Toward IoT-friendly Learning Models

Ernesto Damiani and Gabriele Gianini
CS Department, Università degli Studi di Milano
via Celoria 18, Milano, Italy, and
EBTIC, Khalifa University of Science and Technology
Hadbat Al Zaafran, 127788, Abu Dhabi, UAE

Michelangelo Ceci and Donato Malerba
Dipartimento di Informatica
Università degli Studi di Bari
via Orabona, 4
Bari I-70125 Italy

***Abstract*—In IoT environments, data are collected by many distinct devices, at the periphery, so that their feature-sets can be naturally endowed with a faceted structure. In this work, we argue that the IoT requires specialized ML models, able to exploit this faceted structure in the learning strategy. We demonstrate the application of this principle, by a multiple kernel learning approach, based on the exploration of the partition lattice driven by the natural partitioning of the feature set. Furthermore, we consider that the whole data management, acquisition, pre-processing and analytics pipeline results from the composition of processes pursuing different and non-perfectly aligned goals (most often, enacted by distinct agents with different constraints, requirements competencies and with non-aligned interests). We propose the adoption of an adversarial modeling paradigm across the overall pipeline. We argue that knowledge of the composite nature of the learning process, as well as of the adversarial character of the relationship among phases, can help in developing heuristics for improving the learning algorithms efficiency and accuracy. We develop our argument with reference to few exemplary use cases.**

I. Introduction

A lot of current hype on Machine Learning (ML) models within Artificial Intelligence (AI) is due to applications showcasing the impressive performance of a new generation of Deep Learners (DLs). However, experience has shown that ML analytics can achieve satisfactory performance only provided that two key conditions are met:

a. Input data are an accurate statistical representation of the physical environment, suitable for the chosen ML model.
b. Distributed training and execution of the chosen ML model can meet the deadlines given the applications latency and resource constraints.

Unfortunately, these two key conditions are seldom met by Internet-of-Things (IoT) applications. IoT data are extracted from the physical reality through a transformation process that includes data sensing and acquisition, data preparation and preprocessing. This transformation is performed as part of data gathering and preparation by sensors and other devices at the periphery, as well as by edge processors, and is rather far from an ideal statistical measurement process (e.g. the classic one, mapping a point value into a normally distributed measurement). Also, input data latency, availability, and veracity, as well as the corresponding computational load, may widely vary, depending on the conditions in the field.

A. Context and problem

IoT applications address multi-layered scenarios (Fig. 1) where the input consists of highly dimensional data points coming from multiple sources and/or characterized by different feature subsets. For example, a person can be identified by face, finger-print, EEG brain-waves, and irises, each coming from a different sensor, while the surface of a physical object can be represented by its color and texture attributes, which correspond to two perceptually separate subsets of features. These scenarios involve huge and highly dimensional data flows, where each data item has hundreds, thousands or even millions of dimensions. Creating and managing such flows may be straightforward (e.g. when real estate is monitored via a single multi-spectral camera on a satellite or UAV), or, on the contrary may require the computation of semantic-driven joins (e.g. when a situation is monitored by a sand-dust of heterogeneously distributed sensors not all of which are operational at any given time).

We argue that IoT environments require specialized ML models, for two major reasons.

- Firstly, a feature-set, collected by many different sensors, or mobile devices, at the periphery, will have natively a *faceted structure* that can be exploited in the learning strategy. A set of ML analytics techniques, called *multi-view learning*, treat input data facets (called *views*) differently, e.g. using multiple kernels, when learning classification models, or coordinating training of multiple models (co-training). Notably, *multiple kernel learning* algorithms exploit those kernels that naturally correspond to different views and combine them linearly or non-linearly to improve learning performance; *co-training* algorithms pursue agreement between models trained on distinct views, and *subspace learning* algorithms try to identify a latent subspace shared by multiple views by assuming that the input views are generated from it.
- Secondly, IoT ecosystems are owned and managed by multiple operators, each with its own interests and agenda; therefore, they cannot rely on full mutual trust between the pipeline modules and stages, which is usually assumed in learning. We envision model composition to incorporate *adversarial learning*, which deals with high-dimensional data where features may have diverse veracity, due to the presence of hostile, untrusted or

Fig. 1. Analytics computation in the IoT setting.

semi-trusted components along the model training chain. The adversarial paradigm considers the data gathering/preparation as inherently including a source of perturbation/noise/uncertainty and train ML models considering the uncertainty type and the corresponding uncertainty principles.

