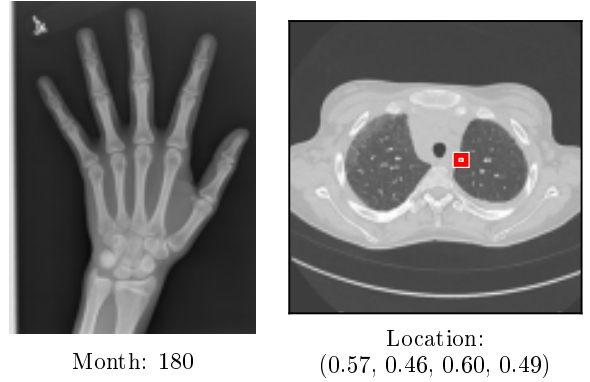


Fig. 3. An example for the task of Bone Age Prediction (left) and Lesion Localization (right).

of lung, where the task is to localize the lesion in a given CT image. The target is in the format of $(x_{1n}, y_{1n}, x_{2n}, y_{2n})$, where $x_{1n}, y_{1n}, x_{2n}, y_{2n}$ denote the normalized x -top-left, y -top-left, x -bottom-right, and y -bottom-right, respectively. Fig. 3 shows examples for the task of BAP and LL.

The CNN-related models are built using the *PyTorch* package [51], where the models with GP methods are implemented with the help of the *GPyTorch* package [52]. We conducted cross-validations (CV) for both tasks with 90% samples extracted in the datasets, where hyperparameters are tuned according to the performance on the validation set. The remaining unseen 10% samples constitute the test set, from which the results reported in the following sections are computed. Common data augmentations, including random crop, rotate, and horizontal flip, are applied. Due to the relatively small size of the datasets, the backbone of ResNet18 and DenseNet121 are chosen in all experiments. Related scripts² of the work will be published to improve the reproducibility.

B. Evaluation Approaches and Baselines

Due to the probabilistic nature of our proposed model, we considered two lines of evaluation approaches in the experiments: the performance of point estimates and the evaluation of predictive variances. We used the well-known Root Mean Squared Error (RMSE) to reflect the prediction performance for the former, where only the mean predictions of the proposed model are involved in the evaluation. For the latter, we included a novel method to validate the meaningfulness of the predictive uncertainty, namely a quantile performance (QP) plot. Intuitively speaking, a good uncertainty-aware model should demonstrate better performance together with higher confidence in its predictions and vice versa. Any uncertainty-aware regression model that produces point estimates and predictive variances can be evaluated against this criterion.

Given an uncertainty-aware model $f(\cdot)$ and its predictive distribution $f_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, we first sort all predictive vari-

²Related scripts see <https://github.com/ZhiliangWu/mDKL>.

ances in an ascending order $\{\sigma_i^2 \mid \sigma_i^2 \leq \sigma_{i+1}^2, i \in \{1, \dots, n\}\}$ and then compute K quantiles denoted as q_1, \dots, q_K .

Second, we compute the RMSE evaluation³ of the subset of predicted point estimates $\hat{y}_i := \mu_i$, whose paired variances σ_i^2 are smaller than or equal to each of the k -th quantile values of $k \in \{1, \dots, K\}$:

$$\text{Performance}_k := \text{RMSE}(\{(y_i, \hat{y}_i) \mid \forall \sigma_i^2 \leq k\text{-quantile}\}),$$

where (y_i, \hat{y}_i) denotes the evaluation pair. By plotting the performance value on the y -axis against the corresponding quantile value k on the x -axis, a monotonically increasing line is expected.

To study the point estimate performance of the SVGP-based output layers, models having the same backbone but with a linear layer, which is optimized directly w.r.t. MSE, are included as baselines. Besides, when investigating the effects of pre-training between various representation learning methods, the models without any pre-training serve naturally as baselines. For the performance of predictive variances, we included MC Dropout [10], a popular method for augmenting uncertainty in the neural networks, as a baseline. In MC Dropout, a dropout layer [53] is added before each layer in the network. In our experiments, we used the default dropout setting for DenseNet121 with a dropout rate of 0.2 during training and testing, whereas dropout layers with the same dropout rate are added after each of the four layers of residual blocks in ResNet18. The predicted value and predictive variances are computed by performing 50 stochastic forward passes through the network as suggested in [53].

