Regression with Uncertainty Quantification in Large Scale Complex Data

Nicholas Wilkins
Google, Inc.
Mountain View, CA
nicholaswilkins@gmail.com

Michael Johnson
Rochester Institute of Technology
Rochester, NY
mxj5897@rit.edu

Ifeoma Nwogu
University at Buffalo
Amherst, NY
inwogu@buffalo.edu

***Abstract*—While several methods for predicting uncertainty on deep networks have been recently proposed, they do not always readily translate to large and complex datasets without significant overhead. In this paper we utilize a special instance of the Mixture Density Networks (MDNs) to produce an elegant and compact approach to quantify uncertainty in regression problems. When applied to standard regression benchmark datasets, we show an improvement in predictive log-likelihood and root-mean-square-error when compared to existing state-of-the-art methods. We demonstrate the efficacy and practical usefulness of the method for (i) predicting future stock prices from stochastic, highly volatile time-series data; (ii) anomaly detection in real-life highly complex video segments; and (iii) the task of age estimation and data cleansing on the challenging IMDb-Wiki dataset of half a million face images.**

I. Introduction

In a standard regression problem, the goal is to learn an optimal mapping (under some loss function) from a feature space X to some target space Y; *i.e.* we wish to learn the function $\hat{f} : X \rightarrow Y$ such that the loss function $\mathcal{L}$ is minimized. In standard regression problems a point estimate is typically predicted but there is no information about the quality or confidence of that prediction. The main focus of this work therefore, is to construct a regressor which efficiently regresses onto a Gaussian distribution (parameterized by its mean and variance) on the target space. The parameters of the target distribution can therefore shed some insight into the quality of the prediction results. The choice of a single Gaussian as the target distribution is in standing with traditional statistics methods where when the measurement errors occurring in regression problems are assumed to follow a normal distribution.

Although transformative and highly successful and useful in a wide range of applications of machine learning and AI, many deep learning techniques only provide point estimates and seldom provide a means to understand the inherent uncertainty in the data. They are therefore frequently incapable of expressing their own limitations. Although in classification one can determine how far training samples are from decision boundaries, this information is significantly different from understanding the inherent limitations of the learning system. Such limitations can potentially have disastrous impacts in many important real-life scenarios.

For many areas of scientific study, especially in areas of critical importance such as in medical image analysis for patient diagnosis, this lack of uncertainty quantification is highly problematic. The inability to understand and quantify the model's confidence in its predicted values is a high source of potential risk and liability [1]. For example, when faced with a difficult diagnosis, the ability for a deep learning system to report large uncertainties would allow for human operators to intervene and review those specific cases. *If deep learning is to be widely used for critical applications in practical settings, a key requirement would be the ability to provide statistically meaningful uncertainty measurements alongside their predictions.*

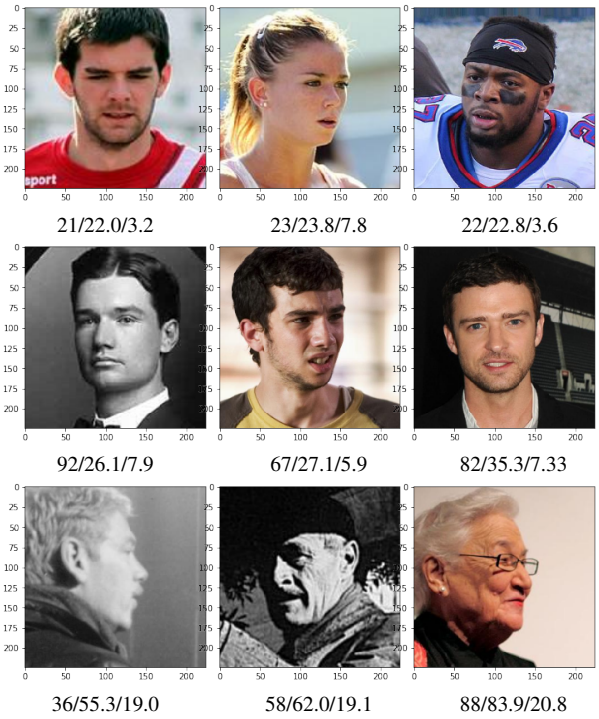


Fig. 1. The three numbers below each image a/b/c correspond to: (a) the target or actual age (as provided in the dataset), (b) the estimated age (as predicted by our regression network) and (c) the uncertainty value reported by the network (the higher the value, the more uncertain the prediction). The top row shows sample faces on which the network reported the lowest error values. The middle row shows faces on which were reported the highest errors (empirically note the clearly wrong target labels); and the last row shows faces on which the network reported the highest uncertainty.

