Establishing Reliability in Crowdsourced Voice Feedback (CVF) and Evaluating the Reliability in the Absence of Ground-Truth

Aashish Jain*
aajain@amazon.com
Amazon Alexa
Boston, MA, USA

Sudeeksha Murari†
sud@amazon.com
Amazon Alexa
Boston, MA, USA

ABSTRACT

Intelligent Voice Assistant (IVA) systems, such as Alexa, Google Assistant and Siri, allow us to interact with them using just the voice commands. IVAs can elicit voice feedback directly from the users and use their responses to improve the various components of IVAs. One concern with using such crowdsourced voice feedback (CVF) data is the reliability of feedback itself such as background noise or disingenuous feedback. In this paper, we propose ways to establish confidence scores to indicate the reliability of the CVF data. We build a probabilistic Bayesian Belief Network (BBN) model, which uses the CVF data as training dataset. Since human annotation of the CVF data can be expensive, we explore ways to evaluate such a model without human labeled data. We propose several metrics that (i) do not need any ground-truth, (ii) can be simply computed using the CVF data, and (iii) can reliably measure the model performance to output confidence scores indicating reliability.

CCS CONCEPTS

• **Mathematics of computing** → **Bayesian networks**; • **Computing methodologies** → **Model verification and validation**; **Uncertainty quantification**; *Bayesian network models.*

KEYWORDS

intelligent voice assistant, crowdsourced voice feedback, Bayesian belief network, ground-truth data, feedback reliability

ACM Reference Format:
Aashish Jain and Sudeeksha Murari. 2021. Establishing Reliability in Crowdsourced Voice Feedback (CVF) and Evaluating the Reliability in the Absence of Ground-Truth. In *Proceedings of Data-Efficient Machine Learning Workshop at KDD (DeMaL'21).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In order to collect feedback on specific aspects of an intelligent Voice Assistant (IVA), a system like Crowdsourced Voice Feedback (CVF) can be valuable. CVF is a system that can target specific interactions based on domain/intent, device type or other aspects of a request made to an IVA. Typically, when a user makes a request to an IVA, the IVA responds with an action or with a response. The user is then asked a question by CVF about the interaction to gauge if the IVA rendered a satisfactory response/action. These CVF questions can be generic or very specific to the interaction, and users' responses are recorded and used to improve the IVA. An example of such an interaction, along with the CVF question can be seen in Fig. 1. Other than binary questions, there are numeric

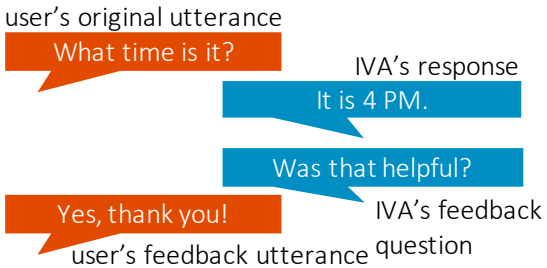

Figure 1: An example of a user-IVA interaction

feedback questions as well, such as "how did I do on a scale of 1-5?". The users' typical responses to CVF can be broadly categorized as *yes, no, maybe, silence* or some numeric response. Annoyed users also say things like "shut up" or "go away" or completely ignore the CVF question and ask for something else. In this work, we only focus on the binary questions and responses are assigned a feedback type of *yes*, *no* and *other*. In the example shown in Fig. 1, the feedback type of the response is *yes*. The *other* feedback type contains several scenarios such as when a user (i) provides uncertain feedback ("maybe" or "I don't know") (ii) provides the feedback in a noisy background such that the feedback cannot be understood, and (iii) ignores the feedback question. In the Fig. 1, we also illustrate what we refer to as original and feedback utterances, as shown on the first and fourth lines, respectively.

In this paper, we propose to (i) use Bayesian Belief Network (BBN) [20, 26] to establish the reliability of crowdsourced voice feedback (CVF) data in a frugal manner without human annotated training data, and (ii) evaluate CVF confidence detection model in the absence of ground-truth data. The BBN model can also be used to add interpretability to the model predictions by providing estimates of marginal contributions of Automated Speech Recognition (ASR), Natural Language Understanding (NLU) etc., towards the

*Corresponding author
†Work done while at Amazon Alexa.

DeMaL'21, August 2021, Virtual

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

three feedback types for a given interaction. However, interpretation of the predictions is not the focus of this paper. This paper focuses on the former, i.e. establishing reliability in CVF data and evaluating model in the absence of the ground-truth data. In an IVA system, millions of CVF data-points are collected each month. This data is then used as training/testing data in models or to generate health metrics around IVA performance. Often, there are concerns on reliability of CVF data due to sarcastic responses or heavy background noise etc. To address this, a BBN model is used to predict feedback type and confidence scores on user's feedback. With a continuous confidence score for each feedback, noisy data can be filtered-out or reliable data can be filtered-in.

2 RELATED WORK AND OUR CONTRIBUTION

Any type of crowdsourced data comes with some level of noise and there have been efforts in the past to clean the labels in crowdsourced data [6, 10], with the purpose to use them for training supervised models. Another research area in dealing with noisy labelled data, not just limited to crowdsourced data, is to estimate the uncertainty in the labels so that more uncertain data points can be kept separate from more confident data points, and there is a dedicated framework for that as well [24]. There is research on learning with noisy data itself, for example using deep neural networks [7]. In all these areas, the focus has been mainly on training the model, i.e. getting reliable training data with minimized noise and building models that have minimal influence due to noise in the training data. However, it is difficult to curate a reliable ground-truth data set for model evaluations using these approaches. In the absence of ground-truth data, it becomes challenging to compare models and select the best one for deployment. There is an idea called reverse testing [5], which does not rely on the ground-truth data. Another idea is in the context of interpretable machine learning, where the focus is on explanation, rather than prediction, and novel evaluation methods were proposed in the absence of the ground-truth data [36]. A more sustainable solution to evaluate models in the absence of ground-truth data is to have a set of reliable success metrics, and there exists work in which some metrics were proposed that can be used in the absence of the ground-truth data, though only for binary classification problems [11]. In these research attempts, reliable labelled data sets were already present to train the model in the first place. We do not find any research on dealing with both issues (i.e. noisy crowdsourced data for training models and absence of ground-truth for model evaluation) for crowdsourced voice feedback domain.

