Data Opinion Propagation in Deep Neural Networks

***Abstract*—In this paper, we propose a method to incorporate the uncertainty associated with the input data of a deep neural network and propagate this uncertainty throughout the training phase using subjective logic. We represent intrinsic statistical properties of input data as subjective opinions and through the propagation of input data opinions, we quantify the total opinion of the learning model that works as a trust metric for the outcome of the DNN. In our experiments, we demonstrate that statistical belief and uncertainty associated with the input data can impact the overall trustworthiness of a DNN model. We observe that belief and projected probability of the final opinion of the model is increasing while uncertainty is decreasing.**

***Index Terms*—Deep Neural Network; Subjective Logic; Opinions; Uncertainty; Trustworthiness; Machine Learning;**

I. Introduction

AI technologies are contributing to a wide range of decision-making processes, from driving a car, cancer diagnosis [1], [2] and even providing legal advice [3]. Although the level of autonomy for an AI-based solution might be different in each usage scenario, the trend is increasingly towards human-independent, algorithmic decision-making. Since machine learning (ML) algorithms are at the core of AI systems, in such scenarios, a significant challenge is answering the question of how much trust we can put into the outcome of the ML model. An important factor contributing to the trustworthiness of ML models is uncertainty associated with input data. This uncertainty can be originated from the intrinsic properties of available data (e.g., missing data, imbalanced data, skewness) or even a subjective opinion expressed by a modeler. In either case, the trust question can be answered only if the modeler is able to take into account the opinion about the data in the training phase of an ML model and report the outcome of the model not only in terms of the model performance but also, in terms of the trustworthiness of the model.

In this paper, we propose a method of quantifying and propagating input data opinions throughout the training phase of a deep neural network. As of the state-of-the-art opinion propagation model proposed in [1], we are also inspired by the probabilistic logic framework defined as Subjective Logic (SL) [2]. We first investigate the limitation of the opinion quantification and propagation method proposed in [2]. We introduce the concept of *interpretable opinions* that allows to capture the intrinsic statistical properties of input data as opinion determinants. To initialize input data opinions, we partition input data into numerical and categorical features and use statistical properties such as skewness and kurtosis. Then we extended the propagation method to be aligned with the newly introduced interpretable opinions. Moreover, we identified a shortcoming in the current method when computing the error opinion, the difference between predicted and actual outcomes (the error) of the DNN model is compared with a single threshold to form the error opinion missing to take into account uncertain evidence about the error. The experimental evaluation of the proposed method demonstrates that during the training process, the trustworthiness of the model is improving while the uncertainty is decreasing according to the different data opinion initialization factors.

We are making the following contributions: 1) we demonstrate the limitations of the current trust quantification model, 2) we extend the state-of-the-art propagation model to take into account input data opinions determinants (i.e., belief, disbelief, and uncertainty) and also enhance the opinion optimization process by incorporating uncertain evidence, and 3) we conduct an extensive experiment using real-world datasets, to demonstrate the effectiveness of our approach.

II. Current Opinion Propagation Models

The following section provides the necessary preliminaries on the background of uncertainty and subjective logic as an approach leveraged to propagate data opinions in a DNN. Then, we provide subjective logic concepts and a summary of the current method of opinion propagation proposed by [1]. Finally, we propose our method for the data opinion initialization and optimization process.

A. Preliminaries and Background

Uncertainty. In the DNN classification task (see Appendix A), the outputs of the model are the softmax probabilities $p_{\hat{y}} \in P_{\hat{Y}}$, and are considered as the confidence of the DNN model in selecting the target labels for the training data $\mathcal{D}$ [3], [4].

The total uncertainty of a DNN model is generally defined as the lack of confidence in classifying each data sample $d \in \mathcal{D}$. This uncertainty denoted by U_d, is calculated using the entropy of the probability vector $P_{\hat{Y}}$ as,

$$U_d(P_{\hat{Y}}) = -\sum_{\hat{y} \in \hat{Y}} p_{\hat{y}} \log p_{\hat{y}} \ . \tag{1}$$

There are two types of uncertainty that can be used in deep learning: first-order and second-order uncertainty. U_d is a first-order type uncertainty, quantified by the first-order probability of softmax function [5], [6]. However, first-order probabilistic logic is limited in capturing ownership of the assigned or computed probability. For example, the probability extracted from softmax function may change, when the modeler or any other sources in a DNN (e.g., a neuron) has a certain opinion about the model outcome probability (or model performance) generated by the softmax function. To capture this notion, second-order uncertainty is defined as the probability that the first-order uncertainty has a certain value. This value can be interpreted as the value that would be assigned if certain information about the first-order probability were available, including opinions of a modeler or any other sources. In addition, we may encounter situations where the owner or the source cannot produce probability for the first-order uncertainty or where multiple owners or sources have different opinions about the uncertainty.

Subjective Logic. Subjective Logic extends probabilistic logic by including uncertainty about first-order probabilities and subjective belief ownership [2]. Subjective logic utilizes arguments called *Opinion* expressing separate beliefs about the truth of statements under an uncertainty mass as a second-order uncertainty. Although uncertainty generally means the lack of confidence, in subjective logic, uncertainty is the vacuity of evidence. This opinion representation is equivalent to the Dirichlet distribution and shows ownership of an opinion. Furthermore, subjective opinion ownership is closely related to trust since when there are different opinions about the same first-order probability, then a source can determine trust levels in other different sources of opinions before being integrated in a reasoning model [2].

B. Current Method of Opinion Quantification

We use the following definitions from subjective logic to capture second-order uncertainty to design our data opinion quantification model.

Definition 1 (DNN Component). In a DNN, an input data feature, a model parameter (weight or bias), and/or a neuron are considered as DNN components.

Definition 2 (Binomial Opinion). An opinion about a subjective variable x in a binary domain $\mathbb{X} = \{x, \overline{x}\}$ is a *Binomial Opinion* owned by a DNN component. A binomial opinion [2] by the DNN component v about the model performance as the subjective variable, is defined as a tuple of four determinants,

$$O_v = \{b_v, d_v, u_v, a_v\} \quad \text{s.t.} \quad b_v, d_v, u_v, a_v \in \mathbb{R}^{[0,1]} \ , \tag{2}$$

where b_v denotes the amount of belief in support of the subjective variable x, d_v denotes the amount of disbelief about x, the uncertainty mass u_v represents the lack of confidence about x caused by vacuity of evidence in a DNN model, and a_v is the prior probability about x without considering any evidence which can take a prior bias measure into account (see Appendix B for more details on opinion determinants).

Furthermore, these opinion masses should meet the following additivity requirement:

$$b_v + d_v + u_v = 1 \ . \tag{3}$$

A binomial opinion by the DNN component v is equivalent to a Beta PDF which is a particular binary version of Dirichlet PDF [2]. The expected probability of the PDF yields to a projected probability as *trustworthiness* of a DNN component which considers the component's belief and uncertainty about model performance simultaneously as,

$$\text{Projected Probability:} \quad PP_v = b_v + u_v.a_v \ . \tag{4}$$

Since DNN components impact the model performance in each training epoch, following subjective logic, each DNN component v has a binomial opinion about the DNN model performance. Furthermore, there are binary [2] or multi-source operators [7] defined in subjective logic like Multiplication ($\otimes$), Comultiplication ($\widehat{\otimes}$), Average Fusion ($\oplus$), Cumulative Fusion ($\widehat{\oplus}$), and Multi-source Average Fusion ($\bigoplus$), to calculate the output opinion achieved based on a specific relation amongst two or more opinions as operands.