Figure 1 shows not only the promising prediction results of the regressor, but goes beyond the currently existing limitation when a standard regressor is applied to a noisy, real-world dataset, by providing measures of uncertainty on the prediction. The architecture only performed poorly when the ground truth was wrong (see the middle row of Figure 1). These results demonstrate that our model is

Identify applicable funding agency here. If none, delete this.

capable of capturing epistemic uncertainty. Additionally, this architecture's uncertainty not only expressed how confident the model was, but also how "clean" the data sample was.

In his 1994 Ph.D. thesis, Bishop [2] introduced Mixture Density Networks (MDNs); where a neural network was used to predict a probability distribution over the target value Y, rather than a single point estimate. The MDNs trained with a fixed number of mixtures of Gaussian components over the course of the training scheme using the negative log likelihood (NLL) as the loss function to the network. This training scheme had the potential to address many of the issues highlighted above, but probably due to limitations in computing power in the 1990's, MDNs did not gain as wide popularity.

In this paper we present an elegant and simplified approach to quantify uncertainty in large-scale regression problems. We propose a compact training scheme that does not require any significant additional overhead when compared with traditional training methods. Using the uncertainties produced by the system, we address a series of real-world problems such as explaining the behavior of specific stocks in the market, prediction age from pictures of faces and detecting anomalies in complex video segments. We also illustrate how this can be utilized for cleaning datasets and removing erroneous data autonomously.

II. Prior Work

As uncertainty estimation for deep learning predictions is immensely useful, much prior research has been conducted in this area. Most deep neural networks based uncertainty estimation methods can be grouped into two categories; (i) the Bayesian neural network category, where a prior distribution is imposed on the network weights and data is used to update its posterior distribution. In these types of systems, inference is done via Markov Chain Monte Carlo (MCMC) based methods [3], [4] or variational methods [5]; (ii) the other more general and broader statistical category involves obtaining frequentist estimates of uncertainty, minimizing KL divergences of distributions of in-domain and out-of-domain samples, and using adversarial samples to build uncertainty estimates.

We summarize the recent deep neural networks methods of uncertainty estimation below:

a) Dropout as a Bayesian Approximation - MC Dropout: [6] This work formulates dropout, the regularization technique in deep learning, as approximate Bayesian inference. The paper demonstrates how training a neural network with dropout is equivalent to doing approximate variational inference in a probabilistic deep Gaussian process. Hence, when dealing with the predictive distribution, a prior distribution can be imposed over the weights so that performing several forward passes through the network and averaging them will be the same as doing Monte Carlo integration to find the expected output value of the model under the predictive distribution. Successful implementation requires ensembling on the network leading to additional computational expense, unlike our proposed framework..

b) Bayesian Deep Learning for Computer Vision: [7] This work, a follow-up to [6], discusses two kinds of uncertainties, epistemic and aleatoric. The authors demonstrate how both uncertainty estimates can be obtained from the same model, where MC Dropout is used for epistemic uncertainty; the model itself predicts a variance term used to handle aleatoric uncertainty of each input. They apply their model to real-world image semantic segmentation problems. Although successful, this model is designed strictly for CNN based problems, whereas the framework we propose here generalizes to any feature extraction network as shown in this work.

c) Weight Uncertainty in Neural Networks - BayesBackprop: [8] This work learns a distribution over neural network weights and applies the reparameterization trick to get a variational approximation to the distribution over weights as opposed to distribution over hidden units as done in VAE papers. They used a scaled Gaussian mixture as the prior and a diagonal Gaussian posterior distribution.

Other Bayesian Regression neural networks include [9]and [10] utilizing a similar methodology to produce uncertainty. As before, this requires storing and optimizing a distribution for each parameter, and thus is computationally expensive. Additionally, to determine a hypotheses and uncertainty, one must sample the network utilizing techniques such as variational inference [5] or MCMC [4].

We are performing ordinary regression on the distribution parameters, where each of our weights and biases take on a single value. Thus, we do not need to sample from our network. This enables us to use our method on large scale datasets of varying structures.

d) Separate Regressor: [11] One popular method for quantifying uncertainty is to regress directly on the uncertainty. Typically, two regressors are utilized: a *value* regressor and an *uncertainty* regressor. These work separately to predict their respective values. This method requires the uncertainty regressor to learn the specifications of the value regressor. Additionally, the training schedule must be carefully designed to ensure that both regressors learn in tandem. Furthermore, it is much easier (due to the complexity of simultaneously optimizing two systems) for this system to get stuck in a local optimum.