We have the challenges of (i) not having reliable labelled data to train the model to make predictions on both feedback type and confidence scores on them, and (ii) not having the ground-truth data to evaluate that model. Having dependence on the ground-truth data is not desirable due to the cost of human annotations and risk to user privacy. In our work, we introduce new metrics and also use existing metrics, that can be computed using only the CVF data, and do not require any ground-truth data. We propose to use these CVF data based metrics to replace ground-truth data based metrics, used in traditional supervised learning, like $F1$ or ROC AUC. In order to show that these CVF data based metrics are reliable and can indeed be good proxies for ground-truth based metrics (as if we had a ground-truth data set available), we curate a small human annotated ground-truth data set to compare these metrics with ground-truth data based metrics. Moreover, we also define some metrics, that use both CVF data and ground-truth data, that can be used to evaluate the predicted confidence scores in the voice feedback, as opposed to only the feedback type prediction. This allows us to establish confidence on model predicted confidence scores. To the best of our knowledge, this is the first work to (i) build a BBN based model using CVF data to predict feedback type and associated confidence scores and (ii) present reliable metrics to measure the performance of crowdsourced voice feedback confidence detection model in the absence of ground-truth data. Last, we also present hyper-parameters, used in building the BBN model, that can be tuned to achieve optimal model performance for applications such as this.

3 DATA

Usually, the IVAs use spoken language understanding (SLU) based models [4, 18, 29] to detect domain, intent and slots, which are in turn used to automatically tag user's feedback types as *yes*, *no* or *other*. Throughout this paper, we will refer to this as *user feedback*. The confidence on the user feedback may depend on many factors such as (i) feedback type determination by the SLU models (ii) users' sentiment scores [19], (iii) SLU model hypothesis confidence scores (iv) ASR model hypothesis confidence scores [16, 17, 22] and many other run-time meta signals that are collected during the user-IVA interaction. We use these signals to build a BBN based model in this work. Note that we do not use any lexical based signals in this work. For most of the signals, we use them corresponding to both the original and the feedback utterance as illustrated in the example shown in Fig. 1. All these signals in a single record, consisting of both the original and the feedback utterances, forms one sample of the CVF dataset.

We collected one month of CVF data from IVA users, where users are not identifiable. From that data, we subsampled 8.5×10^6 samples (Dataset A). Each data sample consists of two utterances, including the original and feedback utterance, as we illustrated in Fig. 1 above. We split the CVF dataset into training and test sets, leaving 7.5×10^6 samples as the CVF training set and 1×10^6 samples as the CVF test set.

To demonstrate that the newly designed metrics computed on the CVF test set represent the true model performance, we study correlations between CVF test set metrics and ground-truth data based metrics. For this aspect of validating the metrics we propose, we randomly select 3.1×10^3 CVF data samples from a different time window than Dataset A. We annotate this smaller dataset to generate the feedback type labels on each CVF data sample, and refer to this as ground-truth test set.

4 METHODOLOGY

A CVF is useful in understanding whether a rendered experience through an IVA was satisfactory or not. For this purpose, the feedback type represents the success or failure of an experience with the IVA. We believe that a Bayesian Belief Network (BBN) is well suited for not only producing confidence scores indicating the reliability of user feedback, but it also helps understand the factors

that contribute to a user responding yes or *no* to a CVF question like "did I play the correct song". This paper focuses on building a BBN based model to establish reliability in the user feedback, and more importantly on introducing metrics to measure the model performance in the absence of ground-truth data. The BBN model predicts the potential user response by generating outputs of probabilities for each feedback type (*yes, no, other*). We then compare the outputs with the actual responses provided by users to assess confidence scores. For instance, if the model predicts *yes* with a probability of 0.8 and the user responds "yes", our confidence in the feedback response is 0.8. If our model predicts *yes* with a probability of 0.7 and *no* with a probability of 0.3, and the user response is "no", the confidence score is 0.3.

4.1 Bayesian Belief Network Model

A Bayesian belief network (BBN) is a directed acyclic graph (DAG) based framework that is used to model the joint probability distribution $\mathbf{P}$ of a set of variables in a data set [20, 26]. The DAG is defined as $G = (V, E)$, with V representing a set of nodes or variables and $E \subseteq V \times V$ representing edges in the network. The joint probability distribution of all variables $|V|$ is the set of parameters Θ in the BBN, and so the BBN is represented as the pair (G, Θ). This network effectively captures the independence relationships between the variables $\mathbf{X} = \{X_1 ... X_N\}$, with graphical separation in G, which also corresponds to the conditional independency in probability, leading to the following factorization [20, 21]

$$\mathbf{P}(\mathbf{X}|G, \Theta) = \prod_{v \in V} P(X_v | \Pi_{X_v}, \Theta_{X_v}), \tag{1}$$

where, Π_{X_v} is the parent node of the variable X_v. Here, the joint probability distribution of $\mathbf{X}$ (with parameters Θ) can be thought of as a multiplication of the local distribution of each variable X_v (with parameters Θ_{X_v}) conditional on each variable's parents Π_{X_v}. Note that for a node, there can be a single or multiple or even no parents. The factorization (Eq. 1) is valid only when we do not have any missing data.

Depending on whether the structure of the BBN is known or unknown, and if the data is full or partial (i.e. with missing values), there are four possible learning approaches [3]. In this work, we learn the structure from the data, hence we use the unknown structure scenario. We also treat our data prior to training (see Sec. 4.2) so that there is no missing values, and so our data is full. The BBN model is learned using the data in two broad steps. In the first step, the DAG structure G of the BBN is learned using the data, which captures the dependence structure in the data.

Once, the structure learning is finished and we find G, the second step is to learn the parameters Θ, to finish learning the pair (G, Θ), i.e. the BBN . Usually, the parameters are learned either using maximum likelihood or Bayes method [31].

4.2 Training

We first pre-process the CVF training data for learning the structure and the parameters of the BBN. The BBN can be learned using data that contains only discrete variables or only continuous variables or mix of both. In this work, we use `bnlearn` [31, 32], a popular `R` package for learning BBNs. This package supports continuous and discrete BBNs but not the hybrid. Since feedback type is discrete, we transform all continuous variables into discrete variables, to train a discrete BBN. The number of levels, n_l, that each continuous variable is discretized into discrete variable, is a hyper-parameter. There are some missing values in some of the variables, and we replace them with a unique level (e.g. "missing"), to make the data full. We believe that a missing field can affect our confidence in the user feedback and therefore treat missing values as a discrete category in our training data. From a reliability perspective, missing data are useful information in establishing confidence.

The pre-processed data is used to first learn the structure of the BBN. We use various constraint-based [2, 9, 14, 23, 27, 33, 34, 37], score-based [28] (using Bayesian Information Criterion (BIC) [30] as scoring metric), hybrid [12, 14, 35] and local discovery [8] learning algorithms to learn the structure. For each model, we generate five network structures by applying nonparametric bootstrap to the training data, and then compute the joint strength of all possible edges. This bootstrapping procedure allows us to generate more generic network structure. Finally, for the parameter learning, we use the Bayes method.

