Formatting Instructions for TMLR Journal Submissions

Anonymous authors
Paper under double-blind review

Abstract

Learning the posterior distribution of a deep neural network is a difficult task. Even when using a Gaussian approximation, one faces the daunting challenge of estimating the $D \times D$ posterior covariance matrix, where D, the number of model parameters, can reach billions by today's standards. Some simplification has to be made, and one option is to use a low-rank plus diagonal approximation. One particular Gaussian distribution with a covariance matrix of this form is the factor analysis (FA) model. This paper introduces a novel approach to learning a FA posterior of a neural network via variational inference (VI). The algorithm - aptly named VIFA due to its use of VI and FA - is model-agnostic and can be readily applied to any type of neural network architecture with no extra effort. Crucially, the implementation scales to high-dimensional deep neural networks. Experiments demonstrate its effectiveness in learning the posterior and making predictions with uncertainty estimates, while keeping the computational overhead small compared to standard neural network training.

1 Submission of papers to TMLR

TMLR requires electronic submissions, processed by `https://openreview.net/`. See TMLR's website for more instructions.

If your paper is ultimately accepted, use option `accepted` with the `tmlr` package to adjust the format to the camera ready requirements, as follows:

`\usepackage[accepted]{tmlr}`.

You also need to specify the month and year by defining variables `month` and `year`, which respectively should be a 2-digit and 4-digit number. To de-anonymize and remove mentions to TMLR (for example for posting to preprint servers), use the preprint option, as in `\usepackage[preprint]{tmlr}`.

Please read carefully the instructions below, and follow them faithfully.

1.1 Style

Papers to be submitted to TMLR must be prepared according to the instructions presented here.

Authors are required to use the TMLR LaTeX style files obtainable at the TMLR website. Please make sure you use the current files and not previous versions. Tweaking the style files may be grounds for rejection.

1.2 Retrieval of style files

The style files for TMLR and other journal information are available online on the TMLR website. The file `tmlr.pdf` contains these instructions and illustrates the various formatting requirements your TMLR paper must satisfy. Submissions must be made using LaTeX and the style files `tmlr.sty` and `tmlr.bst` (to be used with LaTeX2e). The file `tmlr.tex` may be used as a "shell" for writing your paper. All you have to do is replace the author, title, abstract, and text of the paper with your own.

The formatting instructions contained in these style files are summarized in sections 2, 3, and 4 below.

2 General formatting instructions

The text must be confined within a rectangle 6.5 inches wide and 9 inches long. The left margin is 1 inch. Use 10 point type with a vertical spacing of 11 points. Computer Modern Bright is the preferred typeface throughout. Paragraphs are separated by 1/2 line space, with no indentation.

Paper title is 17 point, in bold and left-aligned. All pages should start at 1 inch from the top of the page.

Authors' names are set in boldface. Each name is placed above its corresponding address and has its corresponding email contact on the same line, in italic and right aligned. The lead author's name is to be listed first, and the co-authors' names are set to follow vertically.

Please pay special attention to the instructions in section 4 regarding figures, tables, acknowledgments, and references.

3 Headings: first level

First level headings are in bold, flush left and in point size 12. One line space before the first level heading and 1/2 line space after the first level heading.

3.1 Headings: second level

Second level headings are in bold, flush left and in point size 10. One line space before the second level heading and 1/2 line space after the second level heading.

3.1.1 Headings: third level

Third level headings are in bold, flush left and in point size 10. One line space before the third level heading and 1/2 line space after the third level heading.

4 Citations, figures, tables, references

These instructions apply to everyone, regardless of the formatter being used.

4.1 Citations within the text

Citations within the text should be based on the `natbib` package and include the authors' last names and year (with the "et al." construct for more than two authors). When the authors or the publication are included in the sentence, the citation should not be in parenthesis, using `\citet{}` (as in "See Hinton et al. (2006) for more information."). Otherwise, the citation should be in parenthesis using `\citep{}` (as in "Deep learning shows promise to make progress towards AI (Bengio & LeCun, 2007).").

The corresponding references are to be listed in alphabetical order of authors, in the **References** section. As to the format of the references themselves, any style is acceptable as long as it is used consistently.

4.2 Footnotes

Indicate footnotes with a number[1] in the text. Place the footnotes at the bottom of the page on which they appear. Precede the footnote with a horizontal rule of 2 inches.[2]

[1]Sample of the first footnote
[2]Sample of the second footnote

Table 1: Sample table title

PART	DESCRIPTION
Dendrite	Input terminal
Axon	Output terminal
Soma	Cell body (contains cell nucleus)

4.3 Figures

All artwork must be neat, clean, and legible. Lines should be dark enough for purposes of reproduction; art work should not be hand-drawn. The figure number and caption always appear after the figure. Place one line space before the figure caption, and one line space after the figure. The figure caption is lower case (except for first word and proper nouns); figures are numbered consecutively.

Make sure the figure caption does not get separated from the figure. Leave sufficient space to avoid splitting the figure and figure caption.

You may use color figures. However, it is best for the figure captions and the paper body to make sense if the paper is printed either in black/white or in color.