Based on the the opinion propagation method in [1], for each feature of each data point $d \in \mathcal{D}$ such that $d = \{f_1, f_2, ..., f_n\}$ with the true label $y \in Y$, the data feature opinions are initialized by given and equal opinions $O_d = \{O_{f_j} \mid j \in \mathbb{N}^{[1,n]}\}$ as feature opinion set and O_y as true label opinion. These data opinions are propagated through the DNN similar to the data forward propagation. In each layer i, the forward propagation is fulfilled by the multiplication of the last layer neurons' opinions and the corresponding weight opinions denoted by O_{w_i}. For the first hidden layer, the multiplication is done between weight opinions and data opinions. Thus, each neuron's opinion is dependent to the opinion of weights and the previous layer's neurons. Then, in the layer i, the opinion of each neuron denoted by O_{N_i}, is achieved by combining these multiplications together along with the bias opinions denoted by O_{b_i}:

$$O_{N_1} = \bigoplus(O_D \otimes O_{w_1}, O_{b_1}) \ , \tag{5}$$

$$O_{N_i} = \bigoplus(O_{N_{i-1}} \otimes O_{w_i}, O_{b_i}) \ . \tag{6}$$

Continuing the forward propagation, the forward opinion of the output neuron l denoted by $O^l_{\hat{y}}$ is achieved. Moreover, each output neuron l has a true label opinion denoted by O^l_y and a backward opinion denoted by O^l_{back} achieved by backward propagation of the opinions. To compute the backward opinion, the error opinion O^l_δ of the output neuron l is computed based on a single threshold ϕ. Comparing the error $|y - \hat{y}|$ with the threshold ϕ, the positive or negative evidence are counted to form the belief and disbelief of error opinion, respectively. Then, the backward opinion of the output neuron l is obtained by the multiplication of the target label opinion O^l_y and the error opinion O^l_δ:

$$O^l_{back} = O^l_y \otimes O^l_\delta \ . \tag{7}$$

Then, the opinion of each output neuron l, denoted by O^l_{out}, is computed as,

$$O^l_{out} = O^l_{\hat{y}} \oplus O^l_{back} \ , \tag{8}$$

where $O^l_{\hat{y}}$ and O^l_{back} are forward opinion and backward opinion respectively.

Finally, the opinion of the DNN model $\mathcal{M}$ denoted by $O_{\mathcal{M}}$, is obtained by applying average fusion on all output neurons' opinions:

$$O_{\mathcal{M}} = \bigoplus_{l} (O^l_{out}) \ . \tag{9}$$

Limitations. The opinion propagation method proposed in [1] suffers from some limitations:
1) The input data opinions are initialized by arbitrary values. For example, all data opinions are initialized either as Max Belief ($b = 1$), Max Uncertainty ($u = 1$), Neutral Opinions ($b = d = u = \frac{1}{3}$), or Equal Belief and Disbelief ($b = d = 0.5$). This limitation prevents capturing opinions that are originated from diverse intrinsic properties of input data features (e.g., skewness or kurtosis of a feature distribution).
2) The assigned opinion tuples can only have different values when the stochastic training process is applied. When a mini-batch training process is applied, the opinion tuples are forced to have the same values, i.e., $O_D = \{O_{f_j} \mid \forall j \neq k : O_{f_j} = O_{f_k}\}$. This limitation is originated in the propagation method described in [1], as no algorithm is provided for aggregating opinions in one mini-batch.
3) Following Equation 7, error opinion O^l_δ can be computed only based on positive and negative evidence (comparing the error with a single threshold to decide negativity or positivity), while we may encounter conditions where the evidence observed based on $|y - \hat{y}|$ is itself uncertain. Thus the evidence needs to be evaluated not only as positive or negative but also as uncertain.

III. Proposed Opinion Propagation Model

The input dataset has the feature set $F = \{f_s \mid s \in \mathbb{N}^{[1,n_f]}\}$ and a set of true labels $L = \{l_s \mid s \in \mathbb{N}^{[1,n_l]}\}$ where n_f and n_l are the total number of features and the total number of labels, respectively. We also define a new concept as *Interpretable Opinions* for input data. Interpretable opinions are opinions that are rooted in the statistical properties of the training data set (e.g., skewness). Therefore, contrary to [1], in our model the data opinions are not required to have given arbitrary values.

In a DNN model, each DNN component has its own opinion about the model performance. For each mini-batch of training data, the proposed approach acts as an operator to generate data opinions for each data feature. We first iterate over the features to initialize each feature opinion's determinants due to being a numerical or non-numerical feature as discussed in Section III-A. Then, we can feed the model by the training data along with their interpretable opinions.

A. Data Opinion Initialization

We partition the initial dataset into two groups based on their different statistical nature: numerical and categorical/target features. Following Definition 2, we initialize the opinion determinants of each numerical feature based on two different groups of statistical and distributional factors. The selection of these factors to initialize the opinion determinants, i.e., belief, disbelief, and uncertainty, is based on their interpretations and definitions/applications, which are further discussed in the following. The selected factors are different due to the significant difference in the nature of numerical, categorical/target features. Thus, we devise a different feature opinion initialization for each type of features based on different statistical properties in the batch of training data. Therefore, there is flexibility in selecting different statistical measures to initialize interpretable data opinions. The first group includes Cronbach's Alpha, Data Quality Score (DQS), and Eigenvalue ratio, i.e., explained variance. The second group includes Skewness, Kurtosis, and the Distributional Variance. For categorical features and target labels, we initialize the opinion determinants for each feature based on DQS, Shannon entropy for uncertainty mass, and opinion additivity requirement.

Base rates for each data feature opinion are the prior probability about the model performance without considering training data points or any evidence. Since we intend to initialize the input data opinions in an unbiased manner, base rates for each feature as prior probabilities are selected randomly from a normal (Gaussian) distribution $\mathcal{N}(\mu, \varepsilon^2)$ where μ is the mean value and ε is a relatively small positive constant as the standard deviation. In this research, we select $\mu = 0.5$ and $\varepsilon = 0.05$ to randomly and equiprobably determine the base rates for belief and disbelief about model performance.

Numerical Features. The first determinant of a feature's opinion is the belief b_f for the numerical feature $f \in F$ of the training data, which is initialized by the equally-weighted average of two different factors statistically selected to indicate the belief in data as,

$$b_f = \frac{1}{2}\Big[\frac{EV_f}{\sum_{s=1}^{n_f} EV_f}.\alpha_{\mathcal{D}_n} + \exp\big(-\frac{(skew_f)^{\beta_1}}{\beta_2}\big)\Big] \ , \tag{10}$$

where EV_f is the eigenvalue used for applying explained variance of the feature f extracted from the covariance matrix of feature set, and $skew_f \in \mathbb{R}$ is the amount of skewness of the numerical feature f.

As the results of particular measures for each numerical feature f of each mini-batch of training data, e.g., skewness and kurtosis, have an unlimited range of values, we use the exponential function $func : \mathbb{R} \to \mathbb{R}^{[0,1]}$ to be applied over these statistical measures as,

$$func(z_f) = \exp\big(-\frac{(z_f)^{\beta_1}}{\beta_2}\big) \ , \tag{11}$$

where z_f is the measure of skewness or kurtosis for the feature f.

The function $func$ creates a score function from the skewness and kurtosis of f to be scaled into $\mathbb{R}^{[0,1]}$ with respect to two constants $\beta_1 \in \mathbb{R}^+$ and $\beta_2 \in \mathbb{R}^+$ as the smoothness factor and the scaling factor, respectively.

The skewness measure can show how much the distribution of f is biased in one side around the mean value, i.e., how much it is far from an ideal normal distribution without skewness. In skewed data, the tail of the distribution are considered as outliers which negatively impacts the model accuracy. As in the proposed method, the model trustworthiness is correlated to the model accuracy, the skewness can negatively impact the total model trustworthiness as well. In addition, $\alpha_{\mathcal{D}_n}$ is a metric called *Cronbach's Alpha* [8] for the numerical dataset $\mathcal{D}_n \subseteq \mathcal{D}$ of the DNN as,

$$\alpha_{\mathcal{D}_n} = \frac{n_f}{n_f - 1} \cdot \frac{\sum_{i=1}^{n_f} \sum_{j \neq i}^{n_f} \sigma_{ij}}{\sum_{i=1}^{n_f} \sum_{j=1}^{n_f} \sigma_{ij}} \ , \tag{12}$$

where σ_{ij} is the covariance between two features f_i and f_j.