Today, most standard ML algorithms made available through libraries are designed to be effective and accurate under ideal conditions. Even when data transformation issues are taken into account, the algorithms are devised to be robust with respect to inaccuracy, but not necessarily consider tougher challenges, due to data gathering and preparation choices. Often, the choice of an AI algorithm for analytics makes the implicit assumption that the input data are still representative samples of the actual phenomenon to be studied. When it is known that the data do not fulfill this condition, in principle they can be simply discarded and further data acquisition rounds can be envisaged; in practice, often such data are used anyway, and the final user of the analytics the human decision-maker - is not informed that the analytics outcomes cannot be fully trusted and, even if so, he does not understand why this is the case.

B. Proposed approach

We envision an approach to integrated design and deployment of data preparation/preprocessing and data analytics providing a clear understanding of the entire data pipeline to ground the human decision-maker level of trust in the outcome, improving human ability to compare and select the ML model results.

Such an approach will model the IoT data analytics/machine learning pipeline (acquisition, preparation, preprocessing, and analytics) as a *composition of services* [1] pursuing different and non-aligned goals. Our abstract representation needs to be translated into deployable computations at the device, edge, and core of the IoT. Also, it must be adaptively tunable to address bias and loss of precision due to missing data substitutions, alignment of data from different dimensions, interpolation/extrapolation, and introduction or artificial autocorrelation in time series, and to assure the desired non-functional properties of the computation in terms of privacy, data integrity and data protection.

We propose to base our integrated design process on two pillars: *structural awareness*, making the learning process reminiscent of the multiple sources contributing to multi-dimensional data, and *adversarial composition*, modeling the data preparation/gathering as a source of perturbation/noise/uncertainty; this paradigm would take as parameters the pertinent *uncertainty models* and the related uncertainty principles. One can also consider and investigate ethics and legal concerns as modular sources of perturbation, which may depend on the cultural and regulatory environment where the pipeline is deployed.

This approach will provide to the human designer full visibility and control over distributed preparation of input data, as well as training and execution of the ML models, in order to: (i) achieve certifiable performance, security and quality of the analytics (ii) provide a clear foundation for a chain of trust in the ML-based analytics outcome (iii) provide a lever to enforce ethical and legal constraints (e.g. fairness or privacy-related) within the pipeline, without compromising analytics quality. Within an adversarial paradigm, we can model the overall data analytics/machine learning pipeline (acquisition, preparation, preprocessing, analytics) as the composition of processes pursuing different and non-perfectly aligned goals (most often, enacted by distinct agents with different constraints, requirements competencies and with non-aligned interests).

Thanks to an integrated design process, one can choose the suitable strategies in the different phases. For instance:

- if the interests of preprocessing and analytics are aligned, one can resort to optimization,
- if they are partially unaligned, one can resort to multi-objective optimization
- if the agents are also different and with different objectives, one can resort to game theory

The specification of the uncertainty model involved helps in a principled choice of the algorithms suitable for preprocessing and analytics. The uncertainty model is especially useful to model the relationships among the quantities of interest (e.g. related by uncertainty principles) and to identify the tradeoffs (as in the case of bias-variance-consistency etc. for the statistical estimators).

Hereafter, after pointing to the related works (Section 2), we illustrate the notion of partition-driven multi-view learning (Section 3). Then, we develop the concept of the use of the adversarial paradigm (with its Game Theoretical implications) across the whole data acquisition, management, preprocessing and analytics pipeline (Section 4). A brief discussion (Section 5) concludes the paper.