4.3 Inferencing

We make all the probabilistic inferences using `gRain` package [15], which allows to compute conditional probability of any variable conditioned on any other set of variables, for a given sample. Unlike `bnlearn`, which supports approximate inferencing, `gRain` package supports exact inferencing. In our use-case, we want to compute the probability of the feedback type variable X_F, for all three feedback types X_{F^t} with $t = \{yes, no, other\}$, conditioned on the values of all other variables. We define that conditional probability as $P(X_{F^t}|\mathbf{X}')$, where, $\mathbf{X}'$ is $X_1, ...X_j, ...X_N$ with $j \neq F$. The conditional probability is used as confidence score C^t on feedback type t, in other words,

$$C^t = P(X_{F^t}|\mathbf{X}'); \;\; t \in \{yes, no, other\}, \tag{2}$$

with $\sum_t C^t = 1$. We can also use $\underset{t}{\text{argmax}}\ C^t$ to predict the feedback type t', like a classification problem. Note that, to distinguish the user feedback type from model predicted feedback type, we denote them by t and t', respectively. When the predicted feedback type t' (which is the one with the highest C^t) is the same as the user feedback response t, C^t is retained as our confidence score on the feedback received. If C^t is a low number, it does not mean that the user was incorrect, it means that based on the signals we use, it is difficult to explain the feedback received, and therefore we put lower confidence in that data point. These data points are not readily usable and need additional inspection to understand the feedback provided. Having model's prediction of the feedback type, and the confidence score on the user's feedback type defined, we next discuss the model evaluation methods in the next section.

5 MODEL EVALUATION METHODS

For model evaluation, we propose a combination of new and existing metrics that do not need ground-truth data, but only the CVF data, and we call these CVF based metrics ($\mathbf{M^C}$). For validation of the $\mathbf{M^C}$ metrics, we also compute traditional evaluation metrics ($\mathbf{M^G}$) on a small human labeled ground-truth data, which was curated only to show correlations between $\mathbf{M^C}$ and $\mathbf{M^G}$ metrics.

5.1 CVF Based Metrics ($\mathbf{M^C}$)

We define 11 CVF based metrics and denote them using M_i^C, where $i = 1...11$. These metrics are based on (i) BBN structure, (ii) agreement between user's feedback type t and model predicted feedback type t', and (iii) user's sentiment scores. We summarize all the $\mathbf{M^C}$ metrics in Table 1, and explain them in detail below.

5.1.1 BBN Accuracy. There are information theory based metrics such as Akaike Information Criterion (AIC) [1] and likelihood based metrics such as BIC [30], that can be used to score how well the BBN fits the data. These metrics are based on log likelihood of a BBN given a dataset, and so the accuracy of the BBN depends on the size of the data itself [13]. As a result, these metrics do not provide the "absolute" accuracy but rather the "relative" accuracy of a BBN for a given data set. An alternative approach to measure the accuracy of a BBN was proposed [25], where the accuracy was approximated as conditional independencies present in the structure of the BBN. This alternate accuracy is called the Network Conditional Independencies Mutual Information (NCIMI). The idea behind this metric is to look at all relationships in the network that are conditionally independent, and then compute the mutual information (MI) dependency measure between the variables using the data. The non-zero values of the MI tell us about the inaccuracies in the model. The NCIMI is simply the summation of all possible MIs in the network (computed only on conditionally independent variable pairs). To perform model comparisons, we compute mean statistics of NCIMI. Based on this NCIMI mean statistics, we define two metrics: (i) M_1^C = mean of NCIMI on the CVF test set, (ii) M_2^C = median of NCIMI on the CVF test set. The smaller the values of these metrics, the more accurate the BBN is.

5.1.2 Classification Accuracy Assuming User Feedback As Ground-Truth. If we assume that the user's feedback is totally reliable and there is no noise (which is not the case), we can compute the agreement between the user feedback type t and the model predicted feedback type t'. We compute accuracy to represent this agreement and denote this metric as M_3^C. This metric represents the hypothesis that the users that provide feedback are correct most of the time, which was validated using a small sample annotation. This is not used as a standalone metric to evaluate the model.

5.1.3 Extreme Sentiment Based Metrics. The speech sentiment scores are computed on both the original and feedback utterance using the sentiment score model [19], which generates three types of sentiment scores: (i) activation, which captures if the user is excited or calm, (ii) valence, which represents positive or negative sentiment and (iii) satisfaction, which also measures the positive or negative experience. We find that activation is quite uniformly distributed across the feedback classes, however, satisfaction and valence distributions are sensitive to feedback type. It was also found in another study that the activation scores are not correlated with customer satisfaction [19]. The idea behind extreme sentiment based metrics is to assess what the model predicts when the sentiment scores are extreme. We posit that for extreme positive and negative sentiment scores, user's feedback type t mostly should be *yes* and *no*, respectively. Before defining extreme sentiment based metrics, we first discuss how we subset the data to get the extreme sentiment feedback test sets, both extreme negative sentiment (ENS) test set and extreme positive sentiment (EPS) test set. We consider EPS test set feedback as those ones for which both the satisfaction and the valence scores are greater than $\mu_s + k\sigma_s$ and $\mu_v + k\sigma_v$, respectively. Here, μ_s and μ_v are the means and σ_s and σ_v are the standard deviations of satisfaction and valence, respectively. Similarly, for ENS test set, we consider satisfaction and valence scores to be smaller than $\mu_s - k\sigma_s$ and $\mu_v - k\sigma_v$, respectively. We use $k = 3$ in this work as that provides sufficiently large ENS and EPS test set, while maintaining the sentiment extremeness, as found in an internal study. Having setup the two test sets for EPS and ENS feedback, we next define extreme sentiment based metrics.

Table 1: Summary of $\mathbf{M^C}$ metrics. The hypothesis that a metric should be higher or lower for a better model performance is indicated by ↑ and ↓, respectively, in the description.