C. Evaluation on the Bone Age Prediction

1) *Results on Point Estimates:* Tab. I demonstrates the performance of the proposed method with DenseNets121 on the univariate regression task, Bone Age Prediction. Our proposed models with SVGP-based output layers deliver competitive or, in most cases, even better performances compared to common architectures with linear layers. Overall, the proposed model with SVGP output layers using transfer learning and pre-trained with DML demonstrates the best performance. In addition, significantly superior performance is found on models using the parameters transferred from models trained on the ImageNet dataset, which holds under all pre-training variants. Comparing the models using transfer learning w.r.t. different pre-training methods (upper part in Tab. I), we see an improvement when the model is first pre-trained with DML, whereas the pre-training with CAE does not improve the performance. However, for the models without transfer learning (lower part in Tab. I), both DML and CAE enhance the performance of the models, whereas CAE shows a better pre-training performance than DML. As a reference, [48] reports 10.44 and 7.8 as RMSE values on 200 test samples from human reviewers and model predictions, respectively.

2) *Results on Predictive Variances:* We include the models with transfer learning but without any pre-training for the

TABLE I
BONE AGE PREDICTION WITH DENSENET121

Output Layer	Transfer Learning	RMSE (No pre-training)	RMSE (DML)	RMSE (CAE)
Linear*	Yes	12.118 ± 0.277	11.667 ± 0.231	14.076 ± 0.281
SVGP†	Yes	11.697 ± 0.102	11.440 ± 0.132	13.536 ± 0.279
PPGP†	Yes	11.679 ± 0.061	11.529 ± 0.089	13.694 ± 0.274
Linear*	No	19.934 ± 0.246	15.805 ± 0.157	15.340 ± 0.390
SVGP†	No	17.723 ± 0.298	15.832 ± 0.284	15.323 ± 0.411
PPGP†	No	18.341 ± 0.234	16.084 ± 0.336	15.752 ± 0.352

*Common architectures.

†With our proposed model.

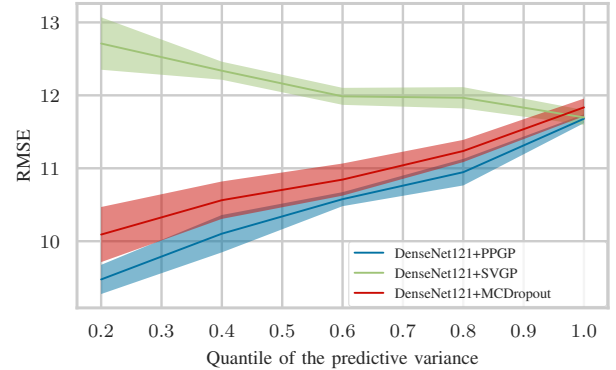


Fig. 4. Quantile Performance for the Bone Age Prediction with DenseNet121

QP plot in Fig. 5, where solid lines and error bars denote the means and standard deviations across different CV splits. A clear, monotonically increasing trend is observed in our proposed models with PPGP output layers and the models with MC Dropout. The line of PPGP is located underneath the MC Dropout, indicating better performance. In contrast, the models with SVGP output layers show a monotonically decreasing trend w.r.t. the quantile of the predictive variance. The models with PPGP output layers have an RMSE of 9.476 ± 0.200 (predictive variances at $q = 20\%$) for the samples they are more confident with, which is a relatively large improvement compared to the values reported in Tab. I.

3) *Discussion:* The superior performance of models with SVGP-based output layers is expected since the ELBO objective is a proxy for the log marginal likelihood objective, which is a generalization to MSE in linear regression. The improvements based on transfer learning conform to its popularity in the computer vision community, which validates the hypothesis that reusing the knowledge from large-scale datasets could help solve a new problem even with domain shift. For pre-training with DML, the models are first trained to embed samples with similar targets into nearby regions, which would be a helpful initialization for the inducing points and non-convex optimization in neural networks. Therefore, we observe a positive contribution from DML to the model performance. For the pre-training with CAE, the goal is to learn a *compressed* representation, which will be recovered in another decoder network. From the experimental results,

³This could be any metrics for evaluating point estimates.

such unsupervised representation learning would improve the performance if the model is trained from scratch but may deteriorate the knowledge transferred from large-scale datasets. It indicates a possibly higher correlation of the current task to the ImageNet classification than the compression task from CAE, which could be attributed to the large number of training samples in the ImageNet dataset. It is also worth mentioning that we also conducted the same experiments with the ResNet18 backbone, where similar results are observed. More details can be found in Appendix A and B.