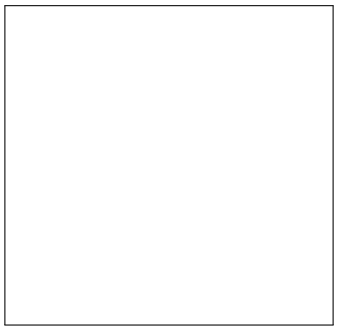


Figure 1: Sample figure caption.

4.4 Tables

All tables must be centered, neat, clean and legible. Do not use hand-drawn tables. The table number and title always appear before the table. See Table 1. Place one line space before the table title, one line space after the table title, and one line space after the table. The table title must be lower case (except for first word and proper nouns); tables are numbered consecutively.

5 Default Notation

In an attempt to encourage standardized notation, we have included the notation file from the textbook, *Deep Learning* Goodfellow et al. (2016) available at `https://github.com/goodfeli/dlbook_notation/`. Use of this style is not required and can be disabled by commenting out `math_commands.tex`.

Numbers and Arrays

a	A scalar (integer or real)
$\boldsymbol{a}$	A vector
$\boldsymbol{A}$	A matrix
$\mathsf{\mathbf{A}}$	A tensor
$\boldsymbol{I}_n$	Identity matrix with n rows and n columns
$\boldsymbol{I}$	Identity matrix with dimensionality implied by context
$\boldsymbol{e}^{(i)}$	Standard basis vector $[0, \dots, 0, 1, 0, \dots, 0]$ with a 1 at position i
$\text{diag}(\boldsymbol{a})$	A square, diagonal matrix with diagonal entries given by $\boldsymbol{a}$
a	A scalar random variable
$\mathbf{a}$	A vector-valued random variable
$\mathbf{A}$	A matrix-valued random variable

Sets and Graphs

$\mathbb{A}$	A set
$\mathbb{R}$	The set of real numbers
$\{0, 1\}$	The set containing 0 and 1
$\{0, 1, \dots, n\}$	The set of all integers between 0 and n
$[a, b]$	The real interval including a and b
$(a, b]$	The real interval excluding a but including b
$\mathbb{A} \backslash \mathbb{B}$	Set subtraction, i.e., the set containing the elements of $\mathbb{A}$ that are not in $\mathbb{B}$
$\mathcal{G}$	A graph
$Pa_{\mathcal{G}}(\mathrm{x}_i)$	The parents of x_i in $\mathcal{G}$

Indexing

a_i	Element i of vector $\boldsymbol{a}$, with indexing starting at 1
a_{-i}	All elements of vector $\boldsymbol{a}$ except for element i
$A_{i,j}$	Element i, j of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{i,:}$	Row i of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{:,i}$	Column i of matrix $\boldsymbol{A}$
$\mathsf{A}_{i,j,k}$	Element (i, j, k) of a 3-D tensor $\mathsf{\mathbf{A}}$
$\mathsf{\mathbf{A}}_{:,:,i}$	2-D slice of a 3-D tensor
a_i	Element i of the random vector $\mathbf{a}$

Calculus

$\frac{dy}{dx}$	Derivative of y with respect to x
$\frac{\partial y}{\partial x}$	Partial derivative of y with respect to x
$\nabla_{\boldsymbol{x}} y$	Gradient of y with respect to $\boldsymbol{x}$
$\nabla_{\boldsymbol{X}} y$	Matrix derivatives of y with respect to $\boldsymbol{X}$
$\nabla_{\mathbf{X}} y$	Tensor containing derivatives of y with respect to $\mathbf{X}$
$\frac{\partial f}{\partial \boldsymbol{x}}$	Jacobian matrix $\boldsymbol{J} \in \mathbb{R}^{m \times n}$ of $f : \mathbb{R}^n \to \mathbb{R}^m$
$\nabla^2_{\boldsymbol{x}} f(\boldsymbol{x})$ or $\boldsymbol{H}(f)(\boldsymbol{x})$	The Hessian matrix of f at input point $\boldsymbol{x}$
$\int f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral over the entire domain of $\boldsymbol{x}$
$\int_{\mathbb{S}} f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral with respect to $\boldsymbol{x}$ over the set $\mathbb{S}$

Probability and Information Theory

$P(\mathrm{a})$	A probability distribution over a discrete variable
$p(\mathrm{a})$	A probability distribution over a continuous variable, or over a variable whose type has not been specified
$\mathrm{a} \sim P$	Random variable a has distribution P
$\mathbb{E}_{\mathrm{x} \sim P}[f(x)]$ or $\mathbb{E} f(x)$	Expectation of $f(x)$ with respect to $P(\mathrm{x})$
$\mathrm{Var}(f(x))$	Variance of $f(x)$ under $P(\mathrm{x})$
$\mathrm{Cov}(f(x), g(x))$	Covariance of $f(x)$ and $g(x)$ under $P(\mathrm{x})$
$H(\mathrm{x})$	Shannon entropy of the random variable x
$D_{\mathrm{KL}}(P \Vert Q)$	Kullback-Leibler divergence of P and Q
$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution over $\boldsymbol{x}$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$