II. Related work

A. Kernel methods

In multidimensional data classification, an *optimal* linear separator is the hyper-plane that has the largest margin between positive examples on one side and negative examples

on the other. Finding it is a well-known quadratic optimization problem; however, this problem may have no solution in the original data space. *Kernel methods* try to map data to a higher-dimensional space where they will be linearly separable. To do so, one has to identify a suitable transformation, creating new dimensions out of the original ones. The invertible functions that are used to operate these transformations are called kernels. Kernels are built combining input features by using basic operations such as the multiplication or exponentiation and their linear combinations [2].

Among the most used kernels are polynomial kernels and radial basis function (RBF) kernels, that provide parametric templates whose parameters can be found by optimization. Choosing a kernel is itself a problem; one can explore the combinatorial space of dimensions and assess results by cross-validation. For highly multi-dimensional data, however, the exhaustive exploration of the combinatorial space is unfeasible.

Much research has been devoted to building complex kernels as combinations of simpler ones: different kernels may correspond to using different notions of similarity or – more interestingly for our current purposes – may be using groups of features that come from related sources or are otherwise semantically related. The kernel function can be a linear or a nonlinear function, it can combine already optimized kernels or perform the optimization of their parameters during the combination. There is a significant amount of work in the literature for combining multiple kernels: a review is provided in [3]. Our proposal, in the next section, relates to a particular technique for the exploration of the partition lattice which represents an example of IoT friendly faceted learning.

B. Adversarial models

Adversarial models based on Game Theoretical principles have recently gained attention thanks to their use in the final phases of the data processing pipeline [4], [5], specifically in the machine learning phase.

Huang et al. [4] define adversarial machine learning as the study of effective machine learning techniques against an adversarial opponent; the opponent might, for instance, affect the veracity of the training samples.

Goodfellow et al. [5] develop this idea into a training technique that can be applied when the analytics involves learning generative models. Generative models are forms of representations that capture the data distribution from the available training data and can then be used to generate new samples, typically with the purpose to make them available to the training process. Goodfellow et al. propose a change of perspective: they provide a framework to estimate generative models by training in parallel two models: the generative one (G) and a discriminative one (D) that estimates the probability that a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake. They frame this concept into the game theoretical model of zero-sum games, which are two-player games where the interest of the players are perfectly opposite to one another (same direction, opposite sign), so that the gain of one player in a realization of the game is equal to the loss of the other.

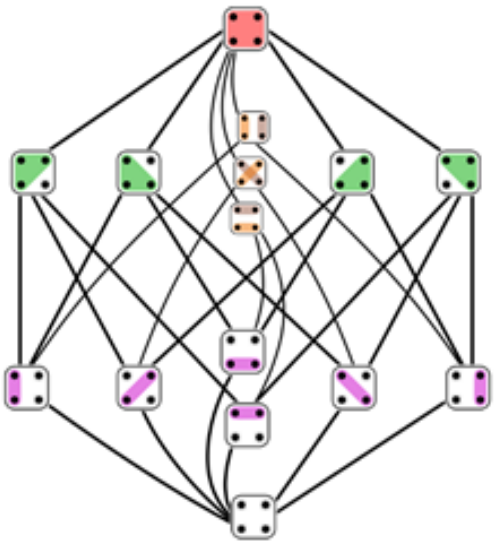

Fig. 2. Lattice of Partitions of a 4-Element Set (courtesy of Tilman Piesk).

Our point here is different. We propose to consider the adversarial component all along the data acquisition management and processing pipeline. In our view, the pipeline is operated by actors/players with non-aligned interests. They need not be opponents to one another: the players in the pipeline are typically driven by compatible objectives, however, the optimization of one players objective prevents the optimization of the other players.