What makes our proposed method appealing lies in the probabilistic nature of its prediction. The principled predictive variance from PPGP output layers is expected since 1) the inducing points technique facilitates explicit modeling of uncertainty, 2) the symmetric treatment of the predictive variance is restored in the training phase compared with SVGP. However, the good performance from MC Dropout also comes with a considerable computational cost. Roughly speaking, the inference time would be t times as much as our proposed model, where t is the number of stochastic forward passes for the inference. With $t = 50$, our experiment with 1424 test samples requires an inference time of 1777.04 seconds (almost half an hour) for the MC Dropout method, whereas our SVGP-based approach takes only 42.90 seconds. With more samples or more advanced backbone structures, the time cost will be more expensive for the MC Dropout method. These observations indeed motivate the application of our proposed models with PPGP output layers when meaningful predictive variances and low time complexity come to a higher priority in a real-world system.

D. Evaluation on the Lesion Localization

1) *Results on Point Estimates:* Tab. II shows the performance of our proposed method with DenseNet121 on the multivariate regression task, Lesion Localization. Similar to the task of BAP, our proposed models with SVGP-based output layers have mostly better performance than the common architectures with linear layers, and models with transfer learning outperform the ones without it by a large margin. On the whole, the proposed model with PPGP output layers demonstrates the best results under all settings. The difference lies in the performance with pre-training methods. Both pre-training methods only improve the performance in the settings without transfer learning.

2) *Results on Predictive Variances:* Due to the multivariate setting in this task, the mean of the predictive variances of different target variables is first computed before quantifying the predictive variances. Like the BAP task, a monotonically increasing trend is observed in models with PPGP and MC Dropout. The line of PPGP overlaps mostly with the one with MC Dropout, indicating relatively similar performance. In contrast, models with SVGP output layers deliver an almost flat trend w.r.t. the quantiles of the predictive variance. The models with the PPGP output layers have RMSE of 0.046 ± 0.012 for the evaluation pairs they are more confident with (predictive

TABLE II
LESION LOCALIZATION WITH DENSENET121

Output Layer	Transfer Learning	RMSE (No pre-training)	RMSE (Metric)	RMSE (CAE)
Linear*	Yes	0.102 ± 0.002	0.101 ± 0.002	0.102 ± 0.003
SVGP†	Yes	0.099 ± 0.003	0.101 ± 0.003	0.104 ± 0.004
PPGP†	Yes	0.098 ± 0.002	0.098 ± 0.002	0.099 ± 0.002
Linear*	No	0.116 ± 0.001	0.114 ± 0.003	0.114 ± 0.002
SVGP†	No	0.118 ± 0.002	0.114 ± 0.003	0.112 ± 0.003
PPGP†	No	0.115 ± 0.002	0.111 ± 0.005	0.110 ± 0.002

*Common architectures.

†With our proposed model.

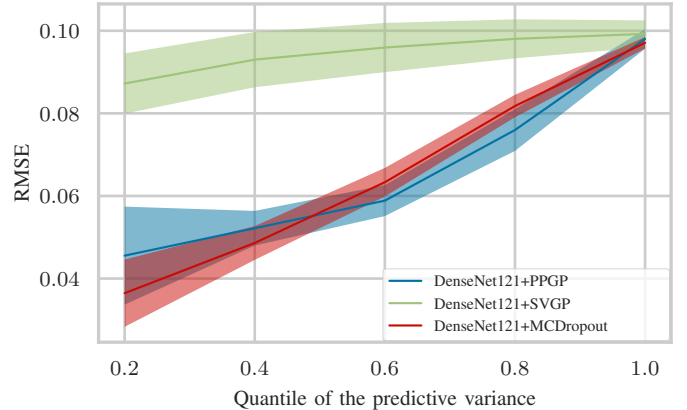


Fig. 5. Quantile Performance for the Lesion Localization with DenseNet121

variance at $q = 20\%$), which is less than one-half of the ones reported in Tab. II.

3) *Discussion:* Most observations and discussions in the univariate regression still hold in the multivariate task. The only difference lies in the performance of DML, where it does not improve the performance in the setting with transfer learning. The smaller size of the dataset compared to the one in the BAP task could be one possible reason. Another possible explanation is that the task’s multivariate nature makes it hard to define a suitable space for DML to facilitate DKL. Further improving the DKL performance in a multivariate setting of DML would be an exciting direction for our future work. Like the BAP task, we also conducted the same experiments with the backbone of the ResNet18. More details can be found in Appendix A and B.

