Qualitative Models of Data Generation Processes: Facilitating Data-Intensive AI Solutions

Gregor Pavlin, Kathryn B Laskey, Franck Mignet, Filip S. Slijkhuis, Erik Blasch, Valentina Dragos, J. Pieter de Villiers, Lennard Jansen

Qualitative Models of Data Generation Processes: Facilitating Data-Intensive AI Solutions

G. Pavlin[1], K. B. Laskey[2], F. Mignet [1], F. S. Slijkhuis [1], E. Blasch[3], V. Dragos[4], J.P. de Villiers[5], Lennard Jansen[1]

[1]Thales Research and Technology, Delft, The Netherlands, [gregor.pavlin, franck.mignet, filip.slijkhuis]@nl.thalesgroup.com
[2]George Mason University, Fairfax, VA, USA, klaskey@gmu.edu
[3]Air Force Research Lab, Rome, NY, USA, erik.blasch.1@us.af.mil
[4]ONERA-The French Aerospace Lab, Palaiseau, France, valentina.dragos@onera.fr
[5]University of Pretoria, Pretoria, South Africa, pieter.devilliers@up.ac.za

***Abstract*—AI-based decision support solutions require life cycles that adequately address critical steps, such as (i) finding suitable machine learning (ML) methods for the problem at hand, (ii) preparing and executing adequate data acquisition processes and (iii) tractable evaluation of the overall solution. Understanding the data generating processes is key in achieving this. Training and test data can be seen as a result of a causal data generation process, a sampling process in which the data is collected from different sources that are influenced by multiple interdependent phenomena. This is represented by a Qualitative Model of Data Generation Processes (QM-DGP), a causal graphical model. QM-DGP facilitates analysis of the complexity of the underlying data generating processes that can inform the development of trustable ML-based solutions in multiple ways. Firstly, this analysis is the basis for the determination of the required complexity of the ML models. Secondly, it facilitates the determination of the quantities of training data supporting good learning results. Thirdly, it can provide guidance for a systematic simplification of the models, supporting tractable solutions without significantly reduced performance. The construction of QM-DGP and the analysis benefit from sound theoretical concepts, such as d-separation and I-Maps. Experimental results with simulated data indicate that the approach can be effective in predicting the required quantities of training data and the determination of the modelling complexity using different types of models.**

I. Introduction

Modern decision support systems require increasingly complex fusion solutions relying on Artificial Intelligence (AI) approaches that make use of Machine Learning (ML). The use of such technologies, however, introduces substantial challenges to the development, evaluation and deployment of mission critical fusion solutions [1], [2], [3], [4]. Some of the main challenges are the choice of the modelling approaches, machine learning methods and reasoning mechanisms. As emphasized in [1], this is not trivial, and it critically influences the entire development life cycle as well as the properties of the resulting fusion solutions. Appropriately chosen representations, associated reasoning mechanisms and ML techniques facilitate the implementation and understanding of the required functions that perform within specific operational constraints. However, if the selected methods do not match the characteristics of the problem at hand, insurmountable obstacles can be introduced into the entire life cycle, impacting the development process as well as the system deployment. Given the complexity of the targeted problems, many aspects must be taken into account when making choices related to the modeling, ML and inference throughout the development life cycle. A URREF-driven life cycle [1] approach was introduced to address these challenges. According to this approach, different representation and reasoning aspects must be evaluated in the inception, design, implementation, commissioning and operational phases. In the inception phase the following questions must be answered: What types of functions can in principle be automated? What types of models are suitable, given the operational conditions? Can models supporting sufficiently accurate inference be created for the given application? Can sufficient amounts of training data be obtained for achieving the needed accuracy? In the design phase, architects need answers to additional questions: What is the best model architecture that supports tractable ML and minimizes the required amounts of training data? How do we gather sufficient amounts of data for the given application? In the implementation and evaluation phases, developers need to answer questions, such as: Can we reliably estimate the expected accuracy of the resulting solutions? Does the training data reflect all situations under which the trained models will operate? Is the solution robust?

Key to answering the above mentioned questions is understanding of the domain, in particular the mechanisms of the processes generating the data that is used for training of the models as well as inputs to the decision support functions at runtime. This paper proposes the concept of Qualitative Models of Data Generating Processes (QM-DGP) that describe causal dependencies between different phenomena influencing the generation of observations in a qualitative manner. The approach is inspired by the principles expressed in [5] and can be viewed as an extension of the Abstraction flow concepts introduced in [6]. In particular, QM-DGP supports modelling of real world entities and processes (RWEPs), the basis for various types of abstractions and analyses throughout a system's life-cycle. The focus in this paper is complexity analysis of the correlations in the data as well as a coarse estimation of the quantities of data required for training of

models that operate satisfactory under all relevant conditions. The QM-DGP approach is agnostic with respect to the adopted modelling paradigm and the ML methods. The paper shows how the approach facilitates evaluations in different phases of the URREF-driven life-cycle. Controlled experiments with synthetic data are used to demonstrate the effectiveness of the approach for both Bayesian network models and Neural Networks.

A. Example Use Case

The concepts in this paper will be illustrated with the help of a synthetic "ground truth" example data generating process (DGP) that gives rise to the observed data in a use case, which simulates aircraft being sensed by a radar. For the sake of clarity, we focus on a snapshot in time that is governed by a probability distribution $\mathcal{P}(\mathcal{V})$ over the set of random variables $\mathcal{V} = \{Class, Flight, Position, Speed, RCS, WindDirection, Rain, Visibility, WindStrength\}$, defined as follows:

- States of $Class$ represent airliners, small planes (general aviation) and helicopters.
- States of $Flight$ denote possible flight phases: depart, climb, descent, approach, holding and cruise for general aviation and cruise for airliners.
- States of $Position$ denote discrete areas in which aircraft execute various flight phases represented by $Flight$.
- $Speed$ represents speeds in four ranges.
- RCS represents the radar cross section in three ranges.
- $WindDirection$ represents easterly/westerly winds.
- States of $Rain$ represent rain/no rain, respectively.
- Discrete variable $Visibility$ has states good, low, fog.
- $WindStrength$ represents normal, wind and storm.