Functions

$f : \mathbb{A} \to \mathbb{B}$	The function f with domain $\mathbb{A}$ and range $\mathbb{B}$
$f \circ g$	Composition of the functions f and g
$f(\boldsymbol{x}; \boldsymbol{\theta})$	A function of $\boldsymbol{x}$ parametrized by $\boldsymbol{\theta}$. (Sometimes we write $f(\boldsymbol{x})$ and omit the argument $\boldsymbol{\theta}$ to lighten notation)
$\log x$	Natural logarithm of x
$\sigma(x)$	Logistic sigmoid, $\frac{1}{1 + \exp(-x)}$
$\zeta(x)$	Softplus, $\log(1 + \exp(x))$
$\Vert \boldsymbol{x} \Vert_p$	L^p norm of $\boldsymbol{x}$
$\Vert \boldsymbol{x} \Vert$	L^2 norm of $\boldsymbol{x}$
x^+	Positive part of x, i.e., $\max(0, x)$
$\mathbf{1}_{\mathrm{condition}}$	is 1 if the condition is true, 0 otherwise

6 Final instructions

Do not change any aspects of the formatting parameters in the style files. In particular, do not modify the width or length of the rectangle the text should fit into, and do not change font sizes (except perhaps in the **References** section; see below). Please note that pages should be numbered.

7 Preparing PostScript or PDF files

Please prepare PostScript or PDF files with paper size "US Letter", and not, for example, "A4". The -t letter option on dvips will produce US Letter files.

Consider directly generating PDF files using `pdflatex` (especially if you are a MiKTeX user). PDF figures must be substituted for EPS figures, however.

Otherwise, please generate your PostScript and PDF files with the following commands:

```
dvips mypaper.dvi -t letter -Ppdf -G0 -o mypaper.ps
ps2pdf mypaper.ps mypaper.pdf
```

7.1 Margins in LaTeX

Most of the margin problems come from figures positioned by hand using `\special` or other commands. We suggest using the command `\includegraphics` from the graphicx package. Always specify the figure width as a multiple of the line width as in the example below using .eps graphics

```
\usepackage[dvips]{graphicx} ...
\includegraphics[width=0.8\linewidth]{myfile.eps}
```

or

```
\usepackage[pdftex]{graphicx} ...
\includegraphics[width=0.8\linewidth]{myfile.pdf}
```

for .pdf graphics. See section 4.4 in the graphics bundle documentation (`http://www.ctan.org/tex-archive/macros/latex/required/graphics/grfguide.ps`)

A number of width problems arise when LaTeX cannot properly hyphenate a line. Please give LaTeX hyphenation hints using the `\-` command.

Broader Impact Statement

In this optional section, TMLR encourages authors to discuss possible repercussions of their work, notably any potential negative impact that a user of this research should be aware of. Authors should consult the TMLR Ethics Guidelines available on the TMLR website for guidance on how to approach this subject.

Author Contributions

If you'd like to, you may include a section for author contributions as is done in many journals. This is optional and at the discretion of the authors. Only add this information once your submission is accepted and deanonymized.

Acknowledgments

Use unnumbered third level headings for the acknowledgments. All acknowledgments, including those to funding agencies, go at the end of the paper. Only add this information once your submission is accepted and deanonymized.

8 Introduction

In recent years, deep learning (Goodfellow et al., 2016) has come to dominate many areas of machine learning. Deep learning models continue to achieve state of the art results in several domains, such as CoAtNet-7 (Dai et al., 2021) for image classification and Megatron-LM (Shoeybi et al., 2019) for language modelling. One reason for their success is the availability of huge training sets, such as the 300 millions images in JFT-300M (Sun et al., 2017) or the 100 million tokens in WikiText-103 (Merity et al., 2016). The number of training examples, however, pales in comparison to the number of learnable parameters in some deep models. CoAtNet-7, for example, has 2.4 billion parameters, whereas Megatron-LM has a whopping 8.3 billion. This is a common feature of deep learning: the number of model parameters is often greater than the number of training examples. In this scenario, the model is severely *underspecified* by the data and many different settings of the parameters are able to explain the training set equally well. In some domains, choosing a single setting of the parameters and predicting a point estimate for each test example is satisfactory. However, for other critical applications, some measure of the uncertainty in the prediction is also required.

This is where Bayesian deep learning comes in. Instead of choosing a single parameter vector $\boldsymbol{\theta} \in \mathbb{R}^D$ for the neural network, predictions are made using *all* possible parameter vectors weighted by their posterior probabilities, given the training data $\mathcal{D}$. Formally, the posterior predictive distribution of the output y given the input $\boldsymbol{x}$ is

$$p(y|\boldsymbol{x}, \mathcal{D}) = \int p(y|\boldsymbol{x}, \boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta}. \tag{1}$$

In practice, computing this integral is intractable due to the dimensionality and non-linearities of a deep neural network. However, the integral can be approximated by a *Bayesian model average*,

$$p(y|\boldsymbol{x}, \mathcal{D}) \approx \frac{1}{L} \sum_{l=1}^{L} p(y|\boldsymbol{x}, \boldsymbol{\theta}_l), \quad \boldsymbol{\theta}_l \sim p(\boldsymbol{\theta}|\mathcal{D}). \tag{2}$$