V. CONCLUSIONS

This manuscript addresses the challenge that deep neural networks (DNNs) are often unable to provide uncertainty estimates for their predictions in regression tasks. Especially in the healthcare domain, this issue could prevent the further application of DNNs. We propose a model that consists of a deep Convolutional Neural Network (CNN) and a sparse Gaussian Process (GP). The former part serves as a trainable feature extractor that embeds raw images into a latent space. This enables the latter part to model the similarity of all

sample pairs with localized kernels more effectively in order to produce a predictive distribution for each data sample.

We show that such an architecture can be trained in an end-to-end fashion using stochastic gradient descent (SGD). We also analyzed multiple ways to boost the performance of such a model with different initialization and pre-training methods. Our approach is by no means limited to Convolutional Neural Networks, but could be generalized to other kinds of neural networks that best fit the nature of the data. We also observe a new specific challenge in this setup: We observe that randomly initialized inducing points in a sparse GP cause the prediction to degenerate to its prior when it consumes outputs from CNN backbones. We propose a simple solution that could also encourage further research in the task of jointly learning representations and GPs. Our experiments on the Bone Age Prediction and Lesion Localization tasks show that the proposed model delivers mostly better performance in terms of point estimates if compared to the baselines with a linear output layer. More importantly, we show that our model's prediction performance increases hand-in-hand with its predictive certainty. In other words, given a difficult test sample, our model can realize and communicate that the prediction thereof might be less trustworthy by generating a larger predictive variance. Finally, our model requires significantly less computational cost than popular MC Dropout methods, which motivates its usage in real-world online applications.

As future work, we are interested in studying the integration of multiple sparse GPs to deal with different types of input sources and model the relations between outputs. In addition, a combination of the interpretation using our uncertainty-aware model with the explainability methods like various saliency methods [54], [55] would be an exciting direction for exploration.

ACKNOWLEDGMENT

The authors acknowledge the support from Yan Ke on the Deep Lesion dataset [49] and the support by the German Federal Ministry for Education and Research (BMBF), funding project "MLWin" (grant 01IS18050).

GEFÖRDERT VOM



REFERENCES

- [1] V. Tresp, J. M. Overhage, M. Bundschuh, S. Rabizadeh, P. A. Fasching, and S. Yu, "Going digital: a survey on digitalization and large-scale data analytics in healthcare," *Proceedings of the IEEE*, vol. 104, no. 11, pp. 2180–2206, 2016.
- [2] C. Xiao, E. Choi, and J. Sun, "Opportunities and challenges in developing deep learning models using electronic health records data: a systematic review," *Journal of the American Medical Informatics Association*, vol. 25, no. 10, pp. 1419–1428, 2018.
- [3] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 427–436.
- [4] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1321–1330.
- [5] J. Zhang, B. Kailkhura, and T. Y.-J. Han, "Mix-n-match: Ensemble and compositional methods for uncertainty calibration in deep learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 11 117–11 128.
- [6] B. Zadrozny and C. Elkan, "Transforming classifier scores into accurate multiclass probability estimates," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002, pp. 694–699.
- [7] C. M. Bishop, "Mixture density networks," Aston University, Tech. Rep., 1994.
- [8] B. Kompa, J. Snoek, and A. L. Beam, "Second opinion needed: communicating uncertainty in medical machine learning," *NPJ Digital Medicine*, vol. 4, no. 1, pp. 1–6, 2021.
- [9] A. L. Beam and I. S. Kohane, "Big data and machine learning in health care," *Jama*, vol. 319, no. 13, pp. 1317–1318, 2018.
- [10] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*. PMLR, 2016, pp. 1050–1059.
- [11] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 5580–5590.
- [12] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.
- [13] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *CoRR*, vol. abs/1312.6114, 2014.
- [14] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *NIPS*, 2017.
- [15] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*, ser. Adaptive Computation and Machine Learning. Cambridge, MA, USA: MIT Press, Jan. 2006.
- [16] H. Liu, Y.-S. Ong, X. Shen, and J. Cai, "When gaussian process meets big data: A review of scalable gps," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 11, pp. 4405–4423, 2020.
- [17] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing, "Deep kernel learning," in *Artificial intelligence and statistics*. PMLR, 2016, pp. 370–378.
- [18] A. Esteva, A. Robicquet, B. Ramsundar, V. Kuleshov, M. DePristo, K. Chou, C. Cui, G. Corrado, S. Thrun, and J. Dean, "A guide to deep learning in healthcare," *Nature medicine*, vol. 25, no. 1, pp. 24–29, 2019.
- [19] P. Rajpurkar, J. Irvin, K. Zhu, B. Yang, H. Mehta, T. Duan, D. Ding, A. Bagul, C. Langlotz, K. Shpanskaya *et al.*, "Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning," *arXiv preprint arXiv:1711.05225*, 2017.
- [20] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.
- [21] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, "Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2097–2106.
- [22] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.
- [23] V. Tresp, "A bayesian committee machine," *Neural computation*, vol. 12, no. 11, pp. 2719–2741, 2000.
- [24] C. Williams and M. Seeger, "Using the nyström method to speed up kernel machines," in *Proceedings of the 14th annual conference on neural information processing systems*, no. CONF, 2001, pp. 682–688.
- [25] C. K. Williams, C. E. Rasmussen, A. Scwaighofer, and V. Tresp, "Observations on the nyström method for gaussian process prediction," *University of Edinburgh*, 2002.
- [26] E. Snelson and Z. Ghahramani, "Sparse gaussian processes using pseudo-inputs," *Advances in neural information processing systems*, vol. 18, pp. 1257–1264, 2005.