$\mathbf{M^C}$	Description
M_1^C	NCIMI mean on CVF test set ↓
M_2^C	NCIMI median on CVF test set ↓
M_3^C	Accuracy between t and t' ↑
M_4^C	Mean of the confidence scores on *yes* class in ENS test set ↓
M_5^C	Recall of *no* class in ENS test set ↑
M_6^C	Rate of false negatives due to *yes* class in ENS test set ↓
M_7^C	Relative reduction in *yes* samples in ENS test set ↑
M_8^C	Mean of the confidence scores on *no* class in EPS test set ↓
M_9^C	Recall of *yes* class in EPS test set ↑
M_{10}^C	Rate of false negatives due to *no* class in EPS test set ↓
M_{11}^C	Relative reduction in *no* samples in EPS test set ↑

Extreme Negative Sentiment Based Metrics: In the extreme negative sentiment (ENS) test set, we posit that the model would generate lower confidence scores on the *yes* class and hence would have lower C^t for *yes*. To capture this, we define $M_4^C = \bar{C}^{yes}(\text{ENS})$ as a metric to represent the mean of the confidence scores on the *yes* class in the ENS test set, and we posit that it's value should be smaller for a better model. If we focus on the *no* class in the ENS test set, then we can make some arguments that if the user feedback type t is *no*, then the model should also predict *no*. In other words, if we consider t as reference and t' as model hypothesis, the false negatives should be much smaller for *no*. It is possible that some of the *no* from users would be predicted as *other* feedback types by the model but it is important that the model predicts mainly *no* and *other* feedback types but not *yes*. It is fine if there are some false positives for *no*, as model would be stealing from *other* feedback types to *no*. Based on these arguments, the recall of *no* class could be an important metric, and we denote that as $M_5^C = \text{recall}^{no}(\text{ENS})$ which should become higher as the model gets better. Along with M_5^C, we can also keep track of how many times the model predicted *yes* when the user feedback was *no*. We capture this information using $M_6^C = \frac{\text{FN}^{yes}}{n_s^{n,no}}\Big(\text{ENS}\Big)$, which is the ratio of the number of false negatives due to *yes*, FN^{yes}, to the total number of *no* responded by users in ENS test set $n_s^{n,no}$, and we hypothesize that this metric's value should be lower. The ENS test set should have as small number of *yes* feedback as possible, so even if the user's feedback is *yes*, the model should predict *no*. We use $M_7^C = 100 \times (n_s^{n,yes} - n_s^{n,yes'})/n_s^{n,yes}(\text{ENS})$ to denote the relative

reduction in the number of *yes* feedback, when going from t to t', where $n_s^{n,yes}$ and $n_s^{n,yes'}$ are the number of *yes* feedback in ENS test set as responded by users and predicted by model, respectively. The value of this metric should be higher for better models.

Extreme Positive Sentiment Based Metrics: Similar to ENS test set based four metrics, we define metrics $M_8^C - M_{11}^C$ for the EPS test set, by simply switching the feedback polarity from *yes* to *no* and vice versa. As a result, in the EPS test set, we have (i) $M_8^C = \bar{C}^{no}$(EPS) as the mean of the confidence scores on the *no* class, (ii) $M_9^C = \text{recall}^{yes}$(EPS) as the recall of *yes* class, (iii) $M_{10}^C = \frac{\text{FN}^{no}}{n_s^{p,yes}}\Big(\text{EPS}\Big)$ as the ratio of the number of false negatives due to *no*, FN^{no}, to the total number of *yes* responded by users $n_s^{p,yes}$ in EPS test set, and (iv) $M_{11}^C = 100 \times (n_s^{p,no} - n_s^{p,no'})/n_s^{p,no}$(EPS) as the relative reduction in the number of *no* feedback, when going from t to t', where $n_s^{p,no}$ and $n_s^{p,no'}$ are the number of *no* feedback in the EPS test set as responded by users and predicted by model, respectively. The values of metrics M_8^C and M_{10}^C should be lower, and values of metrics M_9^C and M_{11}^C should be higher, for better models.

5.2 Ground-Truth Based Metrics ($\mathbf{M^G}$)

Having human annotated ground-truth data with *yes, no, other* feedback type labels and the model predicted feedback type with confidence scores (Eq. 2), we can compute several metrics to measure the model performance. However, we emphasize that $\mathbf{M^G}$ metrics are generated *only* for establishing the efficacy of $\mathbf{M^C}$ metrics, as stated in Sec. 5. The $\mathbf{M^G}$ metrics are described in the Table 2. The

Table 2: List of ground-truth based metrics . The values of all metrics should be higher for better model performance, as indicated by ↑ in the description.

$\mathbf{M^G}$	Description
M_1^G	Accuracy ↑
M_2^G	Mean of precision across three feedback types ↑
M_3^G	Mean of recall across three feedback types ↑
M_4^G	Mean of F1 scores across three feedback types ↑
M_5^G	Mean of ROC AUC across three feedback types ↑
M_6^G	Mean of differences in confidence scores between ground-truth label and t ↑
M_7^G	Mean of confidence scores on ground-truth label ↑
$M_i^{G^u}$	For i = 1..4, computed between the ground-truth label and user feedback type t (rather than model predicted feedback type t'), and hence is a constant number for each i

metrics M_1^G - M_5^G are well established, so we only elaborate M_5^G - M_7^G in more detail. We use the model to get confidence scores on both user's feedback type t and ground-truth feedback type using Eq. 2. For instance, if t is *yes*, we use the C^t of *yes* as it's confidence score; if for the same sample, the ground-truth feedback type label is *no*, the C^t of *no* produced by the model is it's confidence score. We expect that the confidence score generated by the model for the ground-truth feedback type label, in general, should be greater than that generated by the same model for the user provided feedback type t, as that way the model is closer to ground-truth than the user feedback. Therefore, we can compute the difference between the confidence scores on ground-truth label and t for all samples in the ground-truth test set, and the mean of those differences is what we refer to as M_6^G. Higher value of this metric should correspond to better model performance. We also assume that the mean of the confidence scores on the ground-truth feedback type label, across all samples, should be higher for better models, and this value is represented by M_7^G. It is also important to mention that the metrics M_1^G - M_4^G rely only on the feedback types, i.e. classification labels, whereas M_5^G - M_7^G are computed using confidence scores.

6 RESULTS AND DISCUSSION

In the absence of the ground-truth data, we proposed several metrics under $\mathbf{M^C}$ in Sec.5.1 that can be used to evaluate model performance. In order to show that $\mathbf{M^C}$ metrics can be trusted, we carry out correlation studies between $\mathbf{M^G}$ and $\mathbf{M^C}$ metrics. A good correlation would mean that some of $\mathbf{M^G}$ metrics can indeed be proxied by some of $\mathbf{M^C}$ metrics. To obtain correlations between $\mathbf{M^G}$ and $\mathbf{M^C}$ metrics, we build several models by varying three hyper-parameters, to study the effect of them on the metrics in $\mathbf{M^G}$ and $\mathbf{M^C}$. These three hyper-parameters are: (i) the structure learning algorithms *SLA*, (ii) the number of levels n_l that is used while discretizing the continuous variables into categorical variables with n_l levels, and (iii) the number of samples in the training data n_s. For computing correlations, we look at the trends of different metrics as function of all the models generated from varying those three hyper-parameters.