III. Multiple kernel methods based on the partition lattice

As mentioned in the previous section, multiple-kernel methods create each kernel by aggregating (e.g. by multiplication) the elements in a subset of the data features. Intuitively each choice of multiple kernel configuration corresponds to picking a partition of the full set of features and subsequently multiplying together all the elements lying in the same partition block. Our idea relies on exploring the partition lattice by using a technique originally introduced for navigating service descriptions [6], i.e. selectively "smushing" block boundaries by applying lattice operation to obtain new partitions [7]. An analogous exploration with a different approach was used in [8] for the task of hierarchical clustering, based on the navigation of the Boolean lattice.

To outline our "*smushing*" technique, let us briefly recall some background notions. A partition π of a set S is a way of writing S as a disjoint union of nonempty subsets called blocks. The above picture shows the fifteen partitions of a four-element set, ordered by refinement. There is a natural one-to-one correspondence between the partitions of a set S and the equivalence relations on it; each class of the equivalence

relation provides a block of the corresponding partition. A partition π is finer than a partition π' *iff* every block of π' results from the union of blocks of π. This puts a partial ordering on the set $\Pi(S)$ of all partitions of S: we say $\pi \leq \pi'$ (or $\pi \geq \pi'$) if π is *finer* (*coarser*) than π'. This ordering makes $\Pi(S)$ into a complete lattice (unlike the Boolean lattice $\mathcal{B}_n = 2^S$ of subsets of S, $\Pi_n = \Pi_{|S|} = \Pi(S)$ is not distributive). Notice that partitions $\pi \in \Pi_n$ of rank i consist in $(n-i)$ blocks.

Here, we elaborate on the classic result that equivalence relations on a set of multidimensional data points can be induced to obtain approximation spaces over it (see Pawlak [9]). Specifically, let the data-set S contain N data n-tuples. Let us denote by t_j^k the value of the k-th feature ($k \in [n]$) of the j-th n-tuple ($j \in [1, N]$). We can build an equivalence relation on the set of N tuples based on the coincidence of the values of a chosen feature, the k-th feature: any relation $\sim_k$: $S \to S$ such that $t_i \sim_k t_j$ *iff* $t_i^k = t_j^k$, is an equivalence relation on S.

In the classic Pawlak construct, each subset T of S can be expressed, based on the partition $\sim_k$, using a pair, composed of a *lower approximation* $\underline{T}_{\sim_k}$ (the largest class of $\sim_k$ contained in T) and of an *upper approximation* $\overline{T}_{\sim_k}$ (the smallest class containing T of the coarser partitions $\sim'_k \geq \sim_k$). Clearly $\underline{T}_{\sim_k} \subseteq \overline{T}_{\sim_k}$. Pawlak's rough set constructs $\underline{T}_{\sim_k}$ and $\overline{T}_{\sim_k}$ identify an approximate cover of T with the information granules of $\sim_k$. The index k (which selects a single feature) is often substituted by a feature subset K, computed by minimizing an Entropy function or the difference between the upper and lower approximations of benchmark subsets.

Let us consider a simple example in four dimensions:

Device ID	Battery Level	OS	Available
1	AVERAGE	Android	N
2	HIGH	Android	Y
3	AVERAGE	iOS	Y
4	LOW	Symbian	N

Here, there are $N = 4$ instances, or n-tuples, each describing a phone. If $K = \{OS\}$, we get the equivalence relation $\sim_K \equiv \{\{1, 2\}, \{3\}, \{4\}\}$. Consider the concept set T of "available phones" (instances such that $Available = Y$). Under this relation, the lower approximation is $\underline{T}_{\sim_K} = \{3\}$, and the upper approximation is $\overline{T}_{\sim_K} = \{\{1, 2\}, \{3\}\}$. The approximation accuracy (defined by the ratio between the lower approximation and upper approximation cardinalities) is 0.5.

Our idea is to select K dynamically, based on the approximation accuracy on benchmark concepts (as opposed to statically, based of semantic distance between features). We generate a starting partition of S in two blocks $(K, S-K)$ to be exploited for two-kernel computations. Then, we explore the partition lattice using refinements of block $S-K$ of the starting partition (exploring the lattice lower cone). Intuitively, we are looking for an optimal partition, in the sense that adding an additional kernel will not improve the performance of the system.