[27] M. Titsias, “Variational learning of inducing variables in sparse gaussian processes,” in *Artificial intelligence and statistics*. PMLR, 2009, pp. 567–574.

[28] J. Hensman, N. Fusi, and N. D. Lawrence, “Gaussian processes for big data,” in *Uncertainty in Artificial Intelligence*. Citeseer, 2013, p. 282.

[29] J. Bradshaw, A. G. d. G. Matthews, and Z. Ghahramani, “Adversarial examples, uncertainty, and transfer testing robustness in gaussian process hybrid deep networks,” *arXiv preprint arXiv:1707.02476*, 2017.

[30] A. Wilson and H. Nickisch, “Kernel interpolation for scalable structured gaussian processes (kiss-gp),” in *International Conference on Machine Learning*. PMLR, 2015, pp. 1775–1784.

[31] M. Jankowiak, G. Pleiss, and J. Gardner, “Parametric gaussian process regressors,” in *International Conference on Machine Learning*. PMLR, 2020, pp. 4702–4712.

[32] G. E. Hinton, S. Osindero, and Y.-W. Teh, “A fast learning algorithm for deep belief nets,” *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[33] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle *et al.*, “Greedy layer-wise training of deep networks,” *Advances in neural information processing systems*, vol. 19, p. 153, 2007.

[34] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?” in *Proceedings of the 27th International Conference on Neural Information Processing Systems-Volume 2*, 2014, pp. 3320–3328.

[35] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations,” in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607.

[36] K. Musgrave, S. Belongie, and S.-N. Lim, “A metric learning reality check,” in *European Conference on Computer Vision*. Springer, 2020, pp. 681–699.

[37] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[38] J. Hensman, A. Matthews, and Z. Ghahramani, “Scalable variational gaussian process classification,” in *Artificial Intelligence and Statistics*. PMLR, 2015, pp. 351–360.

[39] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, “On the surprising behavior of distance metrics in high dimensional space,” in *International conference on database theory*. Springer, 2001, pp. 420–434.

[40] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, “When is “nearest neighbor” meaningful?” in *International conference on database theory*. Springer, 1999, pp. 217–235.

[41] M. A. Álvarez, L. Rosasco, N. D. Lawrence *et al.*, “Kernels for vector-valued functions: A review,” *Foundations and Trends® in Machine Learning*, vol. 4, no. 3, pp. 195–266, 2012.

[42] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, “Cnn features off-the-shelf: an astounding baseline for recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2014, pp. 806–813.

[43] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, “Stacked convolutional auto-encoders for hierarchical feature extraction,” in *International conference on artificial neural networks*. Springer, 2011, pp. 52–59.

[44] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “Deepface: Closing the gap to human-level performance in face verification,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1701–1708.