Before computing correlations, we first look at the trends of some of the $\mathbf{M^G}$ metrics (since we already curated a small ground-truth data set specifically for this work) for various models that we built. Fig. 2 presents the results for $\frac{M_i^G}{M_i^{G^u}}$ for all models. Here, i = 1..4 correspond to classification based metrics, and $M_i^{G^u}$ metric is computed between ground-truth feedback type and user feedback type t, and so it is independent of the model choice. The first panel in Fig. 2 shows the results by varying the structure learning algorithm (*SLA*) and the *SLA* Index 1..14 indicates the following algorithms: {`chow.liu, fast.iamb, gs, h2pc, hc, iamb, iamb.fdr, inter.iamb, mmhc, mmpc, pc.stable, rsmax2, si.hiton.pc, tabu`}, which are implemented in `bnlearn` [31, 32]. The variation with respect to the number of levels n_l, used in discretizing the continuous variables to discrete variables, is shown in the second panel, where we vary n_l from 1..19. The third panel displays the variations in the number of samples n_s in the training data set, where we vary it from $1 \times 10^3 - 7.5 \times 10^6$. In all three variations, we observe that none of those four $\mathbf{M^G}$ metrics outperform $\mathbf{M^{G^u}}$ metrics. This means that user's feedback t is more closer to the ground-truth compared with model predicted feedback type t', although with minute differences as the values of $\frac{M_i^G}{M_i^{G^u}}$ are close to 1. For classification purpose, the model may not even be needed, however, the purpose of this work is to establish the confidence scores on the user's feedback which is not possible to get directly from users.

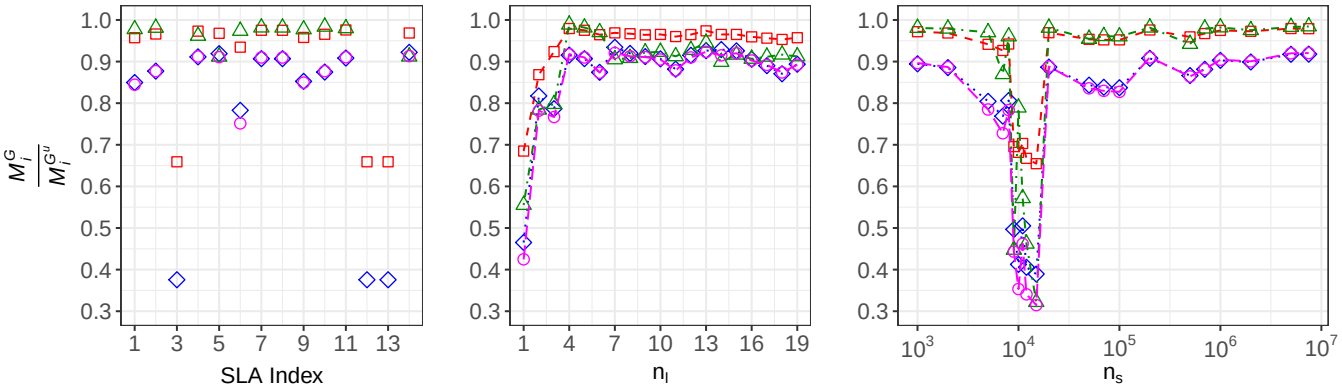

Figure 2: Results for normalized M_i^G, where $i = 1..4$, for all the models generated by varying three different hyper-parameters, shown in three panels: (i) first panel: Structure learning algorithm (*SLA*) index with 14 *SLAs* (ii) second panel: number of levels used while discretizing continuous variables to discrete variables, n_l with values 1..19, and (iii) third panel: number of samples in the training data set, n_s with values $1 \times 10^3 - 7.5 \times 10^6$. $M_i^{G^u}$ is computed between the ground-truth feedback type and the user feedback type t, and is constant for each i.
Markers: □ : $(i = 1)$, ◇ : $(i = 2)$, △ : $(i = 3)$, ○ : $(i = 4)$

For the *SLA* variation experiment, we fix $n_s = 2 \times 10^6$ and $n_l = 5$. From the *SLA* variation experiment, we find that `gs, iamb, si.hiton.pc`, which are constraint-based algorithms and `rsmax2` which is a hybrid method, do not perform well for our problem, if we consider M_1^G, M_2^G and M_4^G metrics. Depending on what metric we consider, there are different best models. However, considering the mean $F1$ score, i.e. M_4^G, `iamb.fdr, inter.iamb, pc.stable`, which are constraint-based algorithms, `hc, tabu`, which are score-based algorithms and `h2pc`, which is a hybrid algorithm, perform well for our problem. To make the selection of the best *SLA*, we look at the training time and we find the following training times in AWS's `m5.24xlarge` EC2 instance: `iamb.fdr`: 140s, `inter.iamb`: 71s, `pc.stable`: 570s, hc: 92s, tabu: 120s and h2pc: 1,710s. As a result, `inter.iamb` is the best choice for *SLA*. For the n_l variation experiment, we fix $n_s = 2 \times 10^6$ and *SLA* to be `inter.iamb`. We see that the values of all four metrics increase initially, and then they level-off with minor fluctuations, with elbow point around $n_l = 5$. Based on this observation, we pick $n_l = 5$ in other experiments as higher n_l increases the computational burden. For the n_s variation experiment, we fix *SLA* to be `inter.iamb`, and find that all four metrics do not change much with n_s, except for data points around $n_s = 1 \times 10^4$.