Should the exploration be exhaustive, its complexity would be given by the sum of the level numbers – known as Stirling numbers of the second kind (sum of Stirling numbers of the second kind are known as Bell numbers) of the partition lattice cone rooted in $(K, S-K)$ [10]. We, on the contrary, are looking at an exploration strategy based on *chain decompositions* [11], which would be linear in the cardinality of $S-K$.

Let a partially ordered set $(\mathcal{P}, <)$ be given (we refer specifically to the Boolean lattice $\mathcal{B}_n$ with the inclusion relation and to $\Pi(S)$ with the refinement relation). A *chain* $C = (x_1, x_2, \ldots, x_c)$ in $\mathcal{P}$ is a sequence $x_1 < x_2 < \ldots < x_k$ where each $x_i \in \mathcal{P}$. For $x, y \in \mathcal{P}$, we say that y *covers* x if $x < y$ and there does not exist $z \in \mathcal{P}$ such that $x < z$ and $z < y$. A *saturated chain* (or *skipless chain*) C in $\mathcal{P}$, is a chain where each element is covered by the next. $\mathcal{P}$ is *ranked* if there exist a function $r : \mathcal{P} \to \mathbb{Z}_{\geq 0}$ such that x *covers* y implies $r(y) = r(x) + 1$. Suppose $\min\{r(x) | x \in \mathcal{P}\} = 0$, the *rank* of $\mathcal{P}$ is denoted $r(\mathcal{P}) = max\{r(x) | x \in \mathcal{P}\}$. A saturated chain $x_1 < \ldots < x_n$ in a ranked poset $\mathcal{P}$ is said to be a *symmetric chain* if $r(x_1) + r(x_n) = r(\mathcal{P})$. A poset $\mathcal{P}$ has a *symmetric chain decomposition* if it can be written as a disjoint union of saturated symmetric chains: $\mathcal{P} = C_1 \cup C_2 \cup \cdots \cup C_k$.

The Boolean lattice $\mathcal{B}_n$ admits symmetric chain decompositions, most notably the Bruijn's decomposition [12]. On the contrary, the partition lattice is not symmetric. For example, there are $2^{n-1} - 1$ partitions of an n-set into two blocks, but only $n(n-1)/2$ partitions of an n-set into $n-1$ blocks. Thus, there is no complete decomposition of the lattice into symmetric chains (for $n \geq 3$). However, using de Bruijn's decomposition together with a certain method to encode sets, Loeb, Damiani and D'Antona [11] find a maximal collection of disjoint symmetric chains of partitions. Specifically, using de Bruijn's decomposition of $\mathcal{B}_n$ they show how to generate a partial decomposition of the lattice Π_{n+1} of the set $\{1, 2, \ldots, n+1\}$: they find a collection of disjoint symmetric chains which includes all partitions of rank $\leq \lfloor (n-1)/2 \rfloor$. Such a collection is clearly maximal. We refer to [11] for the details of the complex mapping procedure based on a specific encoding $c(S)$ of the elements of the chains in $\mathcal{B}_n$.

We provide an illustrative example adapted from [11]. The de Bruijn decomposition of $\mathcal{B}_3$ consists of the 3 chains $C_1 = (\emptyset, \{1\}, \{1, 2\}, \{1, 2, 3\})$, $C_2 = (\{2\}, \{2, 3\})$ and $C_3 = (\{3\}, \{1, 3\})$. Encoding the sets above yields the following partition types (each digit in each sequence represent the cardinality of a block) $(1111, 112, 13, 4)$, $(121, 31)$ and $(211, 22)$. Then one computes the partitions of each type. The resulting decomposition is shown in Table I. The column marked Π_4 lists the three chains in the decomposition of Π_4.

Using the chain decomposition of $\Pi(S)$, we can perform the search for the optimal kernel partition starting from a two block partition $(K, S-K)$ in polynomial time.