Note that this requires samples from the *posterior*, $p(\boldsymbol{\theta}|\mathcal{D})$, which is defined as

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathcal{D})}, \tag{3}$$

where $p(\mathcal{D}|\boldsymbol{\theta})$ is the *likelihood* that $\boldsymbol{\theta}$ generated $\mathcal{D}$, $p(\boldsymbol{\theta})$ is the *prior* and $p(\mathcal{D})$ is the *marginal likelihood*. Unfortunately, this calculation is also intractable due to the high-dimensional integral

$$p(\mathcal{D}) = \int p(\mathcal{D}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta}. \tag{4}$$

A common alternative is to approximate $p(\boldsymbol{\theta}|\mathcal{D})$ with a Gaussian distribution with mean $\boldsymbol{\mu} \in \mathbb{R}^D$ and covariance $\boldsymbol{\Sigma} \in \mathbb{R}^{D \times D}$, denoted by $\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$. The key practical challenge is then finding a way to approximate $\boldsymbol{\Sigma}$, since the full $D \times D$ covariance matrix will not fit into memory for large D.

One of the simplest approaches is to use a diagonal - or *mean-field* - approximation (Blundell et al., 2015; Graves, 2011; Hernández-Lobato & Adams, 2015; Khan et al., 2018; Ranganath et al., 2014). This is convenient in the sense that the covariance matrix is completely specified by D parameters, but it does not allow for any posterior correlations between different elements of $\boldsymbol{\theta}$. A more flexible approach is a low-rank plus diagonal approximation, $\boldsymbol{F}\boldsymbol{F}^T + \Psi$, where $\boldsymbol{F} \in \mathbb{R}^{D \times K}$ and Ψ is a $D \times D$ diagonal matrix. This preserves some of the off-diagonal structure of the covariance matrix and is practically feasible for $K \ll D$. Two recent methods which adopt this approach are SWAG (Maddox et al., 2019) and SLANG (Mishkin et al., 2018). While both algorithms have achieved promising results, they also have fundamental limitations. In the case of SWAG, it does not make use of all the available data to fit the low-rank plus diagonal covariance matrix, whereas the current implementation of SLANG only supports fully-connected neural networks and would require non-trivial modifications for other types of architectures.

The work in this paper is based on the low-rank plus diagonal factor analysis (FA) model (Barber, 2012), which can help to overcome these limitations. The main contribution is a novel variational inference (VI)

algorithm called VIFA which can be applied to any neural network architecture to approximate its posterior with a FA distribution.

TODO: add overview if different sections.

9 Simulations

9.1 Linear regression posterior estimation with synthetic data

The purpose of these simulations is to test that VIFA is able to learn the posterior distribution of a very simple linear regression model with two learnable parameters.

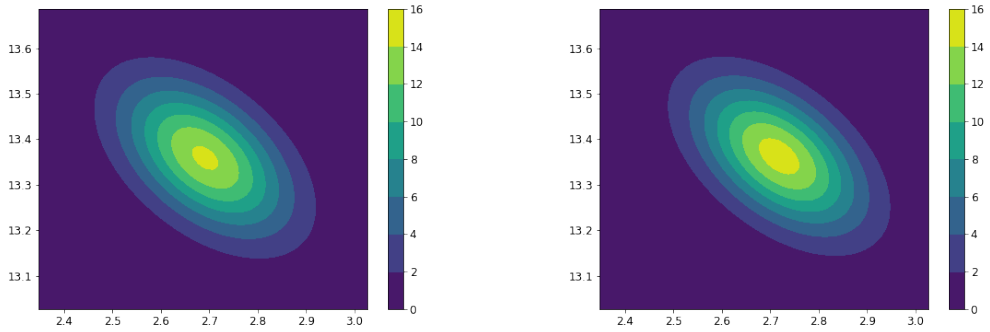
9.1.1 Methodology

Synthetic data was generated as follows. First, 1000 inputs $\boldsymbol{x} \in \mathbb{R}^2$ were sampled from a multivariate zero mean Gaussian distribution with unit variances and covariances of 0.5. Next, the linear regression parameter vector $\boldsymbol{\theta} \in \mathbb{R}^2$ was sampled from $\mathcal{N}(\boldsymbol{\theta}; 0, \alpha^{-1}\boldsymbol{I})$ with $\alpha = 0.01$. Then the outputs $y \in \mathbb{R}$ were generated according to the equation $y = \boldsymbol{\theta}^T\boldsymbol{x} + \epsilon$, where $\epsilon \sim \mathcal{N}(\epsilon; 0, \beta^{-1})$ with $\beta = 0.1$.

Using this data, the true posterior distribution was evaluated in closed form (TODO: needs citation or reference to equation) and an approximate posterior with latent dimension $K = 1$ was estimated via VIFA. VIFA ran for 5000 epochs with a batch size of $M = 100$ and a Monte Carlo average size of $L = 10$. The learning rates $\eta_{\boldsymbol{c}}$, $\eta_{\boldsymbol{F}}$ and $\eta_{\boldsymbol{\gamma}}$ were set to 0.01, 0.0001 and 0.01, respectively. The reasoning for using a smaller learning rate for $\boldsymbol{F}$ was that its contribution to the full covariance matrix is $\boldsymbol{F}\boldsymbol{F}^T$. Since this is regression, the likelihood function used in VIFA was set to $\mathcal{N}(y; \boldsymbol{\theta}^T\boldsymbol{x}, \beta^{-1})$. Finally, to improve numerical stability, any gradients with Frobenius norm greater than 10 were rescaled to have norm of exactly 10.