In our proposed approach we utilize a single network, thereby allowing for various components of the value regressor to interact with components of the uncertainty regressor (and vice versa). This reduces the computational overhead introduced by having two separate regressors. Furthermore, as we are only training a single network, our training schedule is significantly less complex.

e) Deep Ensemble Methods: [12] A departure from the usual Bayesian modeling, there has been research into using deep ensembles (an ensemble of deep learners) to create multiple hypotheses and uncertainty can be inferred from these hypotheses. While extremely promising, utilizing this method requires one to train multiple deep learners and evaluate multiple deep networks to generate uncertainty resulting in a fairly computationally expensive process. We avoid these issues by utilizing a single regressor. Since we are only utilizing a single regressor, we only need to train, evaluate, and store one regressor.

III. Methodology

A. Framing the Problem

Suppose we have samples $\{(x_i, y_i)\}_{i=1}^{N} \sim \mathcal{D}(X, Y)$ where $\mathcal{D}(X, Y)$ is a joint probability distribution of X and Y. For this paper, we assume that

$$\mathcal{D}(X, Y)_{X=x_0} = \mathcal{N}(\mu_{x_0}, \Sigma_{x_0}). \tag{1}$$

That is to say that each cross section of the joint probability distribution function (PDF) degenerates into a normal distribution. Furthermore, we assume that each output dimension is conditionally independent of each other; thus, for all x_0, Σ_{x_0} is a diagonal matrix.

We wish to learn a mapping from X to means and standard deviations, and by utilizing this mapping, we can determine the uncertainty (both epistemic and aleatory) of our model. We demonstrate the capability of capturing both of these uncertainties in the experiments section. As our target distributions are Gaussian, by estimating the distribution on each target variable, we can generate the confidence intervals on that target variable. Achieving such a mapping results in uncertainty quantification as described above.

B. Approach

To learn the mapping described in Section III-A, we train a regressor to output the parameters of our target distribution with the following log-likelihood loss:

$$\mathcal{L} = -\iint_S \log(\rho_X(Y))p(X, Y)dS \tag{2}$$

where $p(X, Y)$ is the true joint distribution on X, Y and ρ_x is a Gaussian induced with parameters from the regressor. A scheme which is optimal under this loss will also have a minimal mean squared error on the target data points.

This loss under finite data degenerates into NLL loss:

$$\mathcal{L} = -\sum_{x \in X}\sum_{y \in Y} \log(\rho_x(y))f(x, y) = -\sum_{i=1}^{N} \log(\rho_{x_i}(y_i)) \tag{3}$$

where $f(x, y)$ is the frequency with which (x, y) occurs in the dataset. As our target distribution is a Gaussian,

$$\begin{aligned}\mathcal{L} &= -\sum_{i=1}^{N} \log\left(\frac{1}{\sqrt{2\pi\sigma_{x_i}^2}} e^{-\frac{(y_i-\mu_{x_i})^2}{2\sigma_{x_i}^2}}\right) \\ &= \frac{1}{2}\sum_{i=1}^{N}\left(\left(\frac{y_i - \mu_{x_i}}{\sigma_{x_i}}\right)^2 + \log(2\pi) + 2\log(\sigma_{x_i})\right)\end{aligned} \tag{4}$$

which will reach its minimum when

$$\mathcal{L}^* = \sum_{i=1}^{N}\left(\frac{y_i - \mu_{x_i}}{\sigma_{x_i}}\right)^2 + 2\sum_{i=1}^{N}\log(\sigma_{x_i}) \tag{5}$$

reaches its minimum. This loss is preferable over a multitude of other losses (such as KL Divergence) as it does not require defining an auxiliary ground truth probability distribution.

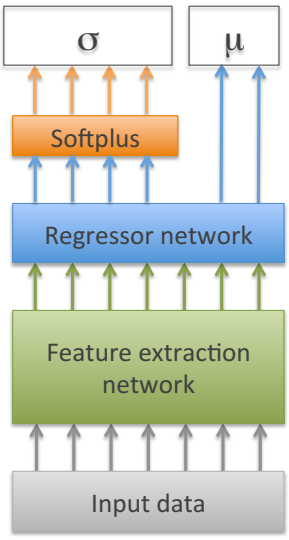


Fig. 2. General architecture for the regressor.