Having discussed the results based on $\mathbf{M^G}$ metrics, we now present results on $\mathbf{M^C}$ metrics, and also look at the correlations. Fig. 3 presents the patterns of these two normalized metrics as a function of the model index in the top three panels. Here, the model index 1...14 refers to all the *SLAs*, in the same order discussed above for Fig. 2, and 15...33 model index refers to n_l variations with $n_l = 1...19$, and the index 34...54 refers to n_s variations with n_s in the range of $1 \times 10^3 - 7.5 \times 10^6$. A table, listing all three hyper-parameters for all models, can be found in the Appendix. We can see negative correlations between M_8^C and M_6^G for all three experiments. This is expected because as the model becomes better, the mean of the differences between confidence scores of the ground-truth feedback type and the user feedback type should increase, but the mean of the confidence scores of user's no feedback in the ENS test set should decrease. To quantify the correlation, we compute Spearman's rank correlation coefficient r between the two metrics. For the pair $M_8^C - M_6^G$, we find $r = -0.90$ (p-value: 1×10^{-7}), -0.67 (p-value: 2.4×10^{-3}), -0.80 (p-value: 1.6×10^{-5}) for *SLA*, n_l and n_s variation experiments, respectively. We can also combine all the models from three different variation experiments to get 54 models or data points to compute r, and we find $r = -0.91$ (p-value: $< 1 \times 10^{-8}$). We present another example for the pair $M_3^C - M_7^G$ in the bottom three panels in Fig. 3, where M_3^C is the accuracy between user feedback type t and model predicted feedback type t', and M_7^G is the mean of the confidence scores on ground-truth feedback type, and we see positive correlations for all three experiments. Again, as the model gets better, both M_3^C and M_7^G should increase. We find $r = 0.79$ (p-value: 7.2×10^{-4}), 0.53 (p-value: 0.022), 0.80 (p-value: 1.4×10^{-5}) for *SLA*, n_l and n_s variation experiments, respectively. Considering all 54 models, for this pair, we find $r = 0.87$ (p-value: $< 1 \times 10^{-8}$). We also note that the best models found for all three variation experiments, based on $\mathbf{M^G}$ metrics in Fig. 2, are also the best models found using $\mathbf{M^C}$ metrics. For example, for n_l variation experiment, we see the best results around $n_l = 5$ for both $\mathbf{M^G}$ and $\mathbf{M^C}$ metrics. Therefore, we can conclude that the $\mathbf{M^C}$ metrics are indeed useful in determining the optimal values for these hyper-parameters.

Considering all 54 models, we computed r for all 77 pairs between $\mathbf{M^G}$ and $\mathbf{M^C}$. The results for r for all those pairs are shown in Fig. 4. The statistically significant correlations that have p-values less than or equal to 0.05 are shown using the black font color, and we find 41 such pairs. Out of those 41 pairs, there are 10 pairs for which $|r| > 0.7$ with p-values $< 1 \times 10^{-9}$ (highlighted using the black box in Fig. 4) that we reckon point to strong correlations.

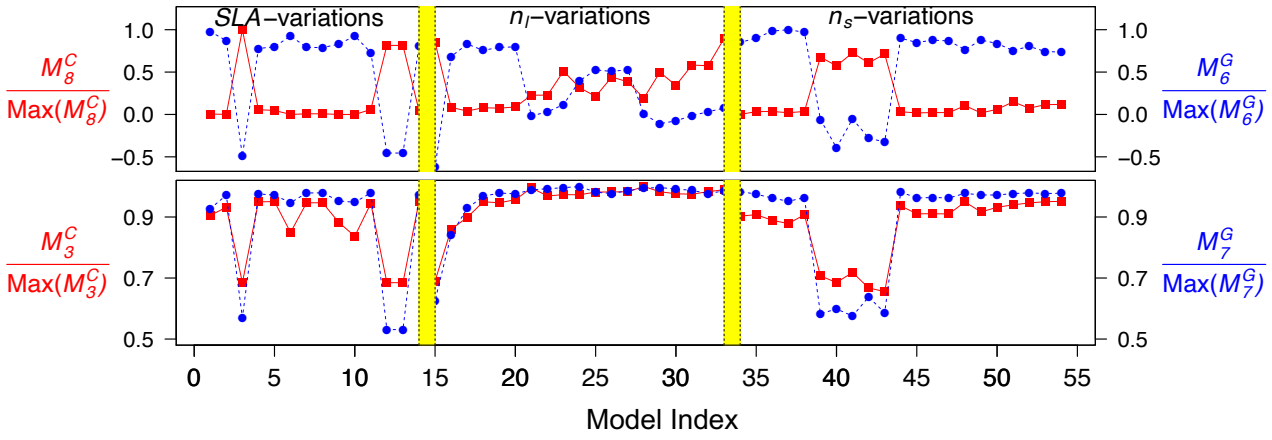

Figure 3: Top panels: Normalized M_8^C and M_6^G for all models generated by varying three different hyper-parameters: (i) Structure learning algorithm (*SLA*), (ii) number of levels used while discretizing continuous variables to discrete variables, n_l, and (iii) number of samples in the training data set, n_s, with a common *x-axis* indicating the Model Index explained in the text. Bottom panels: Normalized M_3^C and M_7^G for all models generated by varying three different hyper-parameters with a common *x-axis*. The normalization is done by simply diving the values by maximum of respective array.
Markers: CVF based metric, Ground-truth based metric.

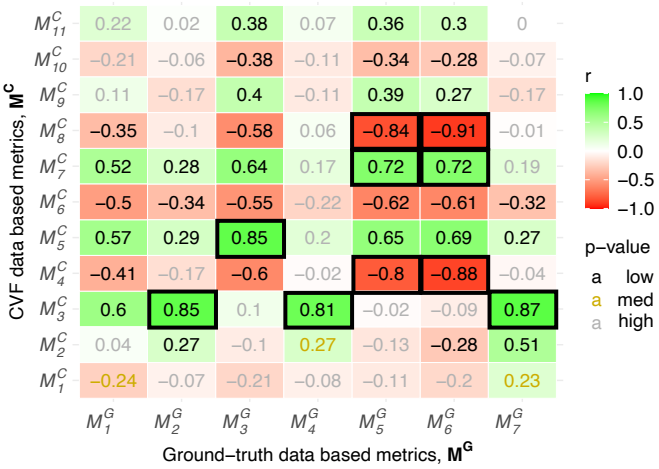

Figure 4: Results for Spearman's rank correlation coefficient r between all 77 $\mathbf{M^G}$ and $\mathbf{M^C}$ metrics pairs. The color scale on cells indicates r with green as $r = 1$, red as $r = -1$ and white as $r = 0$. The p-value for each r statistic is indicated using three font colors in each cell, with labels "low" (black) as p-value ≤ 0.05, "med" (gold) as $0.05 <$ p-value ≤ 0.1 and "high" (grey) as p-value > 0.1. Cells that are highlighted within black box refer to pairs with $|r| > 0.7$ with p-values $< 1 \times 10^{-9}$.

