Expert Independent Bayesian Data Fusion and Decision Making Model for Multi-Sensor Systems Smart Control

Mumbere Muyisa Forrest[1], Masivi Osée Muhindo[2], Kasereka Masivi Samuel[3]

[1]School of Software, Central South University, Changsha 410000, China.
[2]Département d'Informatique de Gestion, ISP/Muhangi, Butembo, DRC.
[3]Departement of Computer Science, Invertis University, India.

---***---

Abstract- *Internet of Things (IoT) and its applications have increased the number of multi-sensors computer applications. Then, the necessity of multi-sensor data merging and expert independent decision algorithms is real. This paper proposes a novel multi-sensor system smart control model based on Bayesian. Proposed "study on intelligent" algorithms are expect-dependent trainable predicting the system only from anterior and actual data. Simulations test on a three sensors system (sol temperature, air temperature, and moisture) an overall prediction precision of more than 96%. However, a real life customizable implementation of the proposed algorithm is needed.*

Keywords: Naïve Bayes, Data fusion, Multi-sensor, System smart control, Expert independent training.

1. INTRODUCTION

The meteoric rise of Internet of Things (IoT) has increased the number of sensors in almost all computer applications and hence increases the necessity of multi-sensor systems smart control algorithms. These algorithms are not only required to merge sensed data from a good number of sensors into a common representational format, but also to make relevant decisions. Despite the fact that suggested principles, procedures, theories and tools are approximately the same [1], decision algorithms depend on the number of sensors [2] [3] in the application context [4]. Furthermore, sensors characteristics increase exponentially the complexity of the system decision algorithm.

2. RELATED WORKS AND OBJECTIVES

Data fusion and decision making and related challenges [4] have been addressed by researchers for decades. Since then, two major approaches have emerged: artificial intelligence methods and probabilistic based methods. Artificial approaches, with the main focus on machine learning, fuzzy logic, have been reputed to yield higher accuracy compared to other techniques [2][5]. In this line, [6] proposed a generic data fusion system which established a relationship between the source of data and the type of processing in order to extract maximum possible information from data collected. The system was able to stand between the source data and the human and helped him to make decisions based on the fused output. The big challenge with these methods is the amount of data and processing need for training the decision making algorithm.

Therefore, Bayesian approach, including Bayesian analysis, Statistic, and recursive operators stood as one of reliable alternatives. Used already for data fusion [7] Bayesian approach is being accepted to be one of the most classical fusion approach. Furthermore, the authors in [8] demonstrated that the data fusion based on Bayes estimation can weaken the possible sensor errors, resulting from the sensor faults and noise inference. The most appealing advantage of Bayes parameters estimation algorithms is the small amount of training needed for classification [9] [10] and its impendence toward system experts.

This paper introduces the Naïve Bayes theorem in the decision algorithm from merged data collected from sensors with different characteristics. The algorithm should to make decision in control systems under multi-sensor context. The proposed method includes system parameters learning and time-based system state prediction and is expected [11] (a) to easy design process with less free parameters to set, (b) to easy result application to a large variety of tasks, (c) to use a small amount of data for learning process (d) to be computationally fast when making decisions.

3. PROPOSED MATHEMATICAL METHOD

In this paper, Bayesian inference is used to draw conclusions about features (parameters) in system control based on a sample from the same system. Both parameters and sample data are treated as random quantities [12]. The proposed algorithm computes the distribution for the system parameters from the likelihood function which defines the random process that generates the data, and a prior probability distribution for the parameters.

By assuming that all the variables are observed with no missing data and that all classes have equal prior probabilities [13], the proposed method estimates the probability of a system feature by the frequency of occurrence of the same feature $x^{(l)}$ in the feature database $x^{(l)} = \{x_{k1}^{(l)} \dots x_{kn}^{(l)}\}$.