9.1.2 Results and discussion

Figure 2 shows a qualitative comparison between the ground truth and approximate linear regression posteriors. (TODO: add discussion here plus results for more random seeds in appendix).



(a) Ground truth (b) VIFA

Figure 2: The ground truth posterior pdf of a linear regression model with two learnable parameters, plus the pdf of a FA model with a single latent dimension which was fit to the same data using VIFA.

9.2 Logistic regression posterior estimation with synthetic data

The purpose of these simulations is to test that VIFA is able to learn the posterior distribution of a very simple logistic regression model with two learnable parameters.

9.2.1 Methodology

Synthetic inputs $\boldsymbol{x} \in \mathbb{R}^2$ and the logistic regression parameter vector $\boldsymbol{\theta} \in \mathbb{R}^2$ were sampled in the same way as in Section 9.1.1. However, this time 3000 inputs were used. To generate each output $y \in \{0, 1\}$, the positive class probability was first evaluated as $p = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$, where $\sigma : \mathbb{R} \to [0, 1]$ denotes the logistic sigmoid function. Then a random number was sampled uniformly from the interval $[0, 1]$ and y was set to 1 if this number was less than p, else 0.

Unlike linear regression, there is no closed form solution for the true posterior of a logistic regression model. In this case, the ground truth posterior was evaluated by first looping over a 2D grid around the true parameter vector $\boldsymbol{\theta}$ and evaluating the unnormalised log posterior probability at each point in the grid. Formally, this is

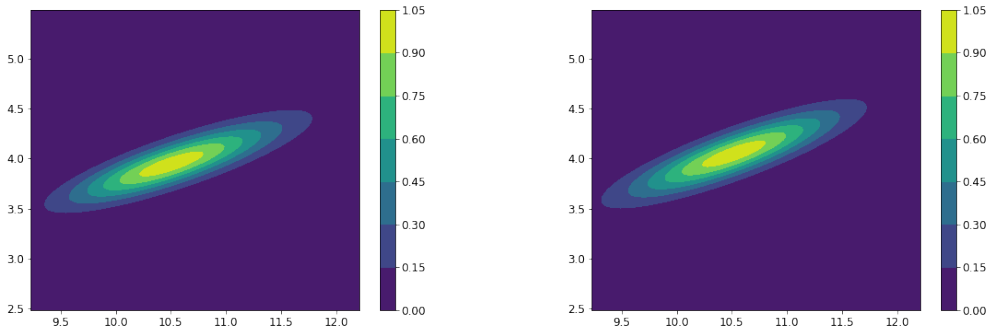
$$\mathcal{N}(\boldsymbol{\theta}; 0, \alpha^{-1}\boldsymbol{I}) \prod_{n=1}^{3000} \sigma(\boldsymbol{\theta}^T \boldsymbol{x}_n)^{y_n} \sigma(\boldsymbol{\theta}^T \boldsymbol{x}_n)^{1-y_n}. \tag{5}$$

Then the values in the grid were scaled such that the maximum value was equal to 1. This posterior is only correct up to a constant, but suffices for a qualitative comparison.

The posterior was then approximated via VIFA using the exact same hyperparameters as in Section 9.1.1. This time, the likelihood function used in VIFA was set to binary cross-entropy. That is, $\sigma(\boldsymbol{\theta}^T \boldsymbol{x}_n)^{y_n} \sigma(\boldsymbol{\theta}^T \boldsymbol{x}_n)^{1-y_n}$.

9.2.2 Results and discussion

Figure 3 shows a qualitative comparison between the ground truth and approximate logistic regression posteriors. (TODO: add discussion here plus results for more random seeds in appendix).



(a) Ground truth

(b) VIFA

Figure 3: The ground truth posterior pdf of a logistic regression model with two learnable parameters, plus the pdf of a FA model with a single latent dimension which was fit to the same data using VIFA. Both posteriors are scaled such that the maximum value is equal to 1.

10 Application: Medical Imaging

To illustrate the model-agnostic nature of VIFA algorithm, we adopt our algorithm to Convolutional Neural Networks (CNN) and evaluate its effectiveness for medical image diagnosis, which is often regarded as a safety-critical task where prediction reliability matters significantly. In particular, we focus on a diabetic retinopathy detection problem in India [3]. The dataset contains 3662 samples, and there are 5 severity levels of

[3] APTOS 2019 Blindness Detection Dataset, https://www.kaggle.com/competitions/aptos2019-blindness-detection/overview

diabetic retinopathy, which mean label $y \in \{0, 1, 2, 3, 4\}$. To simplify the problem, we follow the strategy used in Leibig et al. (2017) to convert all instance labels into binary forms. This is done by categorizing the classes into two groups: sight-threatening diabetic retinopathy and non-sight-threatening diabetic retinopathy. The former includes cases of moderate diabetic retinopathy or more severe (classes 2, 3, and 4), while the latter includes cases with no or mild diabetic retinopathy (classes 0 and 1).