This metric can measure the overall reliability and consistency among various features in a dataset and is partitioned for each feature using their eigenvalue ratios. These eigenvalue ratios are explained variance that can demonstrate the amount of information that each feature carries in a dataset.

The second determinant of a feature's opinion is the disbelief mass d_f for the numerical feature $f \in F$ of the training data and is initialized by the equally-weighted average of two different factors statistically selected to indicate the amount of disbelief in data as,

$$d_f = \frac{1}{2}[(1 - DQS_f) + sigmoid(\frac{kurt_f}{\beta_1} - \beta_2)] \ , \tag{13}$$

$$d_f = \frac{1}{2}[(1 - DQS_f) + (1 - \exp(-\frac{(kurt_f)^{\beta_1}}{\beta_2})) \ , \tag{14}$$

where DQS_f is *Data Quality Score* for the numerical feature f of the DNN. Furthermore, $kurt_f$ is the amount of kurtosis of the numerical feature f that indicates the chance of the presence of outliers in the distribution of f. In fact, this value amounts to the presence of outliers and can be interpreted as unreliability or disbelief for a distribution. Therefore, the model trustworthiness which is correlated to the model performance is again negatively influenced by the kurtosis of each data feature. Moreover, the function $sigmoid : \mathbb{R} \to \mathbb{R}^{[0,1]}$ as,

$$sigmoid(\frac{kurt_f}{\beta_1} - \beta_2) = \frac{1}{1 + \exp(-\frac{kurt_f}{\beta_1} + \beta_2)} \ , \tag{15}$$

creates a scaled function from the kurtosis of f with respect to two constants $\beta_1 \in \mathbb{R}^+$ and $\beta_2 \in \mathbb{R}^+$ as the smoothness factor and the scaling factor, respectively.

Before preprocessing the training data, the data quality score is calculated based on the rate of different data issues. We used IBM Data Quality Score[1] for this purpose. We consider missing or meaningless values (e.g., NaN) and data type mismatch as two major data issues for each feature f. Our choices of data quality issues are arbitrary and the score can incorporate more or less potential issues. The data quality can directly impact the model performance since each missing or misleading data value is considered as lack of information which can deviate the model from convergence or being effectively trained. Thus, in the proposed approach, the data quality is directly correlated to the model trustworthiness.The data quality score is calculated as follows:

$$DQS_f = \prod_{i=1}^{N(Issue_f)} (1 - R(Issue_f(i))) \ , \tag{16}$$

[1]https://www.ibm.com/docs/en/iis/11.7?topic=results-data-quality-score

where $N(Issue_f)$ is the number of data issues considered for the feature f, and $R(Issue_f(i))$ is the rate of ith data issue considered for the feature f.

The last determinant of a feature's opinion is the uncertainty u_f for the numerical feature $f \in F$. The uncertainty is the most prominent determinant since it can be considered as one of the indicators of trust in the system. The uncertainty value is initialized by the equally-weighted average of two different factors statistically selected to indicate the amount of uncertainty in data:

$$u_f = \frac{1}{2}[(1 - \frac{EV_f}{\sum_{s=1}^{n_f} EV_f}) + \sigma_f^2] \ , \tag{17}$$

where σ_f^2 is the amount of total variance of the feature f.

Since the variance of a distribution in data is interpreted as an indicator of the aleatoric uncertainty (capturing the inherent noise in the data) [9], we exploit the variance as one factor for initialization of uncertainty in the feature opinion. The remaining fraction of the explained variance or eigenvalue ratio of a feature is considered the second factor for uncertainty initialization. Finally, to satisfy the opinion requirement stated in Equation 3, we apply the softmax function over the opinion determinants already computed.

Categorical Features and Target Labels. For a categorical feature $f \in F$ of the DNN which contains n_s distinct categories and a target label $l \in L$, entropy is exploited as a measure of uncertainty [6]. We utilize Shannon entropy [10] as an entropy which is widely used in machine learning because of its specific attributes as a function. This uncertainty is initialized as,

$$u_f = -\sum_{i=1}^{n_s} \frac{N(c_i)}{m} . \log \frac{N(c_i)}{m} \ , \tag{18}$$

$$u_l = -\frac{N(l)}{m} \sum_{s=1}^{n_l} \frac{N(l_s)}{m} . \log \frac{N(l_s)}{m} \ , \tag{19}$$

where $N(c_i)$ is the number of the category c_i in the categorical feature f, and m is the number of total samples (data points) of the training data.

To initialize disbelief for categorical data, we use the following equation based on the same Data Quality Score already computed in numerical feature section as,

$$d_f = 1 - DQS_f \qquad \text{and} \qquad d_l = 0 \ . \tag{20}$$

The disbelief for the target labels can be ignored and considered zero since the true labels are constant (ground truth) and do not negatively impact the model performance.

The last determinant is the belief in categorical data features. Following the additivity requirement 3 of binomial opinions, we use the following equations to initialize the belief of the categorical features and target labels as,

$$b_f = 1 - u_f - d_f \qquad \text{and} \qquad b_l = 1 - u_l \ . \tag{21}$$

B. Parameter Opinion Initialization

According to Definition 1, the model parameters, i.e., weights and biases, are considered as DNN components in the model. Before training process, we need to initialize the opinions of the model parameters along with initializing the model parameters. Thus, we initialize the parameters' opinions using *Uncertainty Maximization* [2] according to the lack (vacuity) of observations since before starting the training process, there are no data points seen by the model. In uncertainty maximization, an opinion for the parameter θ should have at least one belief mass of zero, i.e., $b_\theta = 0$, and the uncertainty is calculated as,

$$u_\theta = \min(1, \frac{PP_\theta}{a_\theta}) \ , \tag{22}$$

where PP_θ and a_θ are the projected probability and the base rate of the parameter θ's opinion, respectively.

We exploit randomly generated opinions from the uniform distribution $\mathcal{U}(0,1)$ with respect to the opinion additivity requirement 3 to fairly and uniformly select the opinions' determinants. Then, we apply uncertainty maximization on the generated parameters' opinions to maximize the uncertainty before starting the training phase.

C. Opinion Optimization Process

In the optimization process, predicted outputs in the model are produced during training epochs such that having less difference with the actual output. In the optimization process, the DNN model is trained by minimizing the difference between predicted softmax probabilities as outputs and the actual hard-labeled outputs. The amount of absolute difference between actual and predicted outputs, i.e., $|y - \hat{y}|$, is considered as a piece of evidence to form the determinants of error opinion O_δ. We need to count the positive and negative evidence as well as considering uncertain evidence in the model output to form the error opinion. Therefore, we split the probabilistic interval $[0, 1]$ into three separate partitions based on two thresholds ϕ_1 and ϕ_2:

$$\phi_1 \in \mathbb{R}^{(0,1)} \quad \text{and} \quad \phi_2 \in \mathbb{R}^{(0,1)} \quad \text{s.t.} \quad 0 < \phi_1 \leq 0.5 \leq \phi_2 < 1$$

We compare the absolute error $|y - \hat{y}|$ with the two thresholds ϕ_1 and ϕ_2 to form the error opinion's determinants. In each batch of training data, these three partitions are used to count the number of evidence denoted by r, s, and t to form belief, disbelief, and uncertainty, respectively. Counting the number of evidence for each opinion's determinant is done by categorizing the absolute difference into $\mathbb{R}^{[0,\phi_1]}$ $(0 \leq |y - \hat{y}| \leq \phi_1)$ as positive evidence to form r, $\mathbb{R}^{[\phi_2,1]}$ $(\phi_2 \leq |y - \hat{y}| \leq 1)$ as negative evidence to form s, and $\mathbb{R}^{(\phi_1,\phi_2)}$ $(\phi_1 < |y - \hat{y}| < \phi_2)$ as uncertain evidence to form t. Here, the absolute difference is between two probability values, i.e., a softmax result and actual label one-hot encoding. Thus, the minimum and maximum of the difference will be 0 and 1, respectively. These thresholds can be specified arbitrarily with respect to the grid search, and here, we select $\phi_1 = 0.4$ and $\phi_2 = 0.7$ (see Appendix G for more details).