Fig. 1 shows the directed acyclic graph describing the dependencies in $\mathcal{P}(\mathcal{V})$ influencing the DGP. While the model is a severe simplification, it captures typical dependencies over the set of variables. The emphasis is on significant influences the context exerts on the distribution $\mathcal{P}(\mathcal{V})$. Various external conditions can change the DGP, resulting in major distribution shifts in the sampled data. E.g. the states of $Speed$ directly depend on the $Class$ and the flight phase $Flight$. Moreover, $Position$ directly depends on the $Class$, the $Flight$ phase and the $WindDirection$. Airliners cruise at a much greater altitude than small planes. $Position$ also depends on the $WindDirection$ that determines the landing and departure directions. RCS depends on the $Class$ (the plane size and shape) and the relative position of the plane wrt. the radar. An airliner cruising at a high altitude will inevitably result in small values for RCS due to great distance between the radar and the plane. Small planes could have relatively great RCS when in the vicinity of the airport.

II. Approach

This paper assumes that the training data $\mathcal{D}_{train}$ is a result of a causal data generation process (DGP) that can be seen as a sampling process from some distribution $P(\mathcal{V})$ over correlated variables $\mathcal{V}$. $P(\mathcal{V})$ influences the frequencies of observations during the data sampling processes.

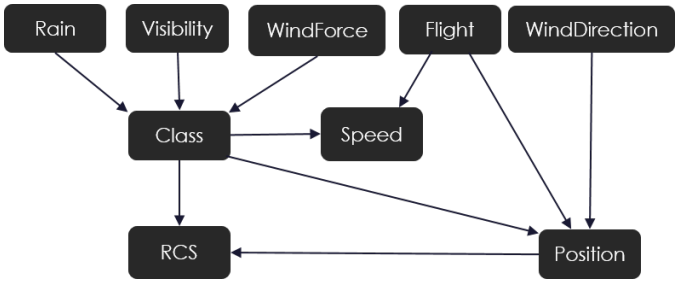

Fig. 1. A directed acyclic graph encoding dependencies between phenomena.

A. Complexity of Correlations

The complexity of a trained model $\mathcal{M}_k$ defined over a set of variables $\mathcal{V}_k \subset \mathcal{V}$ can be expressed as the number of parameters required to specify the model. While $\mathcal{M}_k$ should be as simple as possible, it must be rich enough, i.e. be defined through a sufficient number of parameters, to capture the relevant knowledge about the relations between variables $\mathcal{V}_k$ from $\mathcal{D}_{train}$ during an ML process. The required model complexity depends on the complexity of the relations between variables $\mathcal{V}$ reflected in $P(\mathcal{V})$. The complexity of $P(\mathcal{V})$, in turn, can be expressed with a Qualitative Model of a Data Generation Process (QM-DGP). This is a directed acyclic graph that captures the relations in the data generating process in a qualitative manner. That is, the model contains nodes representing the phenomena of interest and directed links representing direct dependencies between these phenomena; but no strength of these dependencies is considered. Figure 1 shows an example of such a model. A QM-DGP accurately represents a DGP over set of variables $\mathcal{V}$ if it is an Independence Map (I-MAP) of the underlying probability distribution $P(\mathcal{V})$ [7]. In that case we say that QM-DGP is *faithful* with respect to the underlying distribution $P(\mathcal{V})$.

Given that a QM-DGP is faithful as a representation of the data generating process, its topology reveals useful information about the process complexity. An important pattern is a Collider, a variable X_i associated with two or more parent variables $\pi(X_i)$. Variables in $\pi(X_i)$ directly influence the states of X_i; $\pi(X_i)$ define the local context of X_i. Colliders can greatly increase modelling complexity, i.e. a large number of parameters that often requires a large set of training data to learn with reasonable accuracy. Namely, a collider variable X_i is associated with a set of parameters whose size increases exponentially with the number of its parent nodes $\pi(X_i)$ unless modeling assumptions are imposed to control the number of parameters. For simplicity of exposition it is assumed initially that all variables are discrete.[1] In this case, we can assume that the DGP samples states of X_i from a discrete distribution $P(X_i|\pi(X_i))$. The number of possible combinations of states n_{X_i} over which the distribution is defined is expressed as:

$$n_{X_i} = |X_i| \prod_{Y_j \in \pi(X_i)} |Y_j|, \tag{1}$$

[1] We consider relaxing this assumption later.

where $|X_i|$ and $|Y_j|$ denote the number of states of variables X_i and Y_j, respectively. This equation expresses the complexity of the model describing relations between X_i and its parents $\pi(X_i)$. In the model shown in Fig. 1, we can identify multiple colliders: $Speed$, $Position$, $Class$ and RCS for which (1) applies. Given the number of states of each of these variables and their parents, the estimation of the corresponding conditional probabilities:

$P(Speed|Class, Flight, WindForce)$,
$P(Position|Class, Flight, WindDirection)$,
$P(RCS|Class, Position)$ and
$P(Class|Rain, Visibility, WindForce)$

requires observation of 144, 432, 108 and 6 different combinations of states, respectively. It is necessary to have a sufficient number of observations of each combination of state for reliable estimation of the real-valued parameters representing the models of such correlations. Section II-D considers this question.

B. Local Context

Parents $\pi(X_i)$ define the *local context* of X_i. That is, the sampled values of the parents $\pi(X_i)$ determine the probability distribution from which X_i is sampled. We call a specific combination of states of variables $Y_i \in \pi(X_i)$ a *local context*.

The cardinality of $|\pi(X_i)|$ and the number of states of each variable $Y_i \in \pi(X_i)$ and their correlations together determine the context complexity, i.e. the number of distinct contexts leading to manifestly different distributions over the states of X_i. The number of local contexts c_{X_i} can be expressed as a product

$$c_{X_i} = \prod_{Y_j \in \pi(X_i)} |Y_j| \tag{2}$$

It is essential to train the models on data $\mathcal{D}_{train}$ that was sampled under all relevant local contexts for each variable $X_i \in \mathcal{V}$. Otherwise, the trained models are likely to fail/underperform when fed with data sampled in a context that was not considered at the collection of $\mathcal{D}_{train}$.