C. Network Architecture

We model this regressor as a multi-component neural network, which must output two values (assuming we are regressing on a single target variable): the mean and standard deviation (which can also be interpreted as an uncertainty). Figure 2 shows an overview of the generalized architecture with the various components. When applied on high-dimensional numerical input data, the feature extractor can be implemented as a deep neural network which embeds the data into a lower dimensional space; when applied on time series data, the feature extractor can be implemented as some variant of a recursive neural network (RNN); and when applied on complex natural images, a convolutional neural network (CNN) can used for feature extraction.

We typically utilize two fully connected layers as the regressor layer, where the number of nodes is determined by the complexity of the output of the feature extraction layer. The regressor network produces the mean and the standard deviation, and the function `Softplus` ($f(z) = ln(1+e^z)$) is applied to the standard deviation to ensure a valid probability distribution is generated. We demonstrate the efficacy of the architecture on different data types in Section IV

IV. Experiments and Results

To evaluate the proposed method, we utilized it in a variety of experiments described in the ensuing subsections. We applied the method to a basic, well-known 2-dimensional caloric dataset from Kaggle; to the standard benchmark datasets commonly used to measure the quality of a regression algorithm; to highly volatile, stock prices using data from 2015 to mid-year 2018; to video sequences to detect anomalies in video segments and lastly, to the large-scale image dataset, the IMDb-Wiki data for age estimation.

A. Caloric Dataset

As an initial test of our framework, we performed a one dimensional regression utilizing one parameter, on a "toy" dataset obtained from Kaggle (`Exercise and Calories`). This dataset was used purely as a proof-of-concept and for illustrative purposes. The goal was to

estimate how many calories an individual burned based on body heat. We artificially added a small amount of noise to discourage the network from memorizing the mean and standard deviations of each input.

In Figure 3, we observe that this network successfully converged to perform the distribution regression, thus demonstrating that this network (at least for this problem) can capture aleatory uncertainty (the randomness inherent in the system).

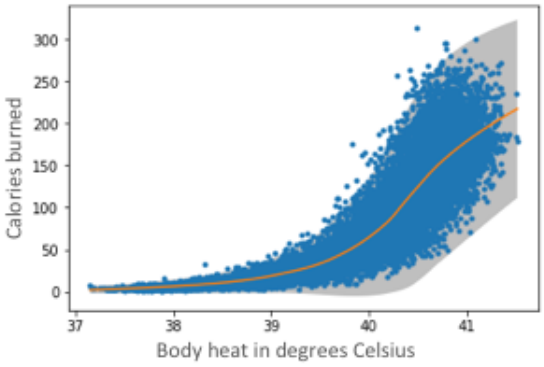


Fig. 3. Regression as described in section IV-A. The orange line is the regression and the gray shaded region is the 3σ confidence interval

B. Regression on benchmark datasets

We applied the method to the standard benchmark datasets commonly used to measure the quality of a regression algorithm and compared with the state-of-the-art techniques described in Section II. As can be observed in Table I, with the exception of the Yacht dataset where our technique was under-par, we performed on-par with or out-performed all other approaches in terms of NLL (negative log-loss), thus demonstrating that our proposed method retained its regression abilities while adding on the ability to also quantify the uncertainties associated with its predictions.

TABLE I
COMPARISON OF DIFFERENT ARCHITECTURES PERFORMANCE FOR NLL ON POPULAR BENCHMARK DATASETS. MEASUREMENTS COURTESY OF DEEP ENSEMBLES PAPER BY LAKSHMINARAYANAN ET AL. [9].

Dataset	[10] PBP	[6] MC-Dropout	[12] Deep Ensembles	Ours
Boston	2.57 ± 0.09	2.46 ± 0.25	2.41 ± 0.25	**2.23 ± 0.05**
Concrete	3.16 ± 0.02	**3.04 ± 0.09**	**3.06 ± 0.18**	**3.05 ± 0.04**
Energy	2.04 ± 0.02	1.99 ± 0.09	**1.38 ± 0.22**	1.91 ± 0.02
Kin8nm	-0.90 ± 0.01	-0.95 ± 0.03	**-1.20 ± 0.02**	**-1.18 ± 0.02**
Naval-propulsion	-3.73 ± 0.01	-3.80 ± 0.05	**-5.63 ± 0.05**	-3.82 ± 0.09
Power plant	**2.84** ± 0.01	**2.80 ± 0.05**	**2.79 ± 0.04**	**2.85 ± 0.01**
Protein	2.97 ± 0.00	2.89 ± 0.01	2.83 ± 0.02	**2.14 ± 0.01**
Wine	0.97 ± 0.01	0.93 ± 0.06	0.94 ± 0.12	**0.87 ± 0.02**
Yacht	1.63 ± 0.02	1.55 ± 0.12	**1.18 ± 0.21**	4.06 ± 0.00
MSD	3.60 ± NA	v3.59 ± NA	3.35 ± NA	3.40 ± NA