For each label l in output neurons, we can build the error opinion O^l_δ based on the observed evidence compared to the thresholds ϕ_1 and ϕ_2 following the equations described in [2]:

$$b^l_\delta = \frac{r}{r+s+t} \quad d^l_\delta = \frac{s}{r+s+t} \quad u^l_\delta = \frac{t}{r+s+t} \ . \tag{23}$$

We create backward opinion O^l_{back} for each label l by comultiplication between error opinion O^l_δ and true label opinion O^l_y. As it is stated in [2], the reason for using comultiplication is to consider the existence (presence) of at least one of the opinion operands in the backward process. Therefore, we intend to cover and consider the existence of either error opinion or actual opinion or both together:

$$O^l_\delta = \{b^l_\delta, d^l_\delta, u^l_\delta, a^l_\delta\} \quad \text{and} \quad O^l_y = \{b_l, d_l, u_l, a_l\} \quad \text{s.t.}$$
$$O^l_{back} = O^l_\delta \ \widehat{\otimes}\ O^l_y \ . \tag{24}$$

Then, according to Equations 8 and 9, we take the average fusion of the backward opinion O^l_{back} and forward (predicted) opinion $O^l_{\hat{y}}$ to calculate the output opinion O^l_{out} for each output neuron l. Finally, the opinion of the DNN model $\mathcal{M}$ denoted by $O_\mathcal{M}$, is obtained by applying average fusion on all output neurons' opinions. The output opinion results are aggregated by average fusion because these opinions observe each neuron simultaneously (the same process over the same period of time).

To optimize the total model opinion during training, i.e., minimizing the uncertainty while maximizing the belief, we update the opinion of each parameter θ using the parameter change opinion denoted by $O_{\Delta\theta}$ based on the error opinions. The parameter change opinion is the opinion that is aggregated to the current parameter opinion by cumulative fusion to update it.

The parameters' opinions O_θ are updated based on the error value $|y - \hat{y}|$ that the model is achieving during training. Thus, we exploit back propagation approach for the parameters' opinion updates while we are updating the actual parameters in mini-batch gradient descent to minimize the loss function. Based on what we perform in back propagation process to update the parameters in each layer, we initialize the opinions of parameters' change, $O_{\Delta w}$ and $O_{\Delta b}$, backward from the last layer towards the first layer using error opinion set O_δ and its pairwise multiplication with predicted opinion set $O_{\hat{y}}$ in the previous layer. For instance, in layer i,

$$O^i_{\Delta b} = O_\delta \qquad \text{and} \qquad O^i_{\Delta w} = O_\delta \otimes O^{i-1}_{\hat{y}} \ . \tag{25}$$

Finally, in each layer i with the neuron set $layer_i$, we update the error opinion set O_δ by assigning a set consists of multi-source average fusion on the pairwise multiplication of the previous O_δ and the weight opinion set O^i_w as follows:

$$O_\delta = \{ \bigoplus_{layer_i} (O_\delta \ \otimes \ O^i_w) \} \ . \tag{26}$$

Then, we update the opinion of each parameter θ denoted by O_θ, by taking cumulative fusion on its previous opinion and its change opinion denoted by $O_{\Delta\theta}$, as follows:

$$O_\theta = O_\theta \ \widehat{\oplus}\ O_{\Delta\theta} \ . \tag{27}$$

IV. Experimental Evaluation

As each data feature has its own distributional properties in the training dataset, we assess the significant impact of data feature distributional shape including intrinsic properties like feature skewness, on the model performance (see Appendix E for more details). We evaluate the proposed method by investigating the correlation of data opinion initialization as interpretable data opinions with the model performance. Since a DNN model with higher accuracy exhibits higher projected probability and lower uncertainty, we expect different input data opinion initialization impact the trustworthiness and total uncertainty of the DNN model. We assess our approach based on the presence of different statistical factors in initializing data opinions. We demonstrate that the belief, uncertainty, and trustworthiness, i.e., projected probability, will be improved in a DNN model based on its loss function optimization during training process.

Datasets. We have evaluated the proposed method on two real-world datasets: 1) SUSY dataset[2] with 5 million samples from particle physics community containing 18 features, two target labels, and no missing/misleading data values [11], 2) KDD Cup 99 dataset[3], with 4 million samples from computer network logs used for intrusion detection task containing 41 features (including categorical features), three selected target labels, and no missing/misleading data values [12].

Evaluation Setup. We implemented the proposed data opinion propagation method in Python[4], and the assessments have been performed on a machine with Intel(R) Core(TM) i7-10750H 2.60GHz (6 cores) processor and 16GB RAM allocated to our computation.

To run our experiments, we ensured the datasets were almost balanced, and the data were encoded (categorical features) and normalized. We ran the experiments over 15% of the entire data points for each dataset with 80% training and 20% testing sampling. We utilized a 4-layer neural network with 20-10 and 30-15 neurons in the hidden layers to train and test the model on Susy and KDDCUP99 datasets over 50 and 20 training epochs, respectively. In addition, we exploited the Cross-Entropy loss function and Adam optimizer with an initial learning rate of 0.001 to train our classifiers. The learning rates used in this paper are the initial ones for the adaptive optimization process using the Adam optimizer. Note that this approach is a topology agnostic method based on the experiments in [1]. Therefore, Although different DNN architectures may impact the total model accuracy and then model trustworthiness, this method is applicable by any arbitrary DNN architecture.

[2]https://archive.ics.uci.edu/ml/datasets/SUSY

[3]https://archive.ics.uci.edu/ml/datasets/KDD+Cup+1999+Data

[4]The source code is available at: https://github.com/[NAMEREMOVED]/sl-uncertainty

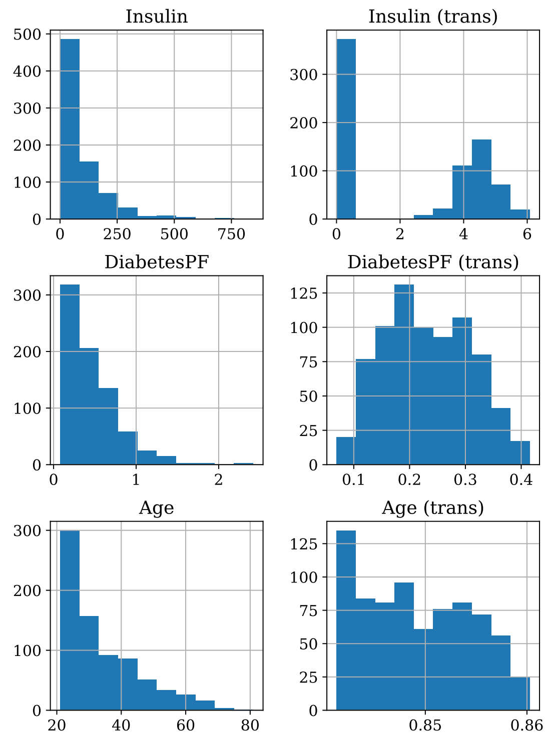

Fig. 1. The original and transformed distributions of the three most skewed features of the Pima Indians Diabetes Dataset

A. Results and Interpretations

A Preliminary Experiment. As the first part of the evaluations, a preliminary experiment is performed to demonstrate how the distributional properties in input data can impact the model performance. We used Pima Indians Diabetes Dataset[5] originally from the National Institute of Diabetes and Digestive and Kidney Diseases.