TABLE I
EXAMPLE OF CHAIN DECOMPOSITION OF Π_4

$S \in \mathcal{B}_3$	$c(S)$	Π_4		
$\emptyset$	$1111 \to 1111$	$1/2/3/4$		
$\{1\}$	$0211 \to 112$	$1/2/34$		
$\{1,2\}$	$0031 \to 13$	$1/234$		
$\{1,2,3\}$	$0004 \to 4$	1234		
$\{2\}$	$1021 \to 121$	$1/23/4,$	$1/24/3$	
$\{2,3\}$	$1003 \to 31$	$123/4,$	$124/3,$	$134/2$
$\{3\}$	$1102 \to 211$	$12/3/4,$	$13/2/4,$	$14/2/3$
$\{1,3\}$	$0202 \to 22$	$12/34,$	$13/24,$	$14/23$

IV. ADVERSARIAL MODELS ALL THE WAY ACROSS THE DATA PIPELINE

The adversarial component is present all along the data acquisition and processing pipeline because the the pipeline is operated by actors/players with non-aligned interests. Although they are typically driven by compatible objectives, the optimization of one player's objective prevents the optimization of the other player's.

As a concrete example, we consider the contrast between the goals of a data preprocessing phase with those of a ML data analytics phase (the contrast between the players enacting the two phases).

- The typical goal of the *data preprocessing phase* consists in improving the quality of the data coming from the data acquisition phase and yielding a final dataset which can be considered in some sense "correct" for further data processing algorithms. We can call these the *reconstructed data*.
  - The data coming from the data acquisition phase can be affected by undesired factors such as noise, missing values, redundant or inconsistent data; sometimes also the huge size of the data represent an undesired factor for the subsequent knowledge extraction phases. The latter kind of concerns is addressed in the sub-phase of data reduction, the former in the sub-phase of data preparation.
  - Data preparation includes tasks such as data normalization, missing value imputation, noise identification, data cleaning, data transformation and data integration. Data reduction includes tasks such as instance-selection, feature-selection, and discretization.
  - Among the preprocessing operations that are most critical to the subsequent analytics are missing value imputation and data integration. A prototypical example of data integration consists in the creation of d-dimensional records out of d single-feature records: the data of each column could have been gathered by different sensors on a homogeneous field, measuring different quantities (temperature, humidity, wind speed) annotated with their time-stamps. Let us assume the measurements of the different sensors are not synchronized. The passage from d 1-dimensional views of the reality to a single d-dimensional view can be obtained by first merging the time-stamps into an ordered list: the data available at each time-stamp will naturally compose a multi-dimensional record typically *plagued by missing feature-values*.
- In passing from the raw data coming from the acquisition phase to the reconstructed data, some manipulations are performed (see below) and a considerable amount of information might be lost. In practice, one can keep track of the uncertainty associated to the reconstructed data only to some point, because of the cost and the operational difficulties of such a task.
- For the sake of illustration, we consider a *ML analytics phase* whose goal is to learn an accurate predictive model. A predictive model is useful, in practice, if it provides also information on the veracity of its predictions because the lack of veracity has a cost. On the one hand, to make available an uncertainty model of the predictions one needs to use in input an uncertainty model associated to the input data. Due to the preprocessing manipulations, this uncertainty model might be not available. On the other hand, devising and using ML algorithms able to act on the raw data would have a high cost.

A. Single player setting

Should the overall data processing be controlled by a single player P, he/she could choose the optimal strategy of data preprocessing suitable for the ML analytics phase. For instance, given a dataset – issued by the data acquisition phase – plagued by missing values of some features, and given the task of learning a decision tree out of the data, player P can decide whether

- to resort to the imputation of convenient substitutes for the missing data and accept the consequent inaccuracies in the prediction
- or to avoid missing data imputation altogether and the learn as many different models as the combination of available features.

This single player should be able to strike a balance between the inaccuracy of the predictor and the cost of learning many models.

This can be done by adopting an optimization approach, typically a multipurpose optimization approach.

B. Many players setting

Most often there are several players and the interest of the players enacting the preprocessing phase are not aligned with the ones of the following phases.