C. Uncertainty measures on stock prices

To test for uncertainty in predictions in large, complex, stochastic, highly volatile *time series data*, we applied the methodology specifically on similar stocks from the entertainment industry[1]. The family of stocks we evaluated comprised of stocks from 21^{st} *Century Fox, Inc. (FOX), Netflix, Inc. (NFLX), Time Warner, Inc. (TWX), Amazon.com, Inc. (AMZN), Walt Disney Co. (DIS), Comcast Corporation (CMCSA)*. Stocks are classified as a family based on their sector, industry, asset class and prices being highly correlated with each other over an extended period of time.

Suppose we are given the stock close prices for n days prior to day T : $\{x_t\}_{T-n-1}^{T-1}$. We wish to predict the closing price on day T: x_T. To do this, we predict to prescribe a distribution onto $x_T \sim \mathcal{N}(\mu_T, \sigma_T)$.

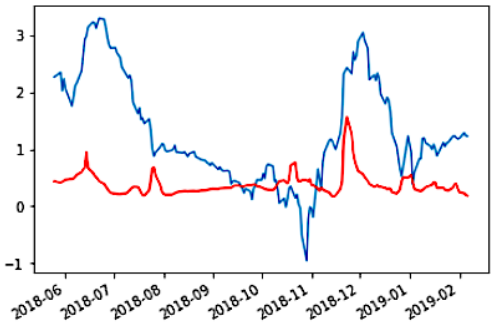


Fig. 4. The blue graph is the stock price chart for *FOX* while the red graph is the measure of uncertainty estimated by the proposed network. Image is best viewed in color

Data preparation, training schedule and results: We downloaded the publicly available stock price information for the family of stocks explained above. Data for the entire family from 2015 till May 2018 was used as training data, with the goal of predicting uncertainty for only the *FOX* stocks from June 2018 till February 2019.

The network shown in Figure 2 was implemented with the feature extraction layer being a gated recurrent unit (GRU) with a look-back of 50 days. The training scheme involved looking at the stock prices over a period of 50 days with the goal of predicting price on the 51st day along with the measure of uncertainty of the prediction. The resulting uncertainties are shown in Figure 4.

To analyze the uncertainties resulting from the implementation, we set a threshold of 0.5 so that days on which the uncertainty measure was above this threshold were flagged as anomalous trading days. We provide a list of *FOX*-related news [2] in that period and compare with the anomalous days predicted by our network. The results are shown in Table II, demonstrating that the proposed methodology successfully picked up on anomalies in the stock market, by examining the uncertainties in the network predictions.

D. Anomaly detection in video segments

In this experiment, we were specifically interested in detecting anomalous behavior from surveillance videos. Although our problem is largely unsupervised, we prescribe the following supervised task: Given $X_{t-k-1:t-1}$, determine $(\mu, \Sigma) \in \Omega$ such that $P(X_{t+P}|\mathcal{N}(\mu, \Sigma))$ is maximized. Note that this problem is identical to the previous stock prediction problem.

[1]One of the authors spent his summer internship at a financial organization and specifically analyzed this family of stocks.

[2]News data was obtained from https://www.reuters.com/finance/stocks/FOX/key-developments. We threw away many other events leaving those related to where our uncertanties were high

TABLE II

THE LEFT COLUMN SHOWS TRUE DATES ON WHICH MAJOR EVENTS OCCURRED AT 21ST CENTURY FOX; THE SECOND COLUMN SHOWS THE CLOSEST DATE ESTIMATED BY OUR NETWORK AND THE LAST COLUMN DESCRIBES THE EVENT IN THE NEWS.