10.1 Methodology

The Convolutional Neural Network model we select is the resnet-18 model, which contains roughly 11.2M trainable parameters. Due to the non-linear nature of the neural network model, there is no closed-form solution for the true posterior of resnet-18 model on the dataset. In this case, our focus is on the prediction performance made by Bayesian model averaging of resnet-18 model trained via VIFA algorithm. To evaluate its effectiveness, we also train resnet-18 model via standard gradient descent and use the performance as a baseline.

To apply VIFA algorithm to resnet-18 model, we utilize a 2-stages strategy for updating posterior distribution parameters $\boldsymbol{c}$, $\boldsymbol{F}$, and ψ. In the first stage of training, covariance-related parameters $\boldsymbol{F}$ and ψ stay unchanged with small absolute values (less than 1e-5), and the parameter $\boldsymbol{c}$ is trained starting from pre-trained resnet-18 weights as initialization. During the second stage, the learning rates for $\boldsymbol{F}$ and ψ gradually increases from 0 to target values, while keeping the learning rate for $\boldsymbol{c}$ constant. Learning rate decay scheduler is in use for $\boldsymbol{c}$ during both stages. To keep things simple, learning rates for $\boldsymbol{c}$, $\boldsymbol{F}$, and ψ are set to the same values η, which are optimized as hyper-parameter on the validation dataset. We run a total of 10 epochs, with 2 epochs in stage 1. The FA distribution has a latent dimension of 1, and the gradients for parameter update are computed based on mini-batches of size 16. All gradients are clipped to have Frobenius norm less than or equal to 10, and parameter update is performed after 12 times of gradients calculations. We use Adam optimizer (Kingma & Ba, 2014) to perform gradient updates.

In order to report robust results, test metrics are averaged over 5 train-valid-test splits with split ratios 50%, 20%, 30% respectively. For each data split, hyper-parameters prior precision α and learning rate η are tuned via 6 times random searches, where α and η are log-uniformly sampled from [0.01, 10] and [1e-6, 5e-4] respectively. After the best hyper-parameter configuration are found for the current split, evaluation metrics are computed on test data with Monte Carlo average size S being 100.

10.2 Prediction with Uncertainty

One advantage of Bayesian deep learning methods compared to traditional Frequentist training is that we are able to obtain predictive uncertainty at the same time as getting predictions. Based on this uncertainty, Band et al. (2022) propose an automated diagnostic workflow for medical imaging: When given input, a model generates a prediction along with an associated uncertainty estimation. If the uncertainty estimation falls below a specified reference threshold, indicating low uncertainty, the diagnosis proceeds without additional examination. However, if the threshold is not met, a medical professional is consulted for further review. In general, we desire a model's predictive uncertainty to have a strong correlation with the accuracy of its predictions. High-quality predictive uncertainty estimates can be a fail-safe against false predictions (Band et al., 2022).

There are multiple ways to define test sample uncertainty [4] One definition adopted in Band et al. (2022) is that for any given test sample $\boldsymbol{x}_*$ the predictive uncertainty is equal to the entropy of the predictive distribution:

$$H(\mathbb{E}_{\boldsymbol{\theta}\sim Q(\boldsymbol{\theta})}[p(\mathrm{y}_* \mid f(\boldsymbol{x}_*;\boldsymbol{\theta}))]) = -\sum_{c\in\{0,1\}} \mathbb{E}_{\boldsymbol{\theta}\sim Q(\boldsymbol{\theta})}[p(\mathrm{y}_* = c \mid f(\boldsymbol{x}_*;\boldsymbol{\theta}))] \log \mathbb{E}_{\boldsymbol{\theta}\sim Q(\boldsymbol{\theta})}[p(\mathrm{y}_* = c \mid f(\boldsymbol{x}_*;\boldsymbol{\theta}))] \quad (6)$$

[4] In literature, the predictive uncertainty are usually argued to have different sources, such as aleatoric uncertainty and epistemic uncertainty, and different definitions may lead to varied implications (Ulmer et al., 2023; D'Angelo & Fortuin, 2021). However in this paper, we do not distinguish them, and just regard predictive uncertainty as a measure which reflects to what extent we trust the model's prediction result.

where H represents the Shannon entropy, $f(\boldsymbol{x}_*;\boldsymbol{\theta})$ is logit value, $p(\mathrm{y}_* = c \mid f(\boldsymbol{x}_*;\boldsymbol{\theta}))$ is a binary cross-entropy likelihood function and $Q(\boldsymbol{\theta})$ is the variational distribution. In practice, the expectation $\mathbb{E}_{\boldsymbol{\theta}\sim Q(\boldsymbol{\theta})}[p(\mathrm{y}_* \mid f(\boldsymbol{x}_*;\boldsymbol{\theta}))]$ is approximated by S Monte Carlo samples, $\mathbb{E}_{\boldsymbol{\theta}\sim Q(\boldsymbol{\theta})}[p(\mathrm{y}_* \mid f(\boldsymbol{x}_*;\boldsymbol{\theta}))] \approx \frac{1}{S}\sum_i^S p(\mathrm{y}_* \mid f(\boldsymbol{x}_*;\boldsymbol{\theta}^{(i)}))$, here $\{\boldsymbol{\theta}^{(i)}\}_{i=1}^S$ are sampled from (trained) variational distribution $Q(\boldsymbol{\theta})$, and $p(\mathrm{y}_* \mid f(\boldsymbol{x}_*;\boldsymbol{\theta}^{(i)}))$ denotes the predictive distribution given parameter realization $\boldsymbol{\theta}^{(i)}$.