On those ten strongly correlated pairs, we see six $\mathbf{M^G}$ metrics: (i) M_2^G, (ii) M_3^G, (iii) M_4^G, (iv) M_5^G, (v) M_6^G, and (vi) M_7^G and five $\mathbf{M^C}$ metrics: (i) M_3^C, (ii) M_4^C, (iii) M_5^C, (iv) M_7^C, and (v) M_8^C. This means that we can rely on those five $\mathbf{M^C}$ metrics (in the absence of ground-truth data) *only* if the true success metric to measure the model performance is one (or more) of those six $\mathbf{M^G}$ metrics. For example, if the objective is to maximize the mean of ROC AUC across all feedback types (M_5^G), as if we had ground-truth data, then we can proxy that using one of the three non-ground-truth based metrics M_4^C, M_7^C or M_8^C. Since we see the highest correlation of M_5^G with M_8^C, which is the mean of the confidence scores on user feedback in the ENS test set, we recommend to use M_8^C. However, if the objective is to optimize some other metrics (from M_2^G, M_3^G, M_4^G, M_5^G, M_6^G, M_7^G), then Fig. 4 can be used to select the most reasonable $\mathbf{M^C}$ choice.

7 CONCLUSIONS

We proposed a way to leverage a BBN model to establish confidence scores for crowdsourced voice feedback (CVF) data. We carried out several experiments by varying different hyper-parameters, such as *SLA*, n_l and n_s, and presented their effects on the model performance using some existing and new metrics identified in this paper. We discussed 11 metrics that can be computed without ground-truth data ($\mathbf{M^C}$ metrics), and merely using the CVF data. We studied correlations of those 11 metrics with 7 ground-truth based metrics ($\mathbf{M^G}$, for which we curated a small ground-truth test set), where all those metrics were computed while carrying out hyper-parameter experiments mentioned above. On those 77 pairs, we found 10 metrics pairs between $\mathbf{M^C}$-$\mathbf{M^G}$ that had strong correlations with absolute value of Spearman's rank correlation greater than 0.7. Using the correlation results, we showed that one or more of five $\mathbf{M^C}$ metrics (M_3^C, M_4^C, M_5^C, M_7^C, and M_8^C) can be used to measure the model performance in the absence of the ground-truth data, depending on what ground-truth metrics (such as (i) mean of precision, (ii) mean of recall, (iii) mean of F1, (iv) mean of ROC AUC, across all feedback types, and (v) mean of differences in confidence scores between ground-truth label and user feedback

label, and (vi) mean of confidence scores on ground-truth label) we want to optimize.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their insightful comments.

REFERENCES

[1] Hirotogu Akaike. 1973. *Information Theory and an Extension of the Maximum Likelihood Principle.* Springer New York, New York, NY, 199–213.

[2] Constantin F. Aliferis, Alexander Statnikov, Ioannis Tsamardinos, Subramani Mani, and Xenofon D. Koutsoukos. 2010. Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classification Part I: Algorithms and Empirical Evaluation. *Journal of Machine Learning Research* 11, 7 (2010), 171–234.

[3] Irad Ben-Gal. 2008. *Bayesian Networks.* American Cancer Society. https://doi.org/10.1002/9780470061572.eqr089

[4] Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics* 22, 1 (1996), 39–71. https://www.aclweb.org/anthology/J96-1002

[5] Dheeraj Bhaskaruni, Fiona Moss, and Chao Lan. 2018. Estimating Prediction Qualities without Ground Truth: A Revisit of the Reverse Testing Framework. 49–54. https://doi.org/10.1109/ICPR.2018.8545706

[6] Pierce Burke and Richard Klein. 2020. Confident in the Crowd: Bayesian Inference to Improve Data Labelling in Crowdsourcing. 1–6. https://doi.org/10.1109/SAUPEC/RobMech/PRASA48453.2020.9041099

[7] Pengfei Chen, Ben Ben Liao, Guangyong Chen, and Shengyu Zhang. 2019. Understanding and Utilizing Deep Neural Networks Trained with Noisy Labels. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 1062–1070.

[8] C. Chow and C. Liu. 1968. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory* 14, 3 (1968), 462–467. https://doi.org/10.1109/TIT.1968.1054142

[9] Diego Colombo and Marloes H. Maathuis. 2014. Order-Independent Constraint-Based Causal Structure Learning. *Journal of Machine Learning Research* 15, 116 (2014), 3921–3962. http://jmlr.org/papers/v15/colombo14a.html

[10] Mohamad Dolatshah, Mathew Teoh, Jiannan Wang, and Jian Pei. 2018. Cleaning crowdsourced labels using oracles for statistical classification. *Proceedings of the VLDB Endowment* (2018), 376–389.

[11] Maksym Fedorchuk and Bart Lamiroy. 2017. Binary Classifier Evaluation Without Ground Truth. In *2017 Ninth International Conference on Advances in Pattern Recognition (ICAPR)*. 1–6. https://doi.org/10.1109/ICAPR.2017.8593175

[12] Nir Friedman, Iftach Nachman, and Dana Pe'er. 1999. Learning Bayesian Network Structure from Massive Datasets: The "Sparse Candidate" Algorithm. 206–215. https://doi.org/10.13140/2.1.1125.2169

[13] Nir Friedman and Zohar Yakhini. 1996. On the Sample Complexity of Learning Bayesian Networks. In *Proceedings of the Twelfth International Conference on Uncertainty in Artificial Intelligence* (Portland, OR) *(UAI'96)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 274–282.

[14] Maxime Gasse, Alex Aussem, and Haytham Elghazel. 2014. A hybrid algorithm for Bayesian network structure learning with application to multi-label learning. *Expert Systems with Applications* 41, 15 (Nov 2014), 6755–6772. https://doi.org/10.1016/j.eswa.2014.04.032

[15] Søren Højsgaard. 2012. Graphical Independence Networks with the gRain Package for R. *Journal of Statistical Software* 46, 10 (2012), 1–26. https://doi.org/10.18637/jss.v046.i10

[16] Hui Jiang. 2005. Confidence measures for speech recognition: A survey. *Speech Communication* 45, 4 (2005), 455–470. https://doi.org/10.1016/j.specom.2004.12.004

[17] Kaustubh Kalgaonkar, Chaojun Liu, Yifan Gong, and Kaisheng Yao. 2015. Estimating confidence scores on ASR results using recurrent neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 4999–5003. https://doi.org/10.1109/ICASSP.2015.7178922

[18] Shubham Kapoor and Caglar Tirkaz. 2019. Bootstrapping NLU Models with Multi-task Learning.

[19] Yelin Kim, Joshua Levy, and Yang Liu. 2020. Speech Sentiment and Customer Satisfaction Estimation in Socialbot Conversations. In *Proc. Interspeech 2020*. 1833–1837. https://doi.org/10.21437/Interspeech.2020-2890

[20] Daphne Koller and Nir Friedman. 2009. *Probabilistic Graphical Models: Principles and Techniques.* MIT Press.

[21] Steffen L. Lauritzen. 1996. *Graphical Models.* Oxford University Press.