We apply a classification task using different traditional classifiers like K-nearest, Logistic Regression, Random Forest, AdaBoost, and a DNN classifier (*MLPClassifier* with three layers optimized by Adam and with an initial learning rate of 0.01) on both original distributions (e.g., skewed) and transformed distributions (e.g., no longer skewed). In Figure 1, the distribution of the three most skewed features of the dataset named *Insulin*, *DiabetesPedigreeFunction*, and *Age*, as well as the transformed features, are shown. Therefore, in Figure 2, we can see the accuracy of different classifiers including the DNN model over the original and transformed distributions of the most skewed features that hold the highest skewness and kurtosis in this dataset.

The results show that the transformed feature distribution mitigates the skewness and kurtosis as two distributional properties of training data. We can see the accuracy improvement of all classifiers in the presence of transformed features. Therefore, the skewed training data features that also have higher kurtosis, can negatively impact the DNN performance. This result also indicates the necessity of considering statistical

[5]https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database

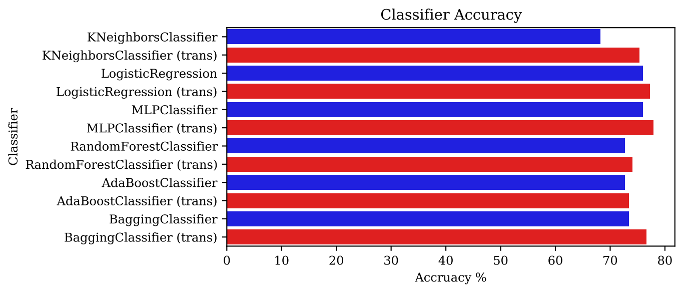

Fig. 2. The model accuracy of different classifiers based on the original (green bars) and transformed (red bars) distributions of the three most skewed features of the Pima Indians Diabetes Dataset

and distributional properties of input data in initializing data opinions before propagation through the DNN network.

Main Experiments. Since the datasets used to train DNN models can be considerably different in terms of their statistical properties, we used the equally-weighted average of two distinct groups of statistical factors for each data opinion determinant in each dataset. We run the experiment for two groups of statistical factors mentioned in Section III-A. We first run the experiment for each group of factors separately and then together. In Figures 3 and 4, there are two lines, blue and red, indicating the change in model opinion determinants as well as the model projected probability considering the first and second group of data opinion initialization factors, respectively. The green line also indicates the change in model opinion determinants and its projected probability considering both sets of factors simultaneously, by the equally-weighted average. During training epochs, in Figures 3 and 4, we can see the increasing change in model belief as well as the decreasing change in model uncertainty for both datasets considering different factors separately and together.Therefore, model belief and uncertainty are relatively improved during training epochs in a way that belief is increased while uncertainty is decreased. Finally, we can observe that during the training process, the model trustworthiness or projected probability is increased when there is no significant increasing change in disbelief.

As the results show, arbitrary selection of statistical factors to initialize the input data opinions may impact the total model trustworthiness. Therefore, this approach is flexible enough in selecting different statistical factors based on the application, data type, research domain, and the objective of the classification task. However, using the weighted average of the most correlated factors in initializing each determinant of interpretable data opinions will lead to a more realistic view of the model trustworthiness because of considering more aspects of data distribution in a batch of training data. Note that as mentioned before, we use equally-weighted average of distinct factors in initialization process. Therefore, based on the application and task domain, different weights can be assigned to different factors for averaging to tune the importance and priority of some specific statistical factors.

Furthermore, Figures 5 and 6 illustrate that the training processes over both SUSY and KDD Cup 99 datasets are performed effectively as decreasing loss and increasing accuracy during the training process shown in these figures is a sign of a successful learning process. A successful training process leads to an increase in model belief and projected probability as a measure of trustworthiness for both datasets while our model uncertainty is decreasing significantly. Thus, the projected probability is positively correlated with the model performance, and we can trust the DNN model much more than before after a successful training process. In comparison with the other trained models in the case of similar accuracy, we can decide to select the model with a higher projected probability at first, and in case of no significant discrepancy, the model with higher belief, lower disbelief, and lower uncertainty will be selected. In addition, we see insignificant change in the amount of disbelief (approximately less than 0.1) during training, which is convincing since our input data that fed to the DNN were of high quality without data type mismatch or any missing or misleading data records.

The low-quality input data may have missing and misleading data points or data type mismatch that give rise to input data with higher disbelief and lower trustworthiness. In Figures 7 and 8, we demonstrate the behavior of the proposed method in the presence of input data with lower quality. In this regard, we experiment the proposed method on both datasets with different ratios of missing/misleading data points. The results show that the lower quality input data with lower trustworthiness have higher model disbelief along with an increase in model disbelief during training rather than input data with higher quality. Therefore, we can observe that the more missing/misleading data ratios are in the datasets, the more model disbelief and the less model belief and trustworthiness (projected probability) we achieve during training epochs. We can also observe that low-quality data with lower trustworthiness may decrease the model belief and eventually, decrease the model trustworthiness during the training phase.

We report the average performance results as well as initial and ultimate model opinions during five distinct runs for each dataset in Table I. In this table, the model opinion determinants for training and testing phases over each dataset are updated and improved based on the loss function and accuracy achieved during a total number of training epochs. Note that the total belief and projected probability or trustworthiness of the model have been increased compared to the first epoch of training while the uncertainty has been relatively decreased. This result confirms the impact of successful training and loss optimization on the quantification and propagation of the total opinion's determinants.

In conclusion, our evaluation demonstrates that the opinion quantification process is adequately flexible when statistical properties of the input data are incorporated into the process. Note that the selection of these statistical factors can be different from one model to another based on the characteristics of the application domain. For example, in the medical domain, the reliability of input data could be a major factor in the opinion generation while in another domain, the skewness or

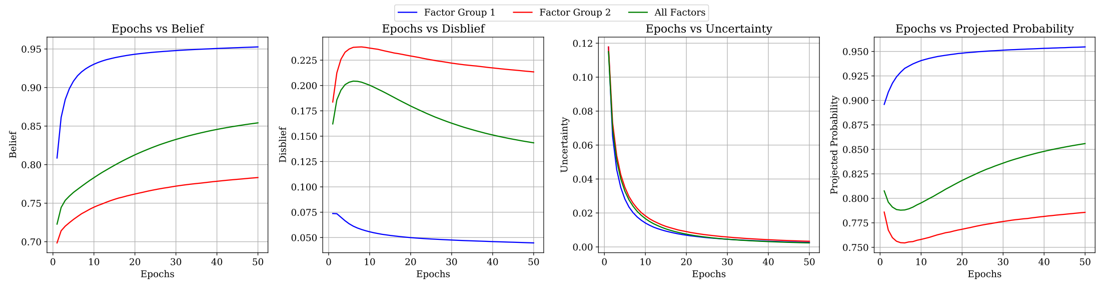

Fig. 3. The average improvement process of opinion determinants during 50 training epochs over 5 different runs based on different factors in numerical features for SUSY dataset