The conditional likelihood of this feature is computed by (1) and the density probability of the set of data D is computed by (2). Since $\theta_{i,j,k}$ occurs whenever $X_i = x_k$ and $P(X_i) = x_j$, (3) is deducted from (2), then the logarithmic notation of (3) gives (4).

$$\begin{aligned} P(X = x^l|\theta) &= P(X_1 = x_{k_1}^{(l)}, \dots, X_n = x_{k_n}^{(l)}|\theta) \\ &= \prod_{i=1}^{n} P(X_i = x_{k_1}^{(l)}|pa(X_i) = x_j^{(l)} \\ &= \prod_{i=1}^{n} \theta_i, j(l), k(l) \end{aligned} \quad (1)$$

$$L(D|\theta) = \prod_{i=1}^{n} P(X = x^l|\theta) = \prod_{i=1}^{n}\prod_{l=1}^{N} \theta_i, j(l), k(l) \quad (2)$$

$$L(D|\theta) = \prod_{i=1}^{n}\prod_{l=1}^{N} \theta_i, j(l), k(l) = \prod_{i=1}^{n}\prod_{j=1}^{qi}\prod_{k=1}^{ri} \theta_{i,j,k}^{N_{i,j,k}} \quad (3)$$

$$LL(D|\theta) = \log L(D|\theta) = \sum_{i=1}^{n}\sum_{j=1}^{qi}\sum_{k=1}^{ri} \theta_{i,j,k}^{N_{i,j,k}} \quad (4)$$

Since [6] demonstrates that (4) formula can be simplified to (5), the Maximum Likelihood (ML) approach used in this paper is expressed by (6).

$$\hat{\theta}_{i,j,k} = \frac{N_{i,j,k}}{\sum_{k=1}^{r_i} N_{i,j,k}} \qquad \forall k \in \{1, \dots, r_i\} \quad (5)$$

$$\hat{P}(X_i = x_k|pa(X_i) = x_j) = \hat{\theta}_{i,j,k}^{MV} = \frac{N_{i,j,k}}{\sum_k N_{i,j,k}} \quad (6)$$

With $N_{i,j,k}$ being the number of features in the database whose variable X_i is in the state x_k and parents are in the configuration x_j.

This classifier was improved by Naive Bayes to handle an arbitrary number of independent variables by constructing the posterior probability for the feature C_j given a set of variables, $X = \{X_1, X_2, \dots. X_d\}$ among a set of possible outcomes $C = \{C_1, C_2, \dots. C_d\}$ by (7). Assuming that the variables are statistically independent, the likelihood is decomposed to a product of terms in (8). Then from (8) the estimation computed by (9).

$$p(C_j|X_1, X_2, \dots. X_d) \,\alpha\, p(X_1, X_2, \dots. X_d|C_j)p(C_j) \quad (7)$$

$$p(X|C_j)\alpha \prod_{k-1}^{d} p(x_k|C_j) \quad (8)$$

$$p(C_j|X)\alpha p(C_j) \prod_{k-1}^{d} p(x_k|C_j) \quad (9)$$

Where:

- $p(C_j|X_1, X_2, \dots. X_d)$: Posterior probability of class membership.
- $p(X_1, X_2, \dots. X_d|C_j)$: Likelihood which is the probability of the predicator given class.
- $p(C_j)$: class prior probability

Using (9), any new case X can be labeled by a class level C_j with the highest posterior probability. Then after normalization, the decision distribution is :

$$\frac{1}{\sigma_{xj}\sqrt{2\pi}} exp\left(\frac{-(x - \mu_{xj})^2}{2\sigma_{xj}}\right) \quad (10)$$

Where :

$-\infty < x <, -\infty < \mu_{xj} <, \sigma_{xj} > 0$

$\mu_{xj}: mean, \sigma_{xj}: standard\ deviation$

4. PROPOSED COMPUTATIONAL MODEL

The proposed computational protocol sketched in Figure 1, uses the following steps: (a) getting data from sensors (b) split training set from test data (c) learning/training the model (d) predict class using Naive Bayes model.