Real Date	Network predictions	News related to 21st Century FOX
05-17-18	05-31-18	Suzanne Scott named CEO Of *FOX* News
06-13-18	06-15-18	Comcast offers to buy 21st Century Fox media assets for $65B in cash
10-19 till	10-19 till	Walt Disney receives unconditional approval
10-20-18	10-22-18	from China For 21st Century Fox deal; Amazon/Blackstone bid for Disney's 22 regional sports networks;
11-26-18	11-26-18	Disney, Fox sued in U.S. for $1B over Malaysia theme park
01-07-19	—	21St Century Fox announces filing of registration statement on Form 10 for Fox

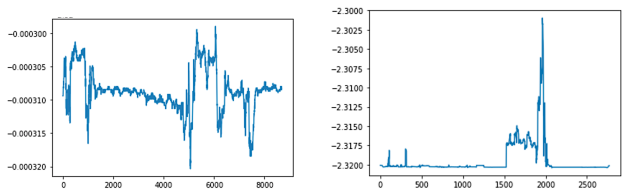


Fig. 5. Normal (top) and anomalous (bottom) data; where the x-axis is the frame number and y-axis is the loss for each frame. Note the scale of the graphs

Recall that we have a collection of normal time series X. In addition, we have a time series sampled from an unknown process. We wish to determine if this sample is anomalous. Thus, we want to quantify the extent to which this sample could have feasibly come from the process P.

We therefore utilized our Gaussian network framework to develop an auto regressive model over the signals in X. We applied the said model to Y and calculate $\frac{1}{N-P-k-1}\sum_t \mathcal{L}^*(Y_{t-k-1:t-1}, Y_{t+P})$. This loss on the entire time series will be large if our regressive model expected a fairly deterministic behavior but was "surprised". We define *surprise* as when the network expected a specific event with relatively high probability (and low uncertainty) but something completely different happens.

Data preparation, training schedule and results: We utilized the UCF-Crime dataset [22], which comprised of 1900 files totaling 128 hours of video. These videos were downsampled to a resolution of 128×128.

We trained a CNN-LSTM-CNN network on the problem illustrated above (predicting frame $t+P$ given frames $t-k-1$ to $t-1$). We assumed for simplicity that each pixel was independent of every other pixel (i.e. we constrained our covariance matrix to be a diagonal matrix). Our network took as input a video stream and output two images: μ, σ. We utilized the NLL loss to train this network.

E. Age estimation from face images

To test how well this architecture works on large complex datasets, we applied it on the nontrivial problem of age estimation. Given an image of a face, the network was tasked with predicting the age of the individual in the picture. Posed as a general problem, this task is a challenging regression problem.

We utilized the IMDb-Wiki Dataset [13]: a dataset of half a million faces scraped from both IMDb and Wikipedia (primarily IMDb), and tagged with the corresponding ages of individuals in the images. The dataset is very noisy, where multiple entries contain either no face or multiple faces. Also, in some cases, the collection year was incorrectly extracted from the webpage.

Although this dataset contained a similar distribution of males to females it contained primarily individuals between 20 and 40 years old. Additionally, because the IMDb dataset contained a random sampling of Hollywood actors, it was primarily composed of young Caucasian individuals, thus, having high implicit bias. We empirically demonstrated that our method was still capable of correctly identifying under-represented samples in spite of the imbalances in the data.

Data preparation, training schedule and results: We did not wish to excessively clean the data, but rather remove the clearly wrong data. We did this by removing samples which had individuals younger than three years old or older than 100 years old. Additionally, we removed images which were smaller than 16×16 and then resized the remaining images to 224×224. We did not remove any other data.

This network was trained using the pre-trained VGG16 CNN system [14] for feature extraction, utilizing an Adam optimizer with a learning rate of 0.00005 and a batch size of 8, for six epochs.

TABLE III

THE ACCURACY (BOTH MEAN ABSOLUTE ERROR AND NEGATIVE LOG LIKELIHOOD) OF VARIOUS APPROACHES ON AGE ESTIMATION.

Method	MAE	NLL
CNN + Regressor	7.54	
CNN + Regressor + Uncertainty	7.57	3.63
CNN + Regressor + Uncertainty + Cleaning	5.22	3.53

The results of these experiments were very promising (first row of Figure 1). Even on this noisy dataset, the architecture only performed poorly when the ground truth was wrong (see the middle row of Figure 1). These results demonstrate that our model is capable of capturing epistemic uncertainty. Additionally, this architecture's uncertainty not only expressed how confident the model was, but also how clean the data sample was. Thus, this model often reported high confidence if a sample was well represented within the dataset. Empirically we can see that difficult samples (those in a class with low representation, poor lighting, side facing faces, ambiguous individual, multiple faces) obtained high uncertainty. Images in the dataset which were incorrectly scraped along with excessively noisy or incorrect data had the highest uncertainty (see Figure **??** and Figure 6 below). Additional predictions on face images are shown at the end of the manuscript. This architecture can therefore be used to evaluate the quality of samples, assuming a large portion of the data is of good quality.