[45] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[46] A. Hermans, L. Beyer, and B. Leibe, “In defense of the triplet loss for person re-identification,” *arXiv preprint arXiv:1703.07737*, 2017.

[47] S. S. e. a. Halabi, “The rsna pediatric bone age machine learning challenge,” *Radiology*, vol. 290, no. 2, pp. 498–503, 2019.

[48] D. B. Larson, M. C. Chen, M. P. Lungren, S. S. Halabi, N. V. Stence, and C. P. Langlotz, “Performance of a deep-learning neural network model in assessing skeletal maturity on pediatric hand radiographs,” *Radiology*, vol. 287, no. 1, pp. 313–322, 2018.

[49] K. Yan, X. Wang, L. Lu, and R. M. Summers, “Deeplesion: automated mining of large-scale lesion annotations and universal lesion detection with deep learning,” *Journal of medical imaging*, vol. 5, no. 3, p. 036501, 2018.

[50] K. Yan, Y. Peng, V. Sandfort, M. Bagheri, Z. Lu, and R. M. Summers, “Holistic and comprehensive annotation of clinically significant findings

on diverse ct images: learning from radiology reports and label ontology,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8523–8532.

[51] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, “Pytorch: An imperative style, high-performance deep learning library,” *Advances in Neural Information Processing Systems*, vol. 32, pp. 8026–8037, 2019.

[52] J. Gardner, G. Pleiss, K. Q. Weinberger, D. Bindel, and A. G. Wilson, “Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration,” *Advances in Neural Information Processing Systems*, vol. 31, pp. 7576–7586, 2018.

[53] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[54] J. Gu, Y. Yang, and V. Tresp, “Understanding individual decisions of cnns via contrastive backpropagation,” in *Proceedings of the Asian Conference on Computer Vision*. Springer, 2018, pp. 119–134.

[55] J. Gu, Z. Wu, and V. Tresp, “Introspective learning by distilling knowledge from online self-explanation,” in *Proceedings of the Asian Conference on Computer Vision*, 2020.

APPENDIX

A. Results on Point Estimates using ResNets

TABLE III
BONE AGE PREDICTION WITH RESNET18

Output Layer	Transfer Learning	RMSE (No pre-training)	RMSE (Metric)	RMSE (CAE)
Linear*	Yes	13.131 ± 0.129	12.419 ± 0.098	13.842 ± 0.283
SVGP†	Yes	12.632 ± 0.149	12.567 ± 0.051	13.375 ± 0.151
PPGP†	Yes	12.899 ± 0.114	12.658 ± 0.048	13.640 ± 0.292
Linear*	No	19.766 ± 0.134	16.533 ± 0.138	16.849 ± 0.275
SVGP†	No	19.090 ± 0.286	16.147 ± 0.264	16.641 ± 0.275
PPGP†	No	20.166 ± 0.371	16.986 ± 0.196	16.469 ± 0.222

TABLE IV
LESION LOCALIZATION WITH RESNET18

Output Layer	Transfer Learning	RMSE (No pre-training)	RMSE (Metric)	RMSE (CAE)
Linear*	Yes	0.110 ± 0.003	0.114 ± 0.001	0.111 ± 0.003
SVGP†	Yes	0.102 ± 0.002	0.107 ± 0.001	0.106 ± 0.003
PPGP†	Yes	0.103 ± 0.001	0.104 ± 0.001	0.103 ± 0.002
Linear*	No	0.148 ± 0.003	0.137 ± 0.002	0.123 ± 0.002
SVGP†	No	0.129 ± 0.002	0.131 ± 0.001	0.121 ± 0.002
PPGP†	No	0.133 ± 0.002	0.134 ± 0.002	0.119 ± 0.002

*Common architectures.

†With our proposed model.

B. Results on Predictive Variances using ResNets

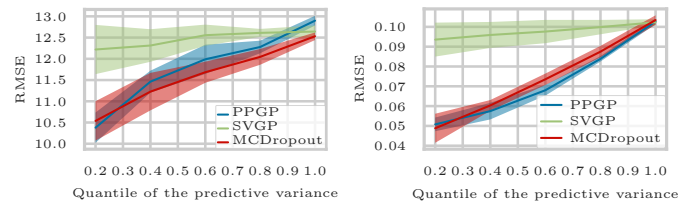


Fig. 6. Quantile Performance for tasks of Bone Age Prediction (left) and Lesion Localization (right) using ResNet18 as the backbone in our model