[22] Qiujia Li, Preben Ness, Anton Ragni, and Mark Gales. 2019. Bi-directional Lattice Recurrent Neural Networks for Confidence Estimation. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 6755–6759. https://doi.org/10.1109/ICASSP.2019.8683488

[23] Dimitris Margaritis. 2003. *Learning Bayesian Network Model Structure From Data.* Ph.D. Dissertation. Carnegie Mellon University.

[24] Curtis G. Northcutt, Lu Jiang, and Isaac L. Chuang. 2021. Confident Learning: Estimating Uncertainty in Dataset Labels. *J. Artif. Intell. Res.* 70 (2021), 1373–1411.

[25] Alexandros Pappas and Duncan F. Gillies. 2002. A New Measure for the Accuracy of a Bayesian Network. In *MICAI 2002: Advances in Artificial Intelligence.* Springer Berlin Heidelberg, Berlin, Heidelberg, 411–419.

[26] Judea Pearl. 2009. *Probabilistic reasoning in intelligent systems : Networks of plausible inference.* Morgan Kaufmann, San Francisco, Calif. Example for explaining away.

[27] Jose M. Peña. 2008. Learning Gaussian Graphical Models of Gene Networks with False Discovery Rate Control. In *Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, Elena Marchiori and Jason H. Moore (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 165–176.

[28] Stuart J. Russell and Peter Norvig. 2009. *Artificial Intelligence: A modern approach* (3 ed.). Pearson.

[29] Ruhi Sarikaya, A. Paul Crook, Alex Marin, Minwoo Jeong, Jean-Philippe Robichaud, Asli Çelikyilmaz, Young-Bum Kim, Alexandre Rochette, Zia Omar Khan, Xiaohu Liu, Daniel Boies, Tasos Anastasakos, Zhaleh Feizollahi, Nikhil Ramesh, H. Suzuki, Roman Holenstein, Elizabeth Krawczyk, and Vasiliy Radostev. 2016. An overview of end-to-end language understanding and dialog management for personal digital assistants. *2016 IEEE Spoken Language Technology Workshop (SLT)* (2016), 391–397.

[30] Gideon Schwarz. 1978. Estimating the Dimension of a Model. *The Annals of Statistics* 6, 2 (1978), 461 – 464. https://doi.org/10.1214/aos/1176344136

[31] Marco Scutari. 2010. Learning Bayesian Networks with the bnlearn R Package. *Journal of Statistical Software* 35, 3 (2010), 1–22. https://doi.org/10.18637/jss.v035.i03

[32] Marco Scutari. 2017. Bayesian Network Constraint-Based Structure Learning Algorithms: Parallel and Optimized Implementations in the bnlearn R Package. *Journal of Statistical Software* 77, 2 (2017), 1–20. https://doi.org/10.18637/jss.v077.i02

[33] Ioannis Tsamardinos, Constantin Aliferis, and A Statnikov. 2003. Time and Sample Efficient Discovery of Markov Blankets And Direct Causal Relations. (04 2003), 673–678. https://doi.org/10.1145/956750.956838

[34] Ioannis Tsamardinos, Constantin F. Aliferis, and Alexander R. Statnikov. 2003. Algorithms for Large Scale Markov Blanket Discovery.. In *FLAIRS Conference*, Ingrid Russell and Susan M. Haller (Eds.). AAAI Press, 376–381.

[35] Ioannis Tsamardinos, Laura Brown, and Constantin Aliferis. 2006. The Max-Min Hill-Climbing Bayesian Network Structure Learning Algorithm. *Machine Learning* 65 (10 2006), 31–78. https://doi.org/10.1007/s10994-006-6889-7

[36] Fan Yang, Mengnan Du, and Xia Hu. 2019. Evaluating Explanation Without Ground Truth in Interpretable Machine Learning. *arXiv e-prints*, Article arXiv:1907.06831 (July 2019), arXiv:1907.06831 pages. arXiv:1907.06831 [cs.LG]

[37] Sandeep Yaramakala and Dimitris Margaritis. 2005. Speculative Markov Blanket Discovery for Optimal Feature Selection. In *Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM '05)*. IEEE Computer Society, USA, 809–812. https://doi.org/10.1109/ICDM.2005.134

A MODEL HYPER-PARAMETERS

Table 3: List of hyper-parameters for all models generated.

Model Index	*SLA*	n_l	n_s
1	chow.liu	5	2×10^6
2	fast.iamb	5	2×10^6
3	gs	5	2×10^6
4	h2pc	5	2×10^6
5	hc	5	2×10^6
6	iamb	5	2×10^6
7	iamb.fdr	5	2×10^6
8	inter.iamb	5	2×10^6
9	mmhc	5	2×10^6
10	mmpc	5	2×10^6
11	pc.stable	5	2×10^6
12	rsmax2	5	2×10^6
13	si.hiton.pc	5	2×10^6
14	tabu	5	2×10^6
15	inter.iamb	1	2×10^6
16	inter.iamb	2	2×10^6
17	inter.iamb	3	2×10^6
18	inter.iamb	4	2×10^6
19	inter.iamb	5	2×10^6
20	inter.iamb	6	2×10^6
21	inter.iamb	7	2×10^6
22	inter.iamb	8	2×10^6
23	inter.iamb	9	2×10^6
24	inter.iamb	10	2×10^6
25	inter.iamb	11	2×10^6
26	inter.iamb	12	2×10^6
27	inter.iamb	13	2×10^6
28	inter.iamb	14	2×10^6
29	inter.iamb	15	2×10^6
30	inter.iamb	16	2×10^6
31	inter.iamb	17	2×10^6
32	inter.iamb	18	2×10^6
33	inter.iamb	19	2×10^6
34	inter.iamb	5	1×10^3
35	inter.iamb	5	2×10^3
36	inter.iamb	5	5×10^3
37	inter.iamb	5	7×10^3
38	inter.iamb	5	8×10^3
39	inter.iamb	5	9×10^3
40	inter.iamb	5	1×10^4
41	inter.iamb	5	1.1×10^4
42	inter.iamb	5	1.2×10^4
43	inter.iamb	5	1.5×10^4
44	inter.iamb	5	2×10^4
45	inter.iamb	5	5×10^4
46	inter.iamb	5	7×10^4
47	inter.iamb	5	1×10^5
48	inter.iamb	5	2×10^5
49	inter.iamb	5	5×10^5
50	inter.iamb	5	7×10^5
51	inter.iamb	5	1×10^6
52	inter.iamb	5	2×10^6
53	inter.iamb	5	5×10^6
54	inter.iamb	5	7.5×10^6