Data preprocessing is a costly operation and seldom is performed with the purpose of feeding a single analytics

algorithm. Most of the time, the preprocessing tries to produce a dataset that is later usable by different algorithms: sometimes the restricted set of analytics algorithm that is going to be used later is known in advance, some other times, on the contrary, the preprocessing phase might even avoid altogether considering specific analytics and point to a general purpose data-set (completely agnostic w.r.t. the applicable analytics).

In the multi-player setting all the players share some parts of one's another goals (all aim at the successful development of the overall data acquisition, preprocessing and analytics process), however the best choice of a player (e.g. in terms of costs) does not necessarily yield the best results to the other. In this contrast consists the "adversary" character of the process.

Although the adversarial character is very common, one has to distinguish different settings based on the information available to the players regarding the other (preceding or subsequent) phases. Those scenarios can be modeled within Game Theory. A possible Game Theoretic frame form modeling the process is the one of sequential games of imperfect information, where a player needs to take decisions only based a partial knowledge of the other players decisions/strategies.

V. Conclusion

In this work, we argued that IoT environments require specialized ML models since the feature-sets of the data, collected by many different devices, at the periphery, will be naturally endowed with a faceted structure, that can be exploited in the learning strategy. We showed an example of such an approach, developed around the exploration of the partition lattice to build multiple kernel configurations.

We also pointed to the need of adopting adversarial models of the IoT data acquisition, preprocessing and analytics pipeline. This is motivated by the fact that IoT ecosystems are managed by multiple operators. Although those interests do not point in completely opposite directions, they are not aligned. As a consequence, one cannot rely on full mutual trust among all the pipeline modules. An adversarial paradigm, based on Game Theory can help in soundly modeling the overall process.

References

[1] E. Damiani, C. Ardagna, P. Ceravolo, and N. Scarabottolo, "Toward model-based big data-as-a-service: The toreador approach," in *Advances in Databases and Information Systems*. Springer, 2017, pp. 3–9.

[2] N. M. Nasrabadi, "Pattern recognition and machine learning," *Journal of electronic imaging*, vol. 16, no. 4, p. 049901, 2007.

[3] M. Gönen and E. Alpaydın, "Multiple kernel learning algorithms," *Journal of machine learning research*, vol. 12, no. Jul, pp. 2211–2268, 2011.

[4] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. D. Tygar, "Adversarial machine learning," in *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*, ser. AISec '11. New York, NY, USA: ACM, 2011, pp. 43–58. [Online]. Available: http://doi.acm.org/10.1145/2046684.2046692

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 2672–2680. [Online]. Available: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[6] P. Bosc, E. Damiani, and M. Fugini, "Fuzzy service selection in a distributed object-oriented environment," *IEEE Transactions on Fuzzy Systems*, vol. 9, no. 5, pp. 682–698, 2001.

[7] E. Damiani, B. Olibini, and L. Tanca, "Fuzzy techniques for xml data smushing," in *International Conference on Computational Intelligence*. Springer, 2001, pp. 637–652.

[8] Z. Markov, "A lattice-based approach to hierarchical clustering." in *FLAIRS Conference*, 2001, pp. 389–393.

[9] Z. Pawlak, *Rough sets: Theoretical aspects of reasoning about data*. Springer Science & Business Media, 2012, vol. 9.

[10] E. Damiani, O. D'Antona, and F. Regonati, "Whitney numbers of some geometric lattices," *Journal of Combinatorial Theory, Series A*, vol. 65, no. 1, pp. 11–25, 1994.

[11] D. Loeb, E. Damiani, and O. D'Antona, "Decompositions of bn and πn using symmetric chains," *Journal of Combinatorial Theory, Series A*, vol. 65, no. 1, pp. 151–157, 1994.

[12] N. De Bruijn, C. van Ebbenhorst Tengbergen, and D. Kruyswijk, "On the set of divisors of a number," *Nieuw Arch. Wiskunde (2)*, vol. 23, pp. 191–193, 1951.