1) Determining the overhead of uncertainty quantification: An error quantification network is often only appealing if it does not have a significant impact on performance. Thus, the quantification of the discrepancy of some error metric (say RMSE) between a classical regressor with ω parameters and that of an uncertainty-aware regressor with ω parameters should be minimized. To this end, we trained two networks utilizing the same initial configuration of parameters and same number of parameters (except for the

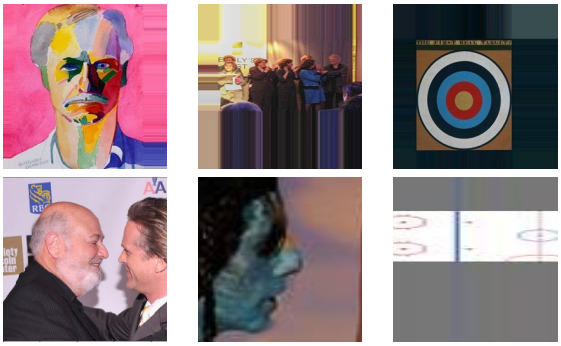


Fig. 6. Examples of invalid face data from the IMDb-Wiki dataset (top 5% uncertainty).

last layer) until convergence (one vanilla regressor and one error quantification regressor).

Examining Table III, we can observe that the discrepancy between the MAE of the uncertainty-agnostic regressor and the uncertainty-aware regressor is negligible. Thus, computing the uncertainty does not provide any significant additional overhead to this model. This final layer can therefore be added to *any* regressor to provide uncertainty metrics.

2) Automated data cleaning: As described earlier, this architecture can be utilized to determine the quality of a sample by examining the uncertainty produced. After training, we identified the samples with the top 5% uncertainty and removed them (note that we left the validation samples unchanged). After removing these samples from our training set, we obtained significantly better results on the validation dataset (see the last row of Table III). Thus, this architecture is uniquely well-suited for unclean datasets, and to improve the performance of regressors.

V. Conclusion and Future Work

We have presented a framework, which can be used for any regression problem to indicate how uncertain the network is about its predictions. In this work, we replace the RMSE loss with one based on NLL which measures the probabilistic distance between predicted y' and the true y, where $y' = \mu_{x_i}$. Imagine placing a spherical Gaussian centered at each prediction μ at the start of training and as training progresses, σ is also updated by side effect. The final σ is the distributional uncertainty of the prediction being made at μ. Hence, by choosing a probabilistic loss function such as NLL, we can tell how confident the network is, of its predictions, with no extra computation.

We have shown this method of uncertainty quantification to work well with large-scale and/or complex datasets, without any additional overhead during training. This uncertainty quantification aspect has been exploited to develop a data cleaning procedure which improved the accuracy on an unchanged validation set. Future work includes generalizing this to arbitrary distribution regression and investigating uncertainty for classification. Furthermore, we intend to continue studying the data cleaning process to determine how and when it can be utilized to boost performance on noisy, real-life datasets.

Tables IV–VII in the supplementary materials show additional results from the age estimation problem.

References

[1] C. Leibig, V. Allken, P. Berens, and S. Wahl, "Leveraging uncertainty information from deep neural networks for disease detection," *Scientific Reports*, vol. 7, 2017.

[2] C. Bishop, "Mixture density networks," Tech. Rep., January 1994. [Online]. Available: https://www.microsoft.com/en-us/research/publication/mixture-density-networks/

[3] D. J. C. MacKay, "A practical bayesian framework for backpropagation networks," *Neural Computation*, vol. 4, no. 3, pp. 448–472, May 1992.

[4] A. Vehtari, S. Sarkka, and J. Lampinen, "On mcmc sampling in bayesian mlp neural networks," *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, 2000.

[5] A. Graves, "Practical variational inference for neural networks," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2011, pp. 2348–2356.

[6] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *International Conference on International Conference on Machine Learning (ICML)*, 2016, pp. 1050–1059. [Online]. Available: http://dl.acm.org/citation.cfm?id=3045390.3045502

[7] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 5574–5584.

[8] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," in *International Conference on Machine Learning (ICML)*, 2015, pp. 1613–1622.

[9] A. Korattikara Balan, V. Rathod, K. P. Murphy, and M. Welling, "Bayesian dark knowledge," in *Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, Inc., 2015, pp. 3438–3446. [Online]. Available: http://papers.nips.cc/paper/5965-bayesian-dark-knowledge.pdf

[10] J. M. Hernández-Lobato and R. P. Adams, "Probabilistic backpropagation for scalable learning of bayesian neural networks," in *International Conference on International Conference on Machine Learning*, ser. ICML'15, 2015.