Another way to define predictive uncertainty is by measuring the model disagreement (D'Angelo & Fortuin, 2021), which is computed as:

$$\mathcal{MD}^2(\boldsymbol{x}_*) = \sum_{c\in\{0,1\}} \mathbb{E}_{\boldsymbol{\theta}\sim Q(\boldsymbol{\theta})}[(p(\mathrm{y}_* = c \mid f(\boldsymbol{x}_*;\boldsymbol{\theta})) - \mathbb{E}_{\boldsymbol{\theta}\sim Q(\boldsymbol{\theta})}[p(\mathrm{y}_* = c \mid f(\boldsymbol{x}_*;\boldsymbol{\theta}))])^2] \tag{7}$$

This quality represents how much 'disagreement' exist among the distribution of models. In practice, this quantity is approximated by $\mathcal{MD}^2(\boldsymbol{x}_*) \approx \sum_{c\in\{0,1\}} \frac{1}{S}\sum_{\boldsymbol{\theta}^{(i)}\in\{\boldsymbol{\theta}^{(1)},\ldots,\boldsymbol{\theta}^{(S)}\}}(p(\mathrm{y}_* = c \mid f(\boldsymbol{x}_*;\boldsymbol{\theta}^{(i)})) - \frac{1}{S}\sum_{\boldsymbol{\theta}^{(i)}\in\{\boldsymbol{\theta}^{(1)},\ldots,\boldsymbol{\theta}^{(S)}\}}[p(\mathrm{y}_* = c \mid f(\boldsymbol{x}_*;\boldsymbol{\theta}^{(i)}))])^2$, where $\{\boldsymbol{\theta}^{(1)},...,\boldsymbol{\theta}^{(S)}\}$ are Monte Carlo samples from the variational distribution $Q(\boldsymbol{\theta})$. It is easy to see if all models agree on the prediction $\boldsymbol{x}_*$, the disagreement measure $\mathcal{MD}^2(\boldsymbol{x}_*)$ becomes zero. On the other hand, the larger the score $\mathcal{MD}^2(\boldsymbol{x}_*)$, the server disagreement exists between predictive distributions.

By computing uncertainty scores for each test sample using either of the two aforementioned methods, we can implement selective prediction in the automated diagnostic workflow. This involves sorting the test samples based on their uncertainty scores and evaluating the model's performance on the 90%, 80%, 70%, 60%, and 50% most certain samples. With high-quality uncertainties, we expect the model's performance to increase as we extract fewer and fewer test samples with ascending predictive uncertainty. From another perspective, the assessment score of selective prediction also reflects the utility of the uncertainty obtained from Bayesian deep learning methods.

10.3 Results and discussion

10.3.1 Prediction Performance

Table 2 shows a comparison between the performance of VIFA and traditional gradient descent training of resnet-18 model. For most metrics, such as Accuracy, F1-score and AU-ROC, the performances of VIFA are very competitive to that of Gradient Descent and their value are very close to each other. VIFA leads to a better Precision value while Gradient Descent has more satisfying Recall score, but their differences are not significant. The only exceptional metric is the training time, for which the time used for Gradient Descent training is around 6% less than VIFA training. However, it is noticeable that VIFA gets reliable predictive uncertainties simultaneously and a little bit more computational time compared to this achievable uncertainty quantification is reasonable.

10.3.2 Uncertainty Effectiveness

Table 3 shows the test accuracy of selective prediction with different uncertainty levels (Results for F1 score and AU-ROC are in Table 4, 5 in the Appendix). Predictive uncertainties calculated from two approaches: predictive entropy and model disagreement score are in use. We observe that prediction performance continuously increase with the decline of predictive uncertainty. The best performance reached when extracting 50% of most certain samples for assessment, which is the lowest uncertainty we accept. This result illustrates that predictive uncertainty has negative correlation with prediction accuracy, which shows the effectiveness of the uncertainty from the VIFA-resnet model.

References

Neil Band, Tim GJ Rudner, Qixuan Feng, Angelos Filos, Zachary Nado, Michael W Dusenberry, Ghassen Jerfel, Dustin Tran, and Yarin Gal. Benchmarking bayesian deep learning on diabetic retinopathy detection tasks. *arXiv preprint arXiv:2211.12717*, 2022.

Table 2: Mean test results for resnet-18 model on diabetic retinopathy detection task. The performances of applying VIFA and Gradient Descent algorithms are compared. Metrics include accuracy, precision, recall, F1-score, and Area Under Receiver Operating Characteristic curve (AU-ROC). The mean results over 5 different train-valid-test splits are shown. The results for test metrics include standard errors. The runtime refers to the total runtime for a single train-valid-test split. All experiments were executed on machines with Intel(R) Xeon(R) Silver 4114 CPU and NVIDIA GeForce RTX 2080 Ti GPU. The best results on each row are highlighted in bold (no score is bolded if both scores are competitive).