The variable $Class$ in Fig. 1 has 12 distinct local contexts, given the local context set $\{Rain, WindStrength, Visibility\}$. Without additional assumptions, accurately estimating $P(Class|Rain, WindStrength, Visibility)$ would therefore require 12 times as much data as required for an estimation of the prior distribution $P(Class)$.

Fortunately, structural properties of $\mathcal{P}(\mathcal{V})$ may permit estimation of the distribution using much less data. For example, when the local distribution of X_i satisfies context-specific independence (CSI) [8], then subsets of different combinations of states of $\pi(X_i)$ can be associated with the same distribution $P(X_i|\pi(X_i))$. Another example is independence of causal influence, or ICI [9]. Common ICI models include Disjunctive Interaction models (Noisy-OR/Noisy-MAX) [7]. CSI and ICI are examples of parametric regression models in which the parent-child relationship is expressed as a parametric model. The amount of data required to estimate the parameters of a parametric model is often much less than required for a separate distribution per context.

In the DGP of Fig. 1, it reasonable to assume that the relationship of $Class$ to its parents satisfies CSI. While there exist 12 possible state combinations for the set $Visibility$, $WindForce$ and $Rain$, they result in only three distinct distributions over states of the variable $Class$. Therefore, we need to ensure that we sample $Class$ with sufficient representation from three these subsets, but ensuring sufficient representation from each combination of parent states in each subset is unnecessary.

C. Expressing the DGP complexity

This section provides a high-level analysis of the DGP complexity that serves as the basis of a heuristic approach to estimating the quantities of data required for learning adequate models for the task at hand. Let's assume a learning task to obtain a classifier using model $\mathcal{M}_k$ defined over variables $\mathcal{V}_k \subseteq \mathcal{V}$. The model relates $X_c \in \mathcal{V}_k$ and the observed variables $X_i \in \mathcal{O} \subset \mathcal{V}_k$, where $\mathcal{O}$ denotes the set of all observed features and $\mathcal{V}$ is the set of variables corresponding to the phenomena that occur in the underlying DGP. The classifier uses $\mathcal{M}_k$ to predict the states of the classification variable X_c given observations of the features represented by variables $X_i \in \mathcal{O}$.

Under the assumption that the QM-DGP is, to good approximation, a faithful representation of direct dependencies between the variables in $\mathcal{V}$, it is possible to provide a rough estimate of the quantities of training data required for training of a model $\mathcal{M}_k$, for given variables $X_i \in \mathcal{V}$. To estimate the needed quantities of data for training $\mathcal{M}_k$, we must consider the complexity of the underlying DGP, i.e. $P(\mathcal{V})$ from which the states of variables in $\mathcal{V}_k$ are sampled. **The positions of X_c and the observed variables $X_i \in \mathcal{O}$ in the QM-DGP graph reveal a lot about the complexity of the learning task at hand.**

To understand the DGP complexity, we introduce the notion of *DGP context variables*. In common parlance, the term context refers to "the interrelated conditions in which something exists or occurs".[2] The DGP context variables $\mathcal{C} \in \mathcal{V}$ in a QM-DGP are intended to represent the background conditions under which the process operates. Generally, a DGP context variable $Y_i \in \mathcal{C}$ is either a root node in the DGP graph or, if not a root node, has no non-context nodes as parents. In addition, a DGP context variable will not be d-separated [7] from X_c or $X_i \in \mathcal{O}$ by other context variables. This means it has an influence on the class and/or feature variables that is not accounted for by the other context variables. The set of DGP context variables satisfying this condition for the set of observable variables $\mathcal{O}$ is called the DGP relevant context $\mathcal{C}_\mathcal{O} \subseteq \mathcal{C}$, i.e. for any $Y_j \in \mathcal{C}_\mathcal{O}$ there exists at least one path between Y_j and X_c or $X_i \in \mathcal{O}$. Such context variables influence the occurrence of the states of the variables that are represented in the trained model $\mathcal{M}_k$. Fig. 2.a shows an

[2]https://www.merriam-webster.com/dictionary/context

example where $\mathcal{O} = \{X_1, X_2\}$ and the corresponding relevant DGP context is defined over $\mathcal{C}_\mathcal{O} = \{Y_1, Y_2, Y_3, Y_4, Y_5\}$. **Note that the architecture of the trained model $\mathcal{M}_k$ could explicitly represent only X_c and the observed variables $X_i \in \mathcal{O}$, i.e. it would not contain variables from $\mathcal{C}_\mathcal{O}$ which are explicitly represented in the QM-DGP. Yet, whether included in $\mathcal{M}_k$ or not, the variables in $\mathcal{C}_\mathcal{O}$ drive the QM-DGP by influencing distributions at different places in the causal process, thus influencing the frequencies of combinations of observed states of X_c and $X_i \in \mathcal{O}$.** Thus, possible combinations of states in $\mathcal{C}_\mathcal{O}$ and states of variables in the trained model M_k must be explicitly considered when estimating the required size of the training sets.

Moreover, for each $\mathcal{V}_k$ we can obtain a reduced model from the full QM-DGP relating $X_i \in \mathcal{O}$, X_c and $C_\mathcal{O}$. Fig. 2.b shows such a reduced model that was obtained from the QM-DGP in Fig. 2.a. In case the conditional probabilities for all relations in the QM-DGP were known, the model shown in Fig. 2.b could be obtained through marginalization of all latent variables along the paths connecting $\mathcal{C}_\mathcal{O}$, X_c and $X_i \in \mathcal{O}$ in Fig. 2.a. If all context variables $\mathcal{C}_\mathcal{O}$ are removed from the reduced model, we obtain a core model defined over X_c and $\mathcal{O}$ with joint probability distribution $P(X_c, \mathcal{O})$. In other words, $P(X_c, \mathcal{O})$ is embedded in the set of relevant DGP context variables $\mathcal{C}_\mathcal{O}$; i.e. there exist different versions of $P(X_c, \mathcal{O})$, each corresponding to a different context. In Fig. 2.b joint distribution $P(X_c, X_1, X_2)$ over the core model is embedded in the set of relevant DGP contexts defined over variables $\mathcal{C}_\mathcal{O} = \{Y_1, Y_2, Y_3, Y_4, Y_5\}$.