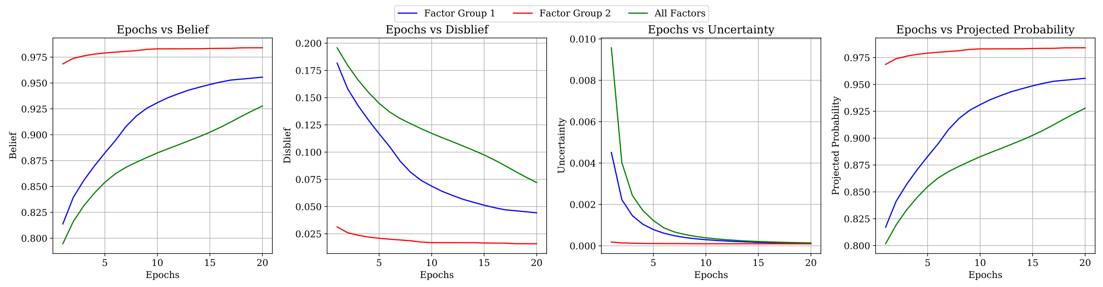

Fig. 4. The average improvement process of opinion determinants during 20 training epochs over 5 different runs based on different factors in numerical features for KDD CUP 99 dataset

TABLE I
THE AVERAGE PERFORMANCE AND OPINION RESULTS FOR TRAINING AND TESTING PHASES DURING 5 DIFFERENT RUNS OVER EACH DATASET

Dataset	**Performance**	**1st Training Epoch Opinion**				**Total Opinion**			
	Total Accuracy	Belief	Disbelief	Uncertainty	Proj. Prob.	Belief	Disbelief	Uncertainty	Proj. Prob.
SUSY: *Training*	72.863%	0.6420	0.0160	0.3420	0.8130	0.8470	0.0080	0.1440	0.9640
SUSY: *Testing*	69.785%	-	-	-	-	0.8495	0.0089	0.1417	0.9203
KDDCup99: *Training*	99.065%	0.9280	0.0050	0.0670	0.9610	0.9978	0.0002	0.0021	0.9981
KDDCup99: *Testing*	97.846%	-	-	-	-	0.9969	0.0002	0.0029	0.9970

kurtosis of data can have higher priority to be considered.

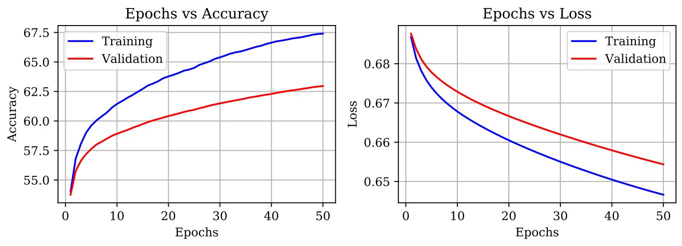

Fig. 5. The accuracy and loss values for training and validation phases during 50 training epochs over SUSY dataset

V. Related Work

There are several uncertainty quantification approaches in deep learning that intend to quantify the concept of trust or uncertainty in neural networks. Bayesian DNNs that learn weights' distribution, are the state-of-the-art techniques for predictive uncertainty estimation [13]–[17]. The Bayesian DNNs are computationally expensive compared to non-Bayesian (standard) DNNs. In addition, there is no method

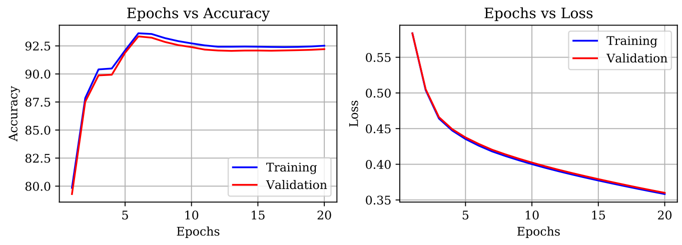

Fig. 6. The accuracy and loss values for training and validation phases during 20 training epochs over KDD CUP 99 dataset

proposed in these studies to measure the trustworthiness of parties which is vital in consensus tasks. In regular DNNs, there are some methods for quantifying trust in the learning models. For instance, in [18], the authors introduce a new trust quantification approach based on a descriptive matrix called Trust Matrix as a complement to their previous work [19] and [20]. However, in case of low confidence and incorrect prediction, a substantial trust value can be seen which is not convincing, and the remedy suggested for this issue

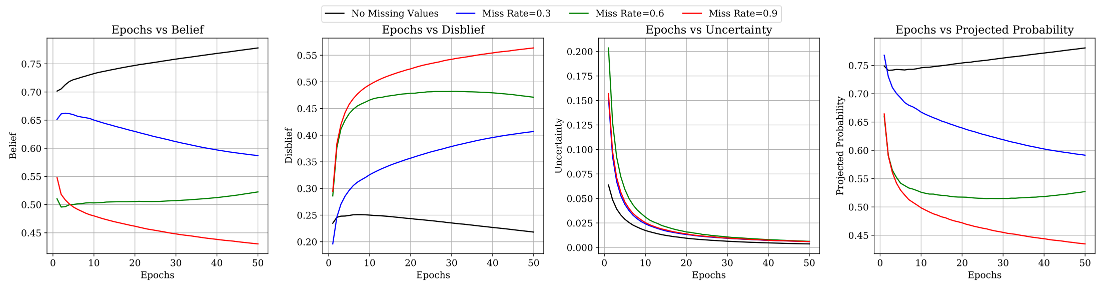

Fig. 7. The improvement process of opinion determinants during 50 training epochs over training data with different level of trustworthiness based on different miss rate for SUSY dataset

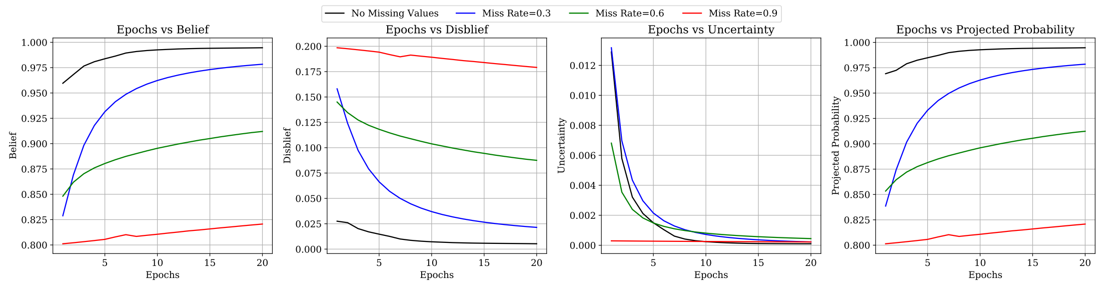

Fig. 8. The improvement process of opinion determinants during 20 training epochs over training data with different level of trustworthiness based on different miss rate for KDD CUP 99 dataset

can cause more cost and complexity for the whole system. Subjective logic is also another approach used by [21]–[23] to consider second-order uncertainty and trustworthiness in their learning systems. A closely related work is a proposal by Cheng et al. [1] that exploits Subjective Logic [2] to produce subjective opinions of a DNN to quantify trust in multi-layer neural networks. They introduce a quantification method named DeepTrust, as a framework based on Subjective Logic in which the model opinion and trustworthiness are quantified. Our proposal can be considered as an extension of the work of [1] where we studied the limitations of the current method (see Section **??**) and extended the method to incorporate interpretable input data opinions.

VI. Conclusion

In this paper, we investigated the limitations of the state-of-the-art opinion propagation method to quantify model trustworthiness. We proposed a new approach to initialize input data opinions by interpretable statistical metrics based on intrinsic properties of training data. To initialize the numerical, categorical, and target features, we selected distinct statistical properties originated in input data feature distributions. We also improved the opinion optimization process by considering uncertain evidence in forming the error opinion. Finally, we demonstrated the impact of input data distribution on the model performance and evaluated our proposed data opinion propagation method over two real-world datasets. Our experimental results confirmed that the model trustworthiness and performance are correlated such that the model belief and uncertainty are improved during the training process with respect to different data opinion initialization.