[11] D. Nix and A. Weigend, "Estimating the mean and variance of the target probability distribution," *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN94)*, 1994.

[12] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, Inc., 2017, pp. 6402–6413.

[13] R. Rothe, R. Timofte, and L. V. Gool, "Dex: Deep expectation of apparent age from a single image," *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, 2015.

[14] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[15] H. M. D. Kabir, A. Khosravi, M. A. Hosen, and S. Nahavandi, "Neural network-based uncertainty quantification: A survey of methodologies and applications," *IEEE Access*, vol. 6, p. 36218–36234, 2018.

[16] A. D. Kiureghian and O. Ditlevsen, "Aleatory or epistemic? does it matter?" *Structural Safety*, vol. 31, no. 2, p. 105–112, 2009.

[17] A. Damianou and N. Lawrence, "Deep gaussian processes," in *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, ser. AISTATS, vol. 31. PMLR, May 2013, pp. 207–215.

[18] J. Deng, W. Dong, R. Socher, L. jia Li, K. Li, and L. Fei-fei, "ImageNet: A large-scale hierarchical image database," in *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[19] D. Harrison and D. L. Rubinfeld, "Hedonic housing prices and the demand for clean air," *Journal of Environmental Economics and Management*, vol. 5, no. 1, p. 81–102, 1978.

[20] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

[21] M. Mathias, R. Benenson, M. Pedersoli, and L. Van Gool, "Face detection without bells and whistles," in *European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2014, pp. 720–735.

[22] W. Sultani, C. Chen, and M. Shah, "Real-world anomaly detection in surveillance videos," in *IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR, 2018, pp. 6479–6488.

TABLE IV

COLLECTION OF FACES WHICH THIS ARCHITECTURE SCORED WELL ON ACCORDING TO NLL (SMALL LOSSES).

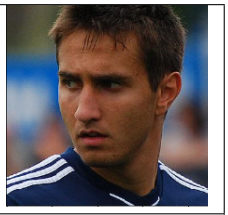
Image					
Actual Age	20	21	24	23	22
Predicted Age	19.6	22.0	22.3	23.8	22.8
Uncertainty	3.1	3.2	7.5	7.8	3.6
Loss	2.07	2.12	2.97	2.98	2.27

TABLE V

COLLECTION OF FACES WHICH SCORED THE WORST ACCORDING TO NLL (HIGH LOSS). WE NOTE THAT FOR THIS ARCHITECTURE TO SCORE POORLY ON A DATAPOINT, IT MUST BE RELATIVELY CERTAIN OF AN INCORRECT VALUE.

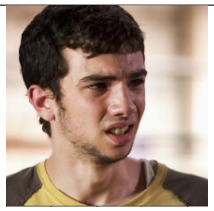
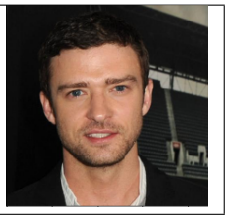
Image					
Actual Age	92	82	67	70	82
Predicted Age	26.1	30.8	27.1	25.6	35.3
Uncertainty	7.9	6.4	5.9	6.6	7.2
Loss	37.9	35.0	25.7	25.2	23.7

TABLE VI

COLLECTION OF FACES WHICH THIS ARCHITECTURE WAS MOST CONFIDENT ABOUT (LOW UNCERTAINTY).

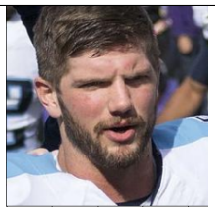
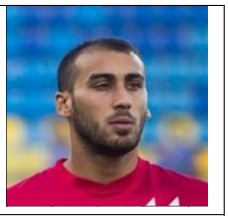
Image					
Actual Age	24	24	20	23	22
Predicted Age	21.0	21.6	19.6	19.2	18.4
Uncertainty	3.2	3.2	3.1	3.2	3.0
Loss					

TABLE VII

COLLECTION OF THE FACES FOR WHICH THIS ARCHITECTURE WAS NOT CONFIDENT (HIGH UNCERTAINTIES), EVEN IF THE LOSES WERE LOW.

Image					
Actual Age	36	58	27	69	88
Predicted Age	55.3	62.0	69.6	76.4	83.9
Uncertainty	19.0	19.1	21.6	20.5	20.8
Loss	4.38	3.89	5.94	4.00	3.97