Let n_{core} denote the number of combinations of states over which the $P(X_c, \mathcal{O})$ of a core model is defined. Then the number of different combinations of states $n_{\mathcal{V}_k}$ of X_c, $\mathcal{O}$ and the relevant DGP context $\mathcal{C}_\mathcal{O}$ in the reduced model can be expressed as follows:

$$n_{\mathcal{V}_k} = n_{core} \prod_{Y_j \in \mathcal{C}_\mathcal{O}} |Y_j|, \tag{3}$$

where $|Y_j|$ denote the cardinality of the discrete DGP context variables. Note that equation (3) takes into account different contexts which may not be observed. Still, these variables must be considered as their states drive the DGP and thus influence the observable variables.

In the case of a fully connected QM-DGP, n_{core} is simply the product of the numbers of states of all variables X_c and $\mathcal{O}$. However, if the graph is sparsely connected, then fewer parameters are required to capture $P(X_c, \mathcal{O})$. Furthermore, if suitable learning methods are used, then subsets of the parameters could be learned independently of each other, requiring less training data. For example, consider a PGM $\mathcal{M}_k$ whose topology is identical to the DGP model in Fig. 1, with the set of feature variables $\mathcal{O} = \{Speed, RCS, Position\}$. This is also a core model featuring two fragments defined over $\mathcal{F}_1 = \{Class, RCS, Position\}$ and $\mathcal{F}_2 = \{Class, Speed\}$, respectively. These two fragments are conditionally independent given the states of variable $Class$, i.e. they are d-separated. Fragments $\mathcal{F}_1$ and $\mathcal{F}_2$ are associated with 108 and 12 combinations of states, respectively. For a given context the parameters in each fragment can be learned independently. Except in special cases, it is likely that more training data is required to learn $P(Class, RCS, Position)$ than $P(Class, Speed)$. Moreover, as each record $d_i \in \mathcal{D}_{train}$ contains observations of the states of variables from both fragments, i.e. $\mathcal{O} = \{Speed, RCS, Position\}$, the required number of data records is dictated by the more complex fragment. Therefore, in (3) we use $n_{core} = 108$, the number state combinations of the more complex fragment $\mathcal{F}_1$.

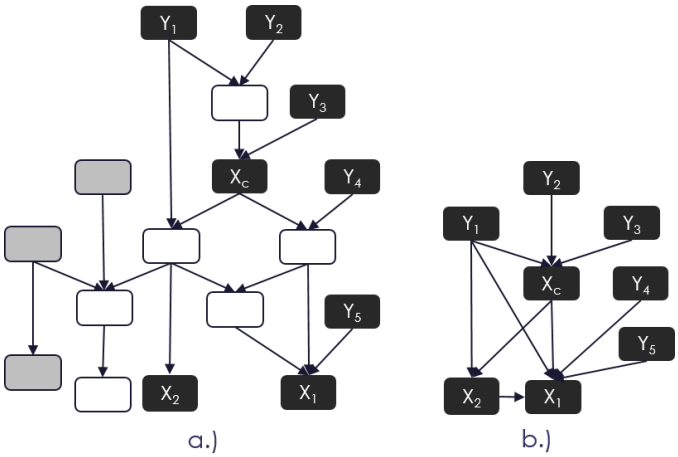

Fig. 2. a.) An example QM-DGP describing dependencies in $\mathcal{V}$. The black nodes correspond to the variables that are relevant for the estimation of the complexity of the relations between the class variable X_c and the observed variables X_1 and X_2 in the trained model $\mathcal{M}_k$. Gray nodes are observed context and feature variables that are irrelevant for inference over X_c. b.) A reduced model explicitly relating only $X_i \in \mathcal{O}$, X_c and $\mathcal{C}_\mathcal{O}$.

Thus far, we have considered only discrete variables with unconstrained probability distributions. Continuous variables are sometimes addressed with discretization, as was done for the model of Fig. 1. Here, we enounter a tension: a fine discretization improves accuracy of inference in the discretized model, but can dramatically increase the data required for learning an unconstrained distribution. This tension can be addressed using a suitable parametric model. Examples include the normal or exponential distribution, or mixtures thereof. Parametric models can generally be learned with many fewer data points than a fine discretization. Accuracy can sometimes be improved with mixtures of parameterized distributions. If discretization is needed, discretization can be performed after learning. Of course, this requires using an appropriate parameter estimation method. The methods of this paper assume unconstrained discrete distributions, but could be extended to parameterized continuous distributions.

D. Estimating the Required Data Quantities

The DGP complexity expressed with equation (3) is the basis for a heuristic approach to estimating the quantities of data required for learning adequate models for the classification task at hand. With equation (3) we can express a coarse

estimate of the required size of the training set $\mathcal{D}_{train}$ to train model $\mathcal{M}_k$:

$$|\mathcal{D}_{train}| = \kappa \cdot n_{\mathcal{V}_k}, \tag{4}$$

where κ denotes the minimum number of occurrences of a specific combination of states of involved variables, such that the corresponding parameter can be estimated to the desired accuracy. Note that the choice of κ is heuristic, but it is often possible to make a rough estimate of the expected accuracy in advance of data collection. In this paper $\kappa = 10$, i.e. it is assumed that $10 \cdot |X_i|$ measurements are required for a variable with $|X_i|$ states for a given local context, i.e. a specific combination of states of X_i's parents $\pi(X_i)$. As the number of states grows also the number of required observations grows, improving the chances of capturing multi-modal distributions that variables with many states can represent.