References

[1] M. Cheng, S. Nazarian, and P. Bogdan, “There is hope after all: Quantifying opinion and trustworthiness in neural networks,” *Frontiers in Artificial Intelligence*, vol. 3, p. 54, 2020.

[2] A. Jøsang, *Subjective logic*. Springer, 2016.

[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[4] J. Moon, J. Kim, Y. Shin, and S. Hwang, “Confidence-aware learning for deep neural networks,” in *international conference on machine learning*. PMLR, 2020, pp. 7034–7044.

[5] J. Van Amersfoort, L. Smith, Y. W. Teh, and Y. Gal, “Uncertainty estimation using a single deep deterministic neural network,” in *International Conference on Machine Learning*. PMLR, 2020, pp. 9690–9700.

[6] D. Sundgren and A. Karlsson, “Uncertainty levels of second-order probability,” *Polibits*, no. 48, pp. 5–11, 2013.

[7] R. W. Van Der Heijden, H. Kopp, and F. Kargl, “Multi-source fusion operations in subjective logic,” in *2018 21st International Conference on Information Fusion (FUSION)*. IEEE, 2018, pp. 1990–1997.

[8] L. J. Cronbach, “Coefficient alpha and the internal structure of tests,” *psychometrika*, vol. 16, no. 3, pp. 297–334, 1951.

[9] N. Tagasovska and D. Lopez-Paz, “Single-model uncertainties for deep learning,” *Advances in Neural Information Processing Systems*, vol. 32, pp. 6417–6428, 2019.

[10] C. E. Shannon, “A mathematical theory of communication,” *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[11] P. Baldi, P. Sadowski, and D. Whiteson, “Searching for exotic particles in high-energy physics with deep learning,” *Nature communications*, vol. 5, no. 1, pp. 1–9, 2014.

[12] S. D. Bay, D. Kibler, M. J. Pazzani, and P. Smyth, “The uci kdd archive of large data sets for data mining research and experimentation,” *ACM SIGKDD explorations newsletter*, vol. 2, no. 2, pp. 81–85, 2000.

[13] A. Kendall and Y. Gal, “What uncertainties do we need in bayesian deep learning for computer vision?” *arXiv preprint arXiv:1703.04977*, 2017.

[14] 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.

[15] B. Lakshminarayanan, A. Pritzel, and C. Blundell, “Simple and scalable predictive uncertainty estimation using deep ensembles,” *arXiv preprint arXiv:1612.01474*, 2016.

[16] W. J. Maddox, P. Izmailov, T. Garipov, D. P. Vetrov, and A. G. Wilson, “A simple baseline for bayesian uncertainty in deep learning,” *Advances*

in Neural Information Processing Systems, vol. 32, pp. 13 153–13 164, 2019.

[17] V.-A. Nguyen, P. Shi, J. Ramakrishnan, U. Weinsberg, H. C. Lin, S. Metz, N. Chandra, J. Jing, and D. Kalimeris, "Clara: Confidence of labels and raters," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 2542–2552.

[18] A. Hryniowski, X. Y. Wang, and A. Wong, "Where does trust break down? a quantitative trust analysis of deep neural networks via trust matrix and conditional trust densities," *arXiv preprint arXiv:2009.14701*, 2020.

[19] A. Wong, X. Y. Wang, and A. Hryniowski, "How much can we really trust you? towards simple, interpretable trust quantification metrics for deep neural networks," *arXiv preprint arXiv:2009.05835*, 2020.

[20] A. Wong, A. Hryniowski, and X. Y. Wang, "Insights into fairness through trust: Multi-scale trust quantification for financial deep learning," *arXiv preprint arXiv:2011.01961*, 2020.

[21] B. Škorić, S. J. de Hoogh, and N. Zannone, "Flow-based reputation with uncertainty: evidence-based subjective logic," *International Journal of Information Security*, vol. 15, no. 4, pp. 381–402, 2016.

[22] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential deep learning to quantify classification uncertainty," *arXiv preprint arXiv:1806.01768*, 2018.

[23] T. Tsiligkaridis, "Information aware max-norm dirichlet networks for predictive uncertainty estimation," *Neural Networks*, vol. 135, pp. 105–114, 2021.

Appendix A
Single Deep Neural Network

A Deep Neural Network (DNN) is a computational network that can be considered as a function $\mathcal{M}_\Theta : \mathcal{D} \rightarrow \hat{Y}$ in which $\mathcal{D}$ is a set of input data fed into the network, Θ is a set of network parameters (weights and biases), and $\hat{Y}$ is the output set of the approximated values $\hat{y} \in \hat{Y}$. Moreover, the DNN model $\mathcal{M}_\Theta$ aims to be trained based on its input data (a finite set of training data) denoted by $\mathcal{D}$, according to the function $\mathcal{Q} : \mathcal{D} \rightarrow Y$ such that,

$$\mathcal{Q} = \{ \ (d_i, y_i) \mid i \in \mathbb{N}^{[1,m]} \ \} \ ,$$

where i is a data point with d_i and y_i representing features and target labels, respectively and m is the number of participating data points (data samples) in the training dataset.

In the DNN model $\mathcal{M}_\Theta$, we intend to optimize a loss function in the form of $\mathcal{L}(\mathcal{D}, \Theta; Y)$. This optimization is performed by minimizing the average of each data sample i's loss value with respect to the parameter set of Θ defined as,

$$\min_{\Theta} \ \mathcal{L}(\mathcal{D}, \Theta; Y) = \frac{1}{m} \sum_{i=1}^{m} loss(d_i, \Theta; y_i) \ , \tag{28}$$

where $loss(d_i, \Theta; y_i)$ is the loss function of the single data sample i in the training dataset.

Appendix B
Opinion Determinants in Subjective Logic

Definition 1 (Belief and Disbelief Masses). The belief b and disbelief d are the amount of being in support of and against the truth of a particular subjective variable in a DNN provided by a DNN component, respectively [2].

Definition 2 (Uncertainty Mass). The uncertainty u is the amount of not being confident (certain) or not having sharp idea regarding the truth of a subjective variable provided by a DNN component. The uncertainty is caused by vacuity of evidence in a DNN model. Thus, the fewer observations (data samples) may give rise to more uncertainty [2].

Definition 3 (Base Rate Mass). The base rate a is the prior probability for a subjective variable that can be interpreted as a kind of bias term which provided by a DNN component. The base rate can help us take the bias measure into account [2].

Appendix C
Parameter Opinion Initialization

We initialize the model parameters opinions according to the lack (vacuity) of observations by Uncertainty Maximization [2] on randomly generated opinions from uniform distribution $\mathcal{U}(0,1)$ with respect to the opinion additivity requirement 3. In Uncertainty Maximization, an opinion for the parameter θ should have at least one belief mass of zero, i.e. $b_\theta = 0$ and the uncertainty is calculated as,

$$u_\theta = \min(1, \frac{PP_\theta}{a_\theta}) \ , \tag{29}$$

where PP_θ and a_θ is the projected probability and the base rate of the parameter θ's opinion, respectively.