	VIFA	**Gradient Descent**
Accuracy	0.928±0.003	0.927±0.004
Precision	**0.921±0.009**	0.914±0.009
Recall	0.904±0.014	**0.907±0.009**
F1-Score	0.912±0.004	0.911±0.005
AU-ROC	0.980±0.002	0.979±0.002
Time(minutes)	463.2±2.8	**435.7±1.8**

Table 3: Mean test accuracies of selective prediction under different uncertainty levels. Uncertainty scores calculated from two distinct approaches are employed, which are predictive entropy and model disagreement. Test samples are arranged in ascending order based on their uncertainty scores. 'Proportion of Samples' indicates the percentage of ordered samples used for evaluation. The results for test accuracy include standard errors. The best results in each column are highlighted in bold.

Proportion of Samples	**Predictive Entropy**	**Model Disagreement**
90%	0.957±0.005	0.956±0.005
80%	0.973±0.004	0.975±0.003
70%	0.983±0.002	0.984±0.002
60%	0.991±0.002	0.99±0.002
50%	**0.994±0.001**	**0.994±0.001**

David Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.

Yoshua Bengio and Yann LeCun. Scaling learning algorithms towards AI. In *Large Scale Kernel Machines*. MIT Press, 2007.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. *International conference on machine learning*, pp. 1613–1622, 2015.

Zihang Dai, Hanxiao Liu, Quoc V Le, and Mingxing Tan. CoAtNet: Marrying convolution and attention for all data sizes. *Proceedings of Advances in Neural Information Processing Systems*, 34, 2021.

Francesco D'Angelo and Vincent Fortuin. Repulsive deep ensembles are bayesian. *Advances in Neural Information Processing Systems*, 34:3451–3465, 2021.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

Alex Graves. Practical variational inference for neural networks. *Proceeding of Advances in Neural Information Processing Systems*, 24, 2011.

José M Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. *International Conference on Machine Learning*, pp. 1861–1869, 2015.

Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

Mohammad E Khan, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava. Fast and scalable Bayesian deep learning by weight-perturbation in Adam. *Proceedings of the International Conference on Machine Learning*, 35:2611—-2620, 2018.

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

Christian Leibig, Vaneeda Allken, Murat Seçkin Ayhan, Philipp Berens, and Siegfried Wahl. Leveraging uncertainty information from deep neural networks for disease detection. *Scientific reports*, 7(1):1–14, 2017.

Wesley Maddox, Timur Garipov, Pavel Izmailov, Dmitry Vetrov, and Andrew G Wilson. A simple baseline for Bayesian uncertainty in deep learning. *Proceedings of Advances in Neural Information Processing Systems*, 32, 2019.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Aaron Mishkin, Frederik Kunstner, Didrik Nielsen, Mark Schmidt, and Mohammad E Khan. SLANG: Fast structured covariance approximations for Bayesian deep learning with natural gradient. *Proceedings of Advances in Neural Information Processing Systems*, 31:6248–6258, 2018.

Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. *Artificial Intelligence and Statistics*, pp. 814–822, 2014.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-LM: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. *Proceedings of the IEEE international conference on computer vision*, pp. 843–852, 2017.

Dennis Ulmer, Christian Hardmeier, and Jes Frellsen. Prior and posterior networks: A survey on evidential deep learning methods for uncertainty estimation. *arXiv preprint arXiv:2110.03051*, 1(3), 2023.

A Appendix

You may include other additional sections here.

A.1 Uncertainty on Medical Imaging

Table 4: Mean F1-scores of selective prediction under different uncertainty levels. Uncertainty scores calculated from two distinct approaches are employed, which are predictive entropy and model disagreement. Test samples are arranged in ascending order based on their uncertainty scores. 'Proportion of Samples' indicates the percentage of ordered samples used for testing. The results for F1 scores include standard errors. The best results in each column are highlighted in bold.

Proportion of Samples	Predictive Entropy	Model Disagreement
90%	0.946±0.007	0.945±0.007
80%	0.965±0.005	0.966±0.004
70%	0.975±0.004	0.976±0.004
60%	0.986±0.003	0.985±0.003
50%	**0.989±0.002**	**0.989±0.002**

Table 5: Mean AU-ROC scores of selective prediction under different uncertainty levels. Uncertainty scores calculated from two distinct approaches are employed, which are predictive entropy and model disagreement. Test samples are arranged in ascending order based on their uncertainty scores. 'Proportion of Samples' indicates the percentage of ordered samples used for testing. The results for AU-ROC include standard errors. The best results in each column are highlighted in bold.

Proportion of Samples	Predictive Entropy	Model Disagreement
90%	0.984+0.002	0.984+0.003
80%	0.989+0.002	0.989+0.002
70%	0.992+0.002	0.992+0.002
60%	0.994+0.002	0.994+0.002
50%	**0.995+0.001**	**0.995+0.002**