Equation (4) is a very crude estimate as it does not consider the strength of correlations in $P(\mathcal{V})$ from which DGP samples. It fails to account for the fact that some combinations of states might never materialize while other could be so rare that the chance of capturing a sufficient number of cases in the training data is very low. The distribution over these combinations depends on the phenomena corresponding to the X_i's ancestor variables in the QM-DGP. Ultimately, there exist one or more DGP context variables in $\mathcal{C}_{\mathcal{O}}$ that influence the values of $X_i \in \mathcal{O}$ and all variables on the paths between $\mathcal{C}_{\mathcal{O}}$ and X_i.

However, an improved estimate $|\mathcal{D}_{train}|'$ for the required data can be obtained, if some amount of training data is already available at the time of the analysis. We introduce an auxiliary PGM $\mathcal{M}_{aux}$ with the topology corresponding to the reduced QM-DGP, as for example the graph in figure 2.b. Such a M_{aux} is trained on the currently available data using simple Maximum Likelihood estimation based on state counting [3]. The learned $\mathcal{M}_{aux}$ encodes the frequencies of the state observations of all variables. For any context $\mathcal{C}(\mathcal{O})$ we can find the state combination with the smallest estimated probability. For each variable $Y_j \in \mathcal{C}(\mathcal{O})$ we find the state with the smallest marginal probability $P_{min}(Y_j)$ and determine the probability of the least likely combination of states in $\mathcal{C}(\mathcal{O})$:

$$P_{min}(\mathcal{C}(\mathcal{O})) = \prod_{Y_j \in \mathcal{C}(\mathcal{O})} P_{min}(Y_j). \tag{5}$$

With $P_{min}(\mathcal{C}(\mathcal{O}))$ and n_{core} we can compute factor $c(\mathcal{V}_k)$, a ratio between the number of required training cases to sufficiently cover all combinations of the states of the core model during learning and the expected number of occurrences of the least likely combination of $\mathcal{C}(\mathcal{O})$'s states in $\mathcal{D}_{train}$:

$$c(\mathcal{V}_k) = n_{core} \cdot \kappa / (|\mathcal{D}_{train}| \cdot P_{min}(\mathcal{C}(\mathcal{O}))). \tag{6}$$

This equation is relevant for simple learning approaches that cannot benefit from the underlying data structure during the training process. However, some training techniques can exploit structural properties of the graph and/or local distributions. If so, the analysis can be carried out on conditionally independent modelling fragments $\mathcal{F}_i$ and we can investigate whether the frequency of observations of their state combinations can support good machine learning. A fragment $\mathcal{F}_i$ is identified in $\mathcal{M}_{aux}$, given the class variable X_c and observations. Examples are $\mathcal{F}_1$ and $\mathcal{F}_2$ identified in the graph from Fig. 1, given the $Class$ variable (see section II-C). Such a fragment $\mathcal{F}_i$ is associated with its relevant context $\mathcal{C}(\mathcal{O}_i)$, where $\mathcal{O}_i$ denotes the set of observed variables in $\mathcal{F}_i$. Like in the derivation of (5), we determine for each variable $Y_j \in \mathcal{C}(\mathcal{O}_i)$ the state with the smallest marginal probability $P_{min}(Y_j)$ and determine the probability of the least likely combination of states in $\mathcal{C}(\mathcal{O}_i)$:

$$P_{min}(\mathcal{C}(\mathcal{O}_i)) = \prod_{Y_j \in \mathcal{C}(\mathcal{O}_i)} P_{min}(Y_j). \tag{7}$$

As next we determine the subset of variables $\mathcal{F}_i' \subseteq \mathcal{F}_i$ that are directly influenced by at least one variable $Y_j \in \mathcal{C}(\mathcal{O}_i)$ in the fragment's context. The set $\mathcal{F}_i'$ can be viewed as a hyper variable whose possible combinations of states $n_{\mathcal{F}_i'}$ can be determined as follows:

$$n_{\mathcal{F}_i'} = \prod_{Y_k \in \mathcal{F}_i'} |Y_k|. \tag{8}$$

With $P_{min}(\mathcal{C}(\mathcal{O}_i))$ and $n_{\mathcal{F}_i'}$ we can compute factor $c(\mathcal{F}_i)$ for a specific fragment, a ratio between the number of required training cases to sufficiently cover all combinations of $\mathcal{F}_i$'s states for learning and the expected number of occurrences of the least likely combination of $\mathcal{C}(\mathcal{O}_i)$'s states in $\mathcal{D}_{train}$:

$$c(\mathcal{F}_i) = n_{\mathcal{F}_i'} \cdot \kappa / (|\mathcal{D}_{train}| \cdot P_{min}(\mathcal{C}(\mathcal{O}_i))). \tag{9}$$

After executing this computation for all fragments in $\mathcal{M}_{aux}$, we take the greatest $c(\mathcal{F}_i)$ as a corrective factor to determine an adapted size of the training data considering the actual distributions of $\mathcal{M}_G$: $|\mathcal{D}_{train}|' = c_{max} \cdot |\mathcal{D}_{train}|$.

If in fragment $\mathcal{F}_i$ there exists a subset $\mathcal{F}_i'' = \mathcal{F}_i \setminus \mathcal{F}_i' \neq \emptyset$, then the same procedure can be repeated for $\mathcal{F}_i''$, by using $\mathcal{F}_i'$ as the context of $\mathcal{F}_i''$. By recursively repeating this procedure for all fragments $\mathcal{F}_i$ until $\mathcal{F}_i'' = \emptyset$, an improved estimate of the required training data size can be obtained, properly considering the dependencies throughout the DGP.

It may be that some of the context variables are controllable, either by setting them to particular values (e.g., choosing a setting on a sensor) or by collecting data when they are in certain states (e.g., waiting to collect data until given weather conditions occur). In such cases, if we determine that some combinations of conditions have insufficient data for reliable estimation, we can collect additional data for those circumstances and adjust the learned probabilities accordingly.

Also, the problem complexity can be controlled through the resolution of the variables used in the trained model. While reducing the resolution of variables results in smaller $|\mathcal{D}_{train}|$, we might be losing some precision and accuracy.

[3]Note that $M_{aux}$is not supposed to be used in the classification task.