Appendix D
Parameters Opinion Update

To optimize the total opinion of our model (minimizing the uncertainty while maximizing the belief), we need to update the parameters' opinions based on the error value $|y - \hat{y}|$ that the model is achieving during training. Thus, we exploit back propagation approach for the parameters' opinion updates while we are updating the actual parameters in mini-batch gradient descent to minimize the loss function. Based on what we perform in back propagation process to update the parameters in each layer, we initialize the opinions of parameters' change, $O_{\Delta w}$ and $O_{\Delta b}$, backward from the last layer towards the first layer using error opinion set O_δ and its pairwise multiplication with predicted opinion set $O_{\hat{y}}$ in the previous layer. For instance, in layer i,

$$O^i_{\Delta b} = O_\delta \qquad \text{and} \qquad O^i_{\Delta w} = O_\delta \otimes O^{i-1}_{\hat{y}} \ . \tag{30}$$

Finally, in each layer i with the neuron set $layer_i$, we update the error opinion set O_δ by assigning a set consists of multi-source average fusion on the pairwise multiplication of the previous O_δ and the weight opinion set O^i_w as follows:

$$O_\delta = \{ \bigoplus_{layer_i} (O_\delta \ \otimes \ O^i_w) \ \} \ . \tag{31}$$

Then, we update the opinion of each parameter θ denoted by O_θ, by taking cumulative fusion on its previous opinion and its change opinion denoted by $O_{\Delta\theta}$, as follows:

$$O_\theta = O_\theta \ \widehat{\oplus} \ O_{\Delta\theta} \ . \tag{32}$$

Appendix E
Impact of Data Distribution on Model Performance

The objective of this preliminary experiment is to demonstrate how the distributional properties in input data can impact the model performance. We used Pima Indians Diabetes Dataset[6] originally from the National Institute of Diabetes and Digestive and Kidney Diseases.

We apply classification task using different traditional classifiers like K-nearest, Logistic Regression, Random Forest, AdaBoost, and a DNN classifier (*MLPClassifier* with three layers optimized by Adam and with initial learning rate of 0.01) on both original distributions (e.g. skewed) and transformed distributions (e.g., no longer skewed). As shown in Figure 1, the distribution of the three most skewed features of the dataset named *Insulin*, *DiabetesPedigreeFunction*, and *Age* as well as the transformed features are shown. Therefore, in Figure 2, we can see the accuracy of different classifiers including the DNN model over the original and transformed distributions of the most skewed features that hold the highest skewness and kurtosis in this dataset. The results show that

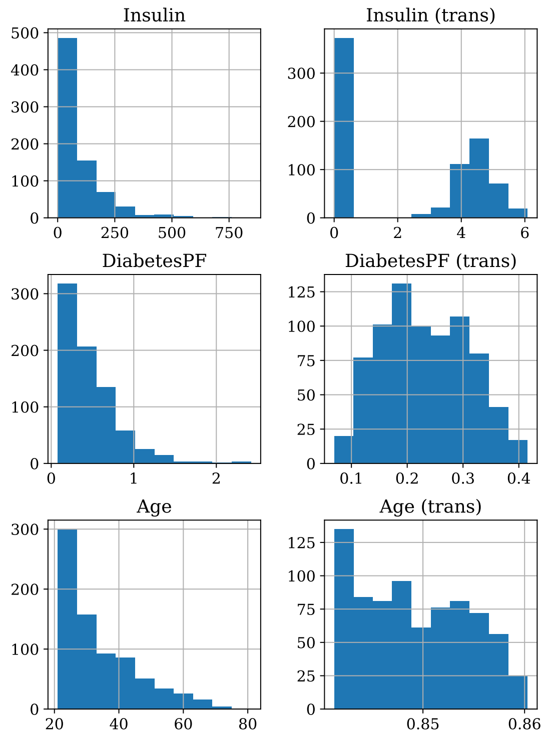

Fig. 9. The original and transformed distributions of the three most skewed features of the Pima Indians Diabetes Dataset

the transformed feature distribution mitigates the skewness and kurtosis as two distributional properties of training data. We can see the accuracy improvement of all classifiers in the presence of transformed features. Therefore, the skewed training data features, that have higher kurtosis as well, can negatively impact the DNN performance. This result also indicates the

[6]https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database

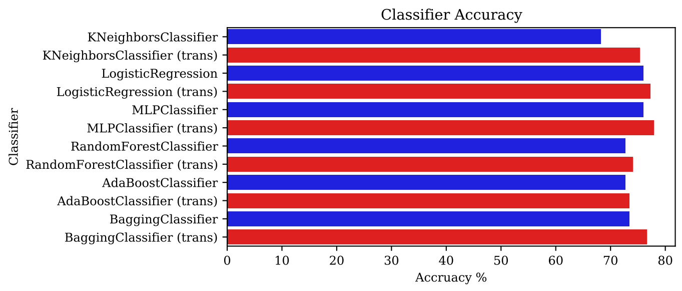

Fig. 10. The model accuracy of different classifiers based on the original (green bars) and transformed (red bars) distributions of the three most skewed features of the Pima Indians Diabetes Dataset

necessity of considering statistical and distributional properties of input data in initializing data opinions before propagation through the DNN network.

Appendix F
Model Training

Figures 5 and 6 illustrate that the training process over both SUSY and KDD Cup 99 datasets has been performed effectively for both datasets as decreasing loss and increasing accuracy during the training process shown in these figures is a sign of a successful learning process.

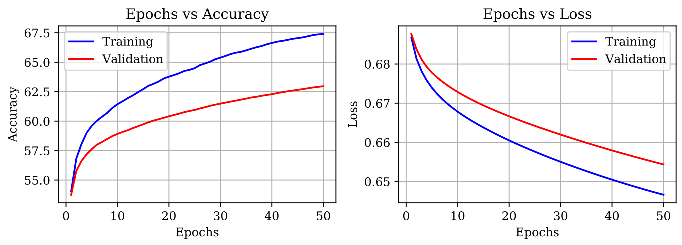

Fig. 11. The accuracy and loss values during 45 training epochs over SUSY dataset

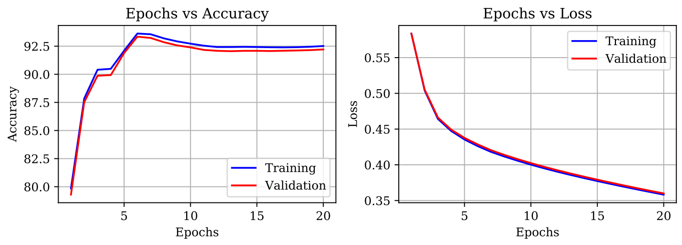

Fig. 12. The accuracy and loss values during 20 training epochs over KDD Cup 99 dataset

Appendix G
Investigation on Optimal Thresholds in Opinion Optimization

According to Equation 23, in the opinion optimization process, we use two different thresholds ϕ_1 and ϕ_2 to determine the error opinion O^l_δ for each target label l. Thus, we need to find the optimal values for selecting the thresholds. In this regard, we applied the proposed method using Susy dataset on different selections of the thresholds with respect to their requirements mentioned in Section III-C. Figure 13 shows the model opinion results during training using different selections of the two thresholds. We can observe that when $\phi_1 = 0.4$ and $\phi_2 = 0.7$ (the purple line), we see the higher amount of model belief and trustworthiness along with lower amount of disbelief rather than other choices of thresholds.

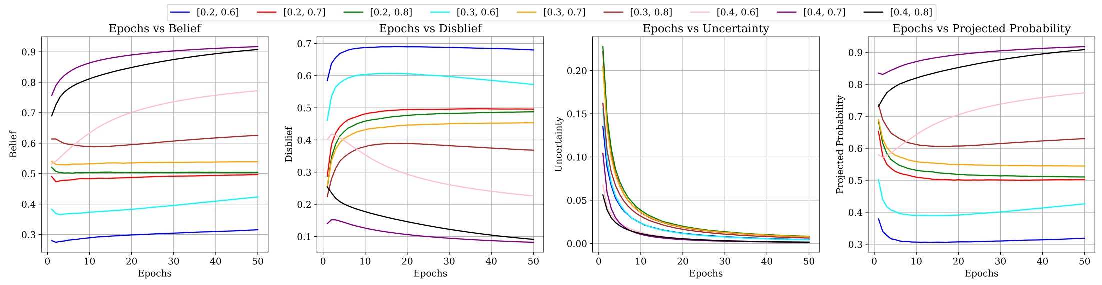

Fig. 13. The behavior of each determinant of total model opinion with respect to different selections of opinion optimization thresholds ϕ_1 and ϕ_2