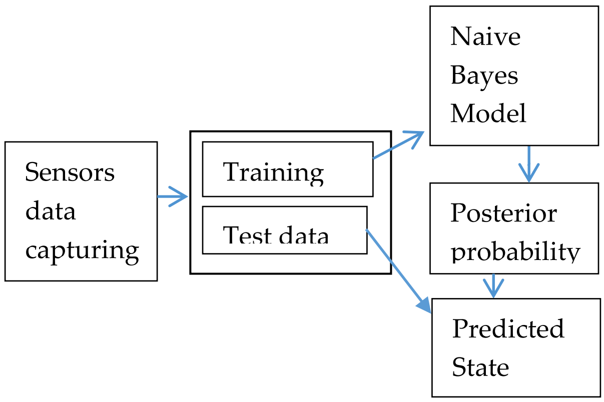


Figure 1: Naïve Bayes protocol for state estimation

A. Collecting data from sensors

During this step, the system collects data from a given number of sensors. Each sensor can be in one of the three states: low, adequate or high. Each sensor is assumed to be independent in terms of data types and collecting rate [14] [15]. The example provided in Table 1 gives the ranges for the training system of three sensors.

Table 1. Adequate range values for the system C.

Sensor	S1	S2	S3
Values	$16 \le S_1 \le 20$	$20 \le S_2 \le 25$	$80 \le S_3 \le 120$

B. Learning & training the model

The proposed model is “study on intelligent” say it was trained from data collected from sensors [15]. Hence, the state of the system depends on values provided by different sensors: at time t, Ct depends on the state of S_1t, S_2t, S_3t, S_nt. From example (Table 1), the System State had 27 different probable classes. Each class is composed by 3 different states provide by the 3 sensors. The class C1can be represented by: S1low, S2adequate, S3high, meaning that at a given time t, sensor 1 sent a value lower than 16, sensor 2 sent a value between 20 and 25 and sensor 3 sent a value higher than 120. The learning process used it this paper can be summarized by algorithm 1 as follow:

Algorithm 1: Training process
Input: K (Training set)
  For each class c in K **do**
    Compute mean(k) and std(k)
  End for
Output: mean, std

During the training stage, the mean and standard deviation values of the features for each class were computed.

C. Classification

Assuming that system parameters are distributed according to the Gaussian density, two parameters were computed. First the parent node without parentsC_j. It probability $p(C_j)$ is the frequency of the C_j class in the training database. Since all the classes have the same probability at this stage:

$$P(C_1) = P(C_2) = \cdots = P(C_{91}) = \frac{N(C_j)}{N} = \frac{N(C_j)}{\sum_1^{91} N(C_j)} \quad (11)$$

$$\sum_1^{91} P(C_j) = 1$$

Where:
N : The size of the database
$N(C_j)$: The number of observation belonging to the class C_j

Second, the probability of the children nodes were computed by using the Normal law in conditional probability of node S_i considering the parent C_j (12) then combine the different values computed in the previous step (13). $P(S_1, S_2, S_3)$ in (13) being constant hence easy to compute, (13) was rewritten as (14).

$$P(S_i|C_j) = N(S_i, \mu_{i,j}, var_{i,j}) \quad (12)$$

$$P(C_j|S_1, S_2, S_3) = \frac{P(C_j) * P(S_1, S_2, S_3|C_j)}{P(S_1, S_2, S_3)} \quad (13)$$

$$P(C_j|S_1, S_2, S_3) = \frac{P(C_j) * P(S_1|C_j) * P(S_2|C_j) * P(S_3|C_j)}{P(S_1, S_2, S_3)} \quad (14)$$

D. Decision

The proposed Bayesian classifier (Algorithm 2) is a probabilistic model based on the Bayes rule [17] in which each element $S = (S_1, S_2, S_3)$ is associated to a C_j class with a maximum a posteriori.

$$C_{MAP} = argmax_{C_j} P(C_j|S_1, S_2, S_3) \; \alpha \; argmax_{C_j} P(C_j) * P(S_1|C_j) * P(S_2|C_j) * P(S_3|C_j) \quad (15)$$

Algorithm 2: Testing
Input: k(Learned features)
  For each class li **do**
    Calculate $P(l_i|k)$
  End for
Output:$L \leftarrow armax_{\forall li \in k}\{P(l_i|k\}$(estimated class)

The probabilities for the objective existing in each class are computed and the highest is chosen as the estimated class.

5. SIMULATION RESULTS

The proposed computational model was evaluated and its goodness was tested using the following evaluation parameters: accuracy, precision, recall, f1 score drawn from a confusion matrix. Furthermore, imbalanced data set results were compared to balanced data set results

A. Confusion matrix

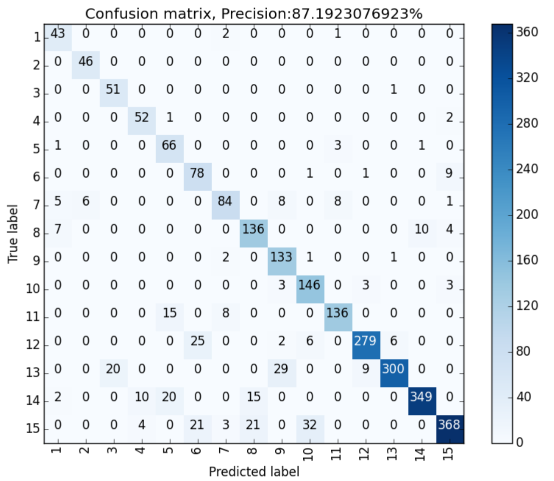


Figure 2. Confusion matrix for imbalanced data set

Figure 2, shows a confusion matrix, with an accuracy rate of 87% for imbalanced data. In Figure 3, the false negative and the false positive have been minimized by balancing the data in the training set; the result shows that the accuracy of the model has improved from 87.33% to 96.33%.

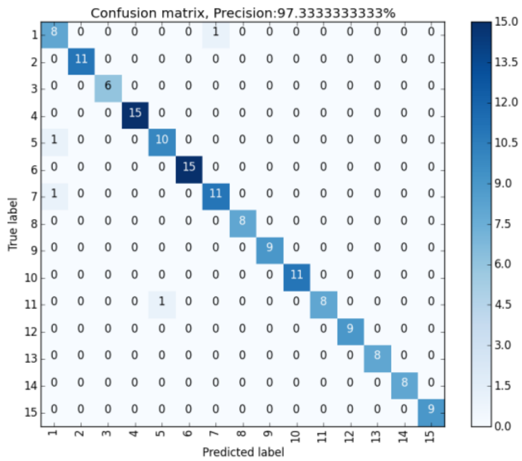
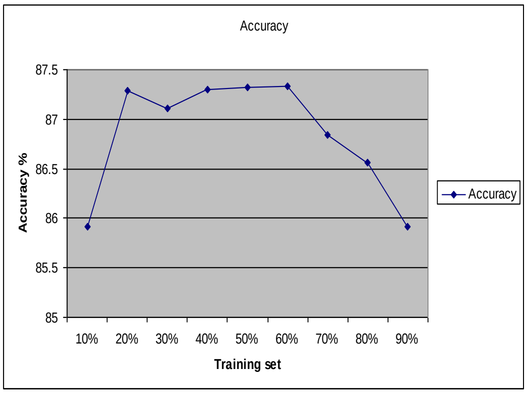


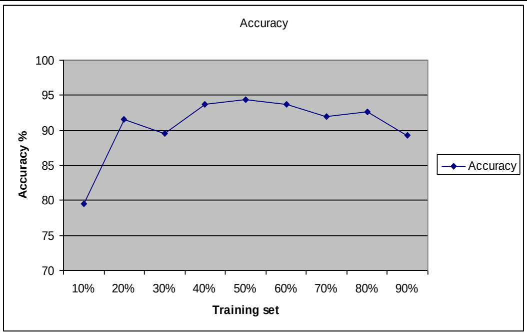
Figure 3. Confusion matrix with balanced data sent.

Comparative Accuracy

After varying the percentage of the learning set and the test set, we create 9 different set of data. Figure 3 show the relationship between the size of the learning set and the test set.



(a)

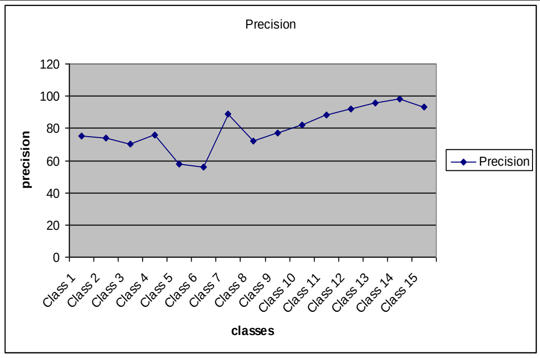


(b)

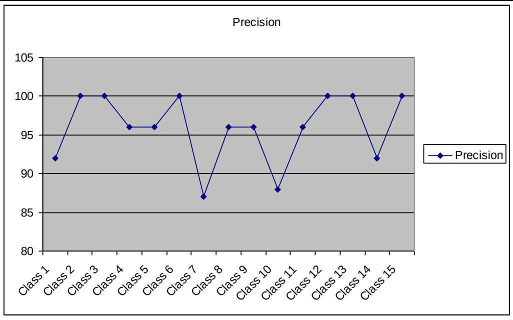
Figure 4: Relation between the size of the learning set and accuracy. (a) Imbalance data set (b) Balanced data set.

Better result are obtained when the size of the training set ranges between 40 to 65 % as shown in (a) and (b) in Figure 4. On the other hand taking the same number of data for each class in the training set (balanced data set) can improve the accuracy of the system from 87.3% to 96.33%. Accuracy is misled by the class with high support, thus reducing the overall accuracy as shown in (a).

Precision, recall, f1-score



(a)

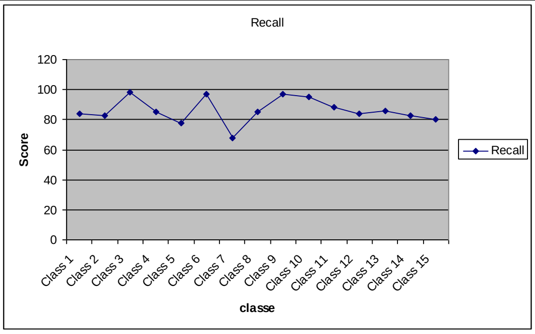


(b)

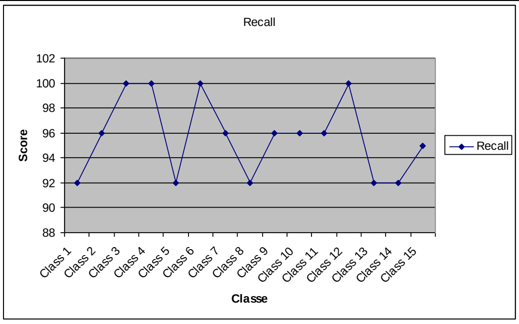
Figure 5: Precision (a) Precision for imbalance data set (b) Precision for balanced data set.

Figure 5 shows that the balanced data set achieves high precision than the imbalance data set, reaching 6 times a precision of 100%, on the other hand, the highest precision for classes in the imbalance data set reached hardly reached 100%. Figure 5 represents the sensitivity of balanced and imbalance data set.

In term of sensitivity, the balanced data set, produce a higher rate of sensitivity than the imbalance data set. F1-score conveys the balance between the precision and the recall, when dealing with imbalance classes, classification alone cannot be trusted to select a well performing model. Figure 8, shows the relationship between precision, recall and f1-score.



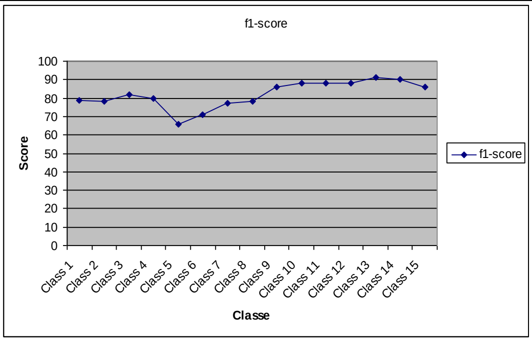
(a)



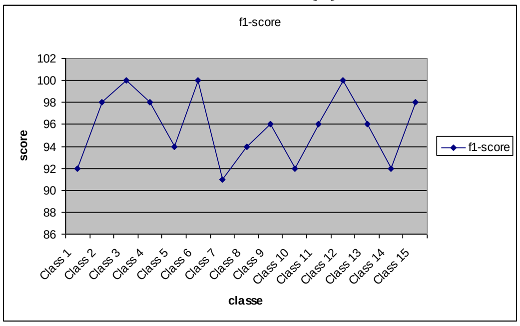
(b)

Figure 6: Recall (a) Precision for imbalance data set (b) Precision for balanced data set.

The support of the class in the training set has a positive impact on the precision of the class, especially when we have imbalance class in the training set. Figure 7 shows the relationship between the support and the precision.

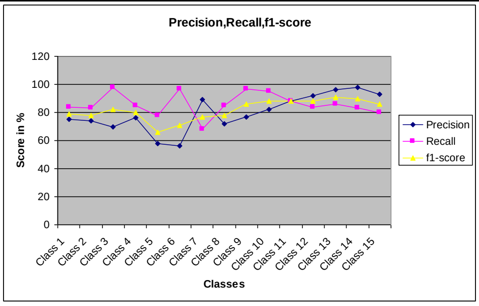


(a)

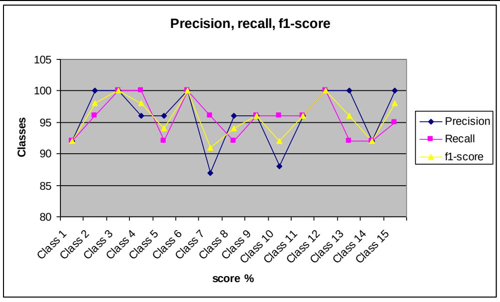


(b)

Figure 7: f1 score (a) Precision for imbalance data set (b) Precision for balanced data set.



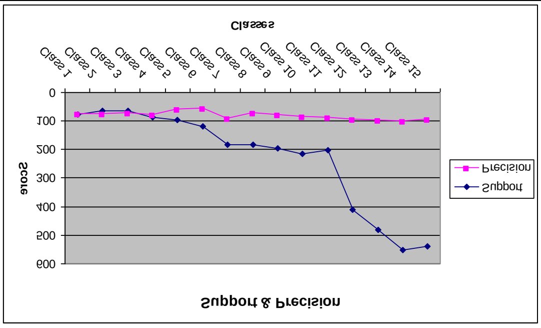
(a)



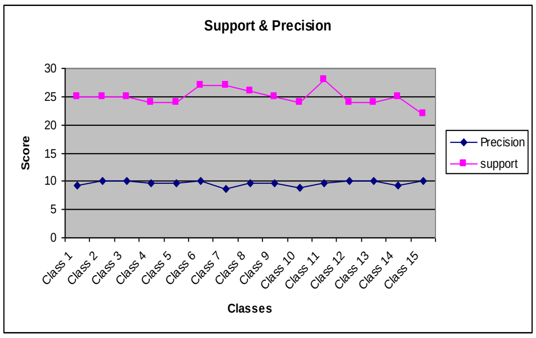
(b)

Figure 8: Precision (a) Precision for imbalance data set (b) Precision for balanced data set.

For imbalance class in the training set, the support of a class has a positive impact on the precision of the class; however, classes with a very high accuracy can create an accuracy paradox problem by predicting the value of the majority class for all predictions and achieve high classification accuracy.



(a)



(b)

Figure 9: Relationship between the support and the precision. (a) Relationship between the support and precision for imbalance data set (b) Relationship between the support and precision for balanced data set.