III. Experimental Results

This section illustrates how the QM-DGP based analysis can facilitate the determination of the model architecture and the estimation of the required training data volumes. Known ground truth models $\mathcal{M}_G^i$ were used to sample training and testing data. This enabled objective comparison of different modelling paradigms with respect to the modelling complexity and the requirements for training good quality models.

A. Model Complexity

The QM-DGP provides guidance on the determination of the architecture, complexity and resolution of different types of ML-based models. To illustrate this different models were used. $\mathcal{M}_1$, $\mathcal{M}_2$ and $\mathcal{M}_3$ were PGMs whose topology is shown in Fig. 3. Their architecture was obtained from the graph in Fig. 1 by adding a $ContextSummary$ variable between the $Rain$, $WindStrength$, $Visibility$ and $Class$. $ContextSummary$ is a latent variable introducing the knowledge that there exist three groups of local contexts of $Class$. Also $\mathcal{M}_4$ was a PGM whose topology was defined over a subset of the nodes indicated by emphasized frames in Fig. 3. The PGMs were trained by using the Expectation Maximization (EM) algorithm. Moreover, $\mathcal{M}_5$ was a Multi-Layer Perceptron (MLP) with 3 hidden layers of 16 neurons each and leaky ReLU activation. While QM-DGP directly provides guidance on the suitable topology of the PGMs, there is no easy way of translating QM-DGP to a Neural Network architecture. Still, the QM-DGP can be helpful in finding a good model. Namely, by being able to estimate the necessary size $|\mathcal{D}_{train}|$, we know that the search of the architecture and subsequent optimization of a neural network $\mathcal{M}_5$ is carried out on a data set that is likely to carry information from all relevant types of situations in which $\mathcal{M}_5$ will operate; i.e. the model will be exposed to all relevant contexts during the training sufficiently often. Thus, $\mathcal{M}_5$ was obtained by training different architectures on the same $\mathcal{D}_{train}$ and choosing the simplest architecture that was also yielding the smallest cross entropy loss given a large validation set of a fixed size.

For the sake of completeness, also a full Joint Probability Distribution (JPD) model $\mathcal{M}_6$ was used. $\mathcal{M}_6$ was defined over all variables shown in Fig. 3 and its parameters were obtained through Maximum Likelihood parameter estimation based on simple counting of state combinations.

B. Adequate Quantities of Training Data

A series of experiments confirmed that the presented QM-DGP approach can provide useful information on the quantities of training data that increase the chances of learning a good model. The experiments were carried out with three versions of ground truth model $\mathcal{M}_G^1$, $\mathcal{M}_G^2$ and $\mathcal{M}_G^3$. All versions had the topology shown in Fig. 1 and the same conditional probabilities, where $P(Position|Class, Flight)$ varied between 0.032 and 0.47, $P(Speed|, Flight, Class)$ varied between 0.048 and 0.89 while $P(RCS|Class, Position)$ varied between 0.05 and 0.88. $\mathcal{M}_G^1$ used a set of prior probabilities for the context variables: $P(Rain = true) = 0.1$, $P(Visibility = good) = 0.7$, $P(Visibility = low) = 0.2$, $P(WindForce = normal) = 0.9$, $P(Flight)$ varied between 0.1 and 0.5 while $P(WindDirection = east) = 0.5$. $\mathcal{M}_G^2$ used priors that were the same as $\mathcal{M}_G^1$, except that $P(WindForce = normal) = 0.97$. In $\mathcal{M}_G^3$ all priors over the context variables were uniform.

Exp	$\lvert\mathcal{D}_{train}\rvert$	$\lvert\mathcal{D}_{train}\rvert'$	$\lvert\mathcal{D}_{train}\rvert^{flat}$	$error^{flat}$	$error^{min}$
1	38880	36000	$\approx$ 36000	15.47%	15.33%
2	38880	72000	$\approx$ 40000	16.06%	15.98%
3	38880	42300	$\approx$ 40000	7.02%	6.92%
4	2160	6000	$\approx$ 6000	23.63%	23.54%
5	155200	172000	$\approx$ 160000	7.02%	6.92%
6	155200	540000	$\approx$ 540000	7.04%	6.92%

TABLE I
Estimated training data volumes $|\mathcal{D}_{train}|$ and $|\mathcal{D}_{train}|'$ vs. the sufficient data volumes $|\mathcal{D}_{train}|^{flat}$.

Each $\mathcal{M}_G^i$ was used to sample (i) a large test set $\mathcal{D}_{test}$ with 500000 records and (ii) a series of training data sets $\mathcal{D}_{train}$ of increasing sizes. For each set of training data $\mathcal{D}_{train}$ the experiments were repeated ten times. The averages of the error rates were plotted for different data set sizes allowing visual determination of the point $|\mathcal{D}_{train}|^{flat}$ after which no significant improvements of the error rates took place. This point corresponded to $error^{flat}$. Each ground truth model $\mathcal{M}_G^i$ was also used for optimal classification yielding the smallest possible error rate $error^{min}$, given the underlying distribution represented by $\mathcal{M}_G^i$. Moreover, as there was a significant class imbalance, the F1 score was monitored. In all experiments the F1 score improved with the overall accuracy.

In each experiment the required quantities of training data were estimated in two steps: (i) estimate initial $|\mathcal{D}_{train}|$ using equation (4) with $\kappa = 10$ and (ii) train $\mathcal{M}_{aux}$ and compute $|\mathcal{D}_{train}|'$ using (9) for the PGMs and the MLP and (6) for the full JPD model, respectively. The $\mathcal{M}_{aux}$'s architecture corresponded to the topology shown in Fig. 3 and its parameters were obtained via Maximum Likelihood parameter estimation, i.e. simple counting of state combinations in a training set of 10000 records. $ContextSummary$ in $\mathcal{M}_{aux}$ was a deterministic function of the states of the context variables. By running inference algorithm, marginal distribution over the sates of $ContextSummary$ was estimated, providing the inputs for (7). Multiple scenarios were investigated:

- **Exp 1:** Sampling on $\mathcal{M}_G^1$, $n_{core} = 108$, $n_{\mathcal{V}_1} = 3888$.
- **Exp 2:** Sampling on $\mathcal{M}_G^2$, $n_{core} = 108$, $n_{\mathcal{V}_2} = 3888$.
- **Exp 3:** Sampling on $\mathcal{M}_G^3$, $n_{core} = 108$, $n_{\mathcal{V}_3} = 3888$.
- **Exp 4:** Sampling on $\mathcal{M}_G^1$, $n_{core} = 12$, $n_{\mathcal{V}_4} = 216$.
- **Exp 5:** Sampling on $\mathcal{M}_G^3$, $n_{core} = 432$, $n_{\mathcal{V}_5} = 15520$.
- **Exp 6:** Sampling on $\mathcal{M}_G^3$, $n_{core} = 432$, $n_{\mathcal{V}_6} = 15520$

As table I shows, the initial estimates $|\mathcal{D}_{train}|$ using (4) indicated the right order of magnitude for the needed data quantity in all cases. Also the corrected $|\mathcal{D}_{train}|'$ resulting from the analysis of the auxiliary PGM was a safe estimate as $|\mathcal{D}_{train}|' \geq |\mathcal{D}_{train}|^{flat}$. Except in **Exp 2**, the classifier achieved close to optimal performance when the size of the training set was close to the estimated $|\mathcal{D}_{train}|'$. In **Exp 2** ,

$|\mathcal{D}_{train}|'$ was almost two times larger than $|\mathcal{D}_{train}|^{flat}$. This is likely a consequence of very small probability of certain state combinations, such that even in the cases of suboptimal parameter estimation these cases did not have a significant impact on the overall classification accuracy. It is also evident that the MLP model in **Exp 5** requires significantly more training data to achieve close to optimal performance than the PGM models. This is a consequence of the training method which does not fully benefit from the data structure as is the case with the EM algorithm. Still, the MLP learning seems to be able to exploit some structure in the data, as the required quantities of training data were still significantly smaller than in **Exp 6**. The hypothesis is that the MLP approach could not use the conditional independence between fragments $\mathcal{F}_1$ and $\mathcal{F}_2$, but it could exploit the d-separation between RCS_{disc} and the rest of the variables via the observations of $Class$, $Speed$ and $Position$. This is supported by the fact that for the MLP, accurate $|\mathcal{D}_{train}|'$ was obtained by using (9) on the entire core model (a single fragment). **Exp 6** clearly demonstrates the consequences of ignoring the data structure, resulting in huge quantities of training data to achieve close to optimal classification performance. In this case $|\mathcal{D}_{train}|'$ was accurately predicted by using equation (6) .

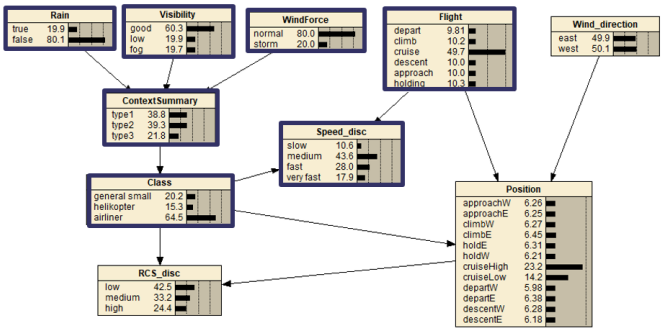

Fig. 3. Topology of the basic PGM model defined over variables $\mathcal{V}$ captured by the QM-DGP shown in Fig. 1. Variable $ContextSummary$ was added as it was known that there are three types of contexts influencing $Class$.

IV. Using QM-DGP in Life Cycles

The presented approach can play an important role at multiple stages of a system life cycle involving components obtained through machine learning. It provides support for informed decisions throughout a life-cycle. In particular, this paper assumes a URREF-driven life cycle [1] that puts the evaluation of uncertainty representations and reasoning mechanisms at the center of all stages of system development and operation. At a high level, the URREF life-cycle can be subdivided into three main phases: (i) Inception, (ii) Design, implementation and Testing, and (iii) Operation. In each phase the choices are based on different types of evaluations [1] that can be supported by the presented QM-DGP based analysis.

A. Inception

In the Inception phase of a URREF-driven life cycle the viability of AI-based solutions is investigated and decisions are made about the use of specific AI techniques and the data acquisition processes. This is the phase in which the QM-DGP graph is constructed by using qualitative knowledge of the dependencies between the relevant variables. Such knowledge is likely to be available if the data is produced by (i) processes whose physical properties are understood at a high level or (ii) by processes that are governed by clear protocols. Examples of the former are human made cyber-physical systems where direct influences between different data producing components are known. An example of the latter is the presented aviation use case, where it is known that the air crews use clear rules in different flight phases establishing direct dependencies between the flight phases, the speed, and the location. Also, we know that the RCS depends on the distance from the sensor and the aircraft type. Moreover, it is known that there are three weather situations, such that a.) only jetliners can fly, b.) small planes and helicopters are rare and c.) all three types of aircraft are likely. Even when the DGP is not known exactly, it can often be specified to good approximation through expert judgemnt. In such cases, the approximate QM-DGP can still be useful to provide a rough guide to the necessary number of training samples.

The developed QM-DGP graph supports the initial assessment of the relevant variables that should be explicitly modeled in $\mathcal{M}_k$. Based on the QM-DGP graph, we can see the relevance of variables for the task at hand and assess the complexity of correlations. This step is an example of the evaluation of **Expressivity**, **Relevance** and **Weight Of Information** criteria from the URREF ontology and provides guidance for the type and the architecture of the model $\mathcal{M}_k$. For example, if the nodes in the QM-DGP indicate different types of (discrete) observations correlated in complex ways, a Probabilistic Graphical Model (PGM) approach could be a good choice. On the other hand, if the nodes represent multi dimensional continuous measurements, such as images or other raw signals, Neural Networks may be preferred. In some cases, the $\mathcal{M}_k$ complexity can be estimated in terms of required numbers of parameters as a consequence of the QM-DGP topology and the choice of the discretization, which is straightforward if PGMs are used. Moreover, the assessment of $|\mathcal{D}_{train}|$ based on equation (4) provides useful input in the Inception phase. By knowing $|\mathcal{D}_{train}|$ and the relevant variables in the QM-DGP, the viability of the data acquisition process can be evaluated, a data collection plan prepared and the initial estimate of its costs can be made. Thus, at an early stage in the life cycle we can see whether certain parts of the system can be automated in reliable and economical ways. This is an example of an evaluation that contributes to the **Correctness** criterion from the URREF ontology.

B. Design, Implementation and Testing

The second phase of the URREF-driven life cycle consists of multiple iterations of design, implementation and testing steps. In this phase the QM-DGP provides guidance for the determination of the model architecture and the model resolution. As it explicitly encodes the qualitative dependencies

between the variables that are also represented in $\mathcal{M}_k$, it can provide guidance for systematic architectural simplifications leading to models with fewer parameters, without jeopardizing the overall classification performance. For example, given the direct dependencies in the example QM-DGP in Fig. 1, it is obvious a Naive Bayes model is likely to be inadequate. [10] provides an example where such a technique was used to obtain significantly simpler models for vessel track forecasting by introducing latent variables. While the direct dependencies often cannot be ignored, the complexity of the models can be controlled by choosing a lower resolution for discretization of variables or use of parametric models, thus reducing the number of parameters. This is an example of the evaluation of the **Expressivity** criteria according to the URREF ontology. As illustrated with the MLP experiment, in case of model types for which the architecture cannot be directly derived from the QM-DGP, the training data of sufficient size increases the chances that the search for suitable architectures will produce models that can cope with all major settings under which the input data is produced during operation.

An important step in this phase is the preparation of adequate training data sets that cover all major operational conditions. As the data is obtained, corrected $|\mathcal{D}_{train}|$' can be estimated according to equation (9) or (6), dependent on the training method. In this way the forecast of the data acquisition costs can be re-estimated and the chances of training a good model can be improved.

V. Conclusions

Life-cycles of decision support solutions involving AI technologies must include the following critical development steps:

- Choosing suitable AI techniques.
- Acquisition of sufficient amounts of training data.
- Determining the complexity of the models, such that the key knowledge can be extracted from the available data.

The presented qualitative causal models of the Data Generating Processes (QM-DGP) facilitate the above mentioned steps. The QM-DGP's topology exposes the complexity of the relations between different phenomena in a data generating process (DGP). This provides the guidance for the determination of (i) the model's complexity capable of absorbing the key information during the ML process, (ii) the volumes of training and testing/validation data required for the extraction of adequate models and (iii) the types of data the acquisition process should focus on. Based on the dependencies encoded in the QM-DGP, simple analysis rules enable coarse estimation of the data volumes enabling the training of good quality models. It should be noted that the QM-DGP based approach is agnostic to the used AI paradigm. Controlled experiments using synthetic data sampled from known ground truth models were carried out with Bayesian Networks and Neural Networks. The results suggest that the approach could indeed provide useful guidance for the preparation of the training data and generation of models covering all major contexts. For the used experimental setup, near perfect models could be trained with the data quantities estimated with the presented method. The experimental results are consistent over different parametrizations of the ground truth model. However, the ground truth model always had the same topology. Moreover, the approach can be viewed as a tool that is used throughout a system life cycle. In particular, the paper shows how the method supports critical evaluations in a URREF-driven life cycle. Such evaluations provide critical feedback for the technical choices and guidance for the planning of activities, making different phases of the life cycle more efficient and making high-quality results more likely.

The future work will focus on multiple aspects of the presented approach. Firstly, the effectiveness of the approach will be validated with more experiments based on a variety of ground truth models. While no special modelling "tricks" were used when building the ground truth model used in this paper, it might still be biased, due to unintended special structural and parametric properties. Secondly, the exploitation of assumptions allowing the use of parametric distributions as well as simplifications based on Context-Specific Independence and Independence of Causal Influence will be further investigated. In this way the estimates of the required sizes of the training data sets could get closer to the true required data set sizes.

References

[1] G. Pavlin, A. Jousselme, J. P. de Villiers, P. C. G. Costa, and P. de Oude, "Towards the rational development and evaluation of complex fusion systems: A URREF-driven approach," in *21st International Conference on Information Fusion, FUSION 2018, Cambridge, UK, July 10-13, 2018*. IEEE, 2018, pp. 679–687.

[2] C. Insaurralde and E. Blasch, "Uncertainty in avionics analytics ontology for decision-making support," *Journal for Advances in Information Fusion*, vol. 13, pp. 255–274, 12 2018.

[3] ——, "Situation awareness decision support system for air traffic management using ontological reasoning," *Journal of Aerospace Information Systems*, vol. 19, pp. 1–22, 02 2022.

[4] R. Sabatini, E. Blasch, I. Majid, A. Gardi, and A. Roy, "Application and certification challenges for ai/ml techniques in safety critical avionics systems," in *IEEE/AIAA 41st Digital Avionics Systems Conference (DASC)At: Portsmouth, VA, USA*, 09 2022.

[5] J. Pearl and D. Mackenzie, *The Book of Why*. Basic Books, 2018.

[6] J. De Villiers, G. Pavlin, A. Jousselme, S. Maskell, A. de Waal, K. Laskey, E. Blasch, and P. Costa, "Uncertainty representation and evaluation for modeling and decision-making in information fusion," *Journal for Advances in Information Fusion*, vol. 13, 2018.

[7] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[8] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller, "Context Specific Independence in Bayesian Networks," in *Uncertainty in Artificial Intelligence: Proceedings of the Twelfth Conference*. San Francisco, CA: Morgan Kaufmann, 1996, pp. 115–123.

[9] M. van Gerven, P. Lucas, and T. van der Weide, "A generic qualitative characterization of independence of causal influence," *International Journal of Approximate Reasoning*, vol. 48, no. 1, pp. 214–236, 2008.

[10] L. Jansen, G. Pavlin, A. Atamas, and F. Mignet, "Context-based vessel trajectory forecasting: A probabilistic approach combining dynamic bayesian networks with an auxiliary position determination process," in *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*, 2020, pp. 1–10.