Learning from Crowds in Multi-dimensional Classification Domains

Jerónimo Hernández-González, Iñaki Inza, and José A. Lozano

Intelligent Systems Group, University of the Basque Country UPV/EHU
{jeronimo.hernandez,inaki.inza,ja.lozano}@ehu.es

Abstract. Learning from crowds is a recently fashioned supervised classification framework where the true/real labels of the training instances are not available. However, each instance is provided with a set of noisy class labels, each indicating the class-membership of the instance according to the subjective opinion of an annotator. The additional challenges involved in the extension of this framework to the multi-label domain are explored in this paper. A solution to this problem combining a Structural EM strategy and the multi-dimensional Bayesian network models as classifiers is presented.

Using real multi-label datasets adapted to the crowd framework, the designed experiments try to shed some lights on the limits of learning to classify from the multiple and imprecise information of supervision.

Keywords: Multi-label classification, multi-dimensional classification, Learning from crowds, Structural EM method, Bayesian network models.

1 Introduction

The process of training a classifier in the standard supervised classification paradigm requires a training dataset of examples which are class-labeled by a domain expert, who establishes to which class each example belongs. Other related paradigms, under the general name of partially supervised classification, deal with datasets in which the expert is not able to label completely/certainly all the training examples. In one way or another, all these paradigms provide expert supervision of the training data. Moreover, the reliability of this information of supervision is a strong assumption, based on which most of the techniques taking part in the learning process have been developed (evaluation techniques, performance scores, learning methods, etc.). However, obtaining this kind of reliable supervision can be expensive and difficult, even for a domain expert.

In the last decades, the Web has emerged as a large source of information, providing a quick and easy way to collect data. Actually, the main drawback of the data collected in this way is its reliability. As it has been usually produced by non-expert annotators, this subjective data may involve incompleteness, impreciseness and/or incorrectness. Learning from noisy data (or labeled by an unreliable annotator) is a known problem [2,18]. However, these new technologies provide an easy and cheap way to obtain *not one but many* different personal

C. Bielza et al. (Eds.): CAEPIA 2013, LNAI 8109, pp. 352–362, 2013.

opinions about the class-membership of a given example. Thus, the idea of learning from data labeled by taking into account diverse and multiple (subjective) opinions has led to a new learning paradigm.

In the related literature, the problem of learning from multiple noisy labelers or annotators is known as *learning from crowds* [9]. In this problem, the real class-membership information of the training instances is not provided. However, a crowd of mainly non-expert labelers provides different subjective (noisy) opinions about the class-membership of the training instances. Note the differences with [6], where the opinions of a fixed number of *domain experts* have to be combined.

Learning to classify from this kind of data is possible and useful [12,14]. The learning algorithm has to cope with the individual unreliability of the annotators in order to build accurate classifiers from the consensus opinion. The ability to learn an accurate classifier from a given dataset of this type is largely influenced by two related factors: the quality of the annotators and the degree of consensus between them. Learning can be feasible even when the annotators do not have a high reliability if, for each instance, a subset of annotators agree in their predictions. Based on both concepts, we present our initial solution to the additional challenges which involves the application of the learning from crowds paradigm to multi-label classification (in a broader sense than [11]).

The rest of the paper is organized as follows. In the next section, a formal definition of the problem is presented, together with its adaptation to the multi-dimensional classification framework. Then, our method (an adaptation of a state-of-the-art algorithm) for learning multi-dimensional Bayesian network classifiers from this kind of data is described. Next, the experiments show some limits in the learning ability of our method (according to *noise rate* and *consensus degree*). And finally, some conclusions and future work are presented.

2 Learning from Crowds in Multi-label Domains

In the problem of multi-label learning from crowds, the examples are provided without the true labels (a.k.a. gold-standard), and only the label(s) assigned by multiple (non-expert) annotators are available. Here, an annotator assigns one or several labels to an instance according to their subjective opinion.

Like the classical multi-label (ML) learning paradigm, the problem is described by a set of n predictive variables $(X_1, \ldots, X_n)$ and a class variable C. Moreover, $\mathcal{X}$ denotes the instance space (all the possible value assignments to the n predictive variables) and $\mathcal{C} = \{c_1, \ldots, c_q\}$ denotes the label space (the set of q possible class labels). A ML dataset $D = \{(\mathbf{x}_1, \mathbf{c}_1), (\mathbf{x}_2, \mathbf{c}_2), \ldots, (\mathbf{x}_m, \mathbf{c}_m)\}$ consists of a set of m examples of the problem, where $\mathbf{x}_i \in \mathcal{X}$ is a n-tuple that assigns a value to each predictive variable and $\mathbf{c}_i \subseteq \mathcal{C}$ is the corresponding set of class labels, denoting the class-membership of the example.

Similarly, the dataset D in a multi-label learning from crowds framework is composed of m examples $D = \{(\mathbf{x}_1, \mathcal{A}_1), (\mathbf{x}_2, \mathcal{A}_2), \ldots, (\mathbf{x}_m, \mathcal{A}_m)\}$, which are assumed to have been sampled i.i.d. from some underlying probability distribution. Each instance $\mathbf{x}_i$ is provided together with a group $\mathcal{A}_i$, which contains the labels (annotations) provided by different annotators: $\mathcal{A}_i = \{\mathbf{c}_{i1}, \ldots, \mathbf{c}_{it}\}$, with $\mathbf{c}_{ij} \subseteq \mathcal{C}$

and t being the number of annotators. As a classical multi-label classification problem, the objective is to infer the class label(s) of new unseen instances.

A Transformation to Multi-dimensional Classification. In a multi-dimensional (MD) classification problem [1,10], there is more than one class variable $(C_1, \ldots, C_d)$, and each one has its own set of possible labels. In this case, the label space $\mathcal{C} = \mathcal{C}_1 \times \cdots \times \mathcal{C}_d$ denotes all the possible joint label assignments to the d class variables (label configurations). An example $(\mathbf{x}_i, \mathbf{c}_i)$ of a MD training dataset includes a d-tuple $\mathbf{c}_i \in \mathcal{C}$ that assigns a label to each class variable, apart from the instance predictive values $\mathbf{x}_i \in \mathcal{X}$. Given a new instance, the multi-dimensional classifier predicts a class label for each class variable.

In this paper, in order to deal with the presented multi-label problem, we transform it to the multi-dimensional classification framework. As explained in the related literature [1,10,17], the multi-label learning paradigm can be described as a multi-dimensional problem in which there are as many binary class variables as class labels in the multi-label problem ($d_{MD} = q_{ML}$). Thus, each binary class variable (MD) represents the presence/absence of a class label (ML).

The adapted dataset D of multi-dimensional learning from crowds is composed of m examples $D = \{(\mathbf{x}_1, \mathbf{A}^1), (\mathbf{x}_2, \mathbf{A}^2), \ldots, (\mathbf{x}_m, \mathbf{A}^m)\}$, where the information of supervision for each instance $\mathbf{x}_i$ is provided in a $(t \times d)$-matrix $\mathbf{A}^i$. Thus, the position A^i_{ac} indicates the class label predicted for the class variable C_c by the annotator L_a.

3 Learning from Crowds in Multi-dimensional Domains

The main characteristic of the learning from crowds framework is the availability of much and diverse information of supervision. A natural solution to this problem could be the transformation of the crowds information to some kind of probabilistic supervision. From this point of view, the problem is closely related with other problems with imprecise labels such as learning with partial labels [3], learning from probabilistic information [8], etc. Nevertheless, in the presence of imprecise or incorrect data, it is worth modeling the source of noise.

As explained previously, the crowd supervision consists of the class labels assigned to the instances according to the subjective opinion of several annotators. Certainly, each annotator can be considered as a source of noise. Based on this idea, Raykar et al. [9] proposed an EM-based algorithm to solve the learning from crowds problem in single-dimensional domains, using a set of weights to model the reliability of the annotators.

Under the realistic assumption that the annotators might show different reliability in different prediction tasks, we have extended the idea of Raykar et al. [9] to the multi-dimensional paradigm, independently modeling the reliability of each annotator predicting each class variable.

To sum up, we have reformulated the problem as searching the weights (w_{ac}) that better describe the ability of each annotator, L_a, to predict each class variable C_c, and leading to the generation of accurate classifiers. For solving both interrelated problems, we propose a learning algorithm based on the Structural

Expectation-Maximization (SEM) strategy, which iteratively alternates to improve the initially obtained reliability weights (several techniques are proposed) and to look for an improved fit of the model. A basic adaptation of a state-of-the-art local-search algorithm is used to learn the model, a multi-dimensional Bayesian network classifier (MBC [1]), from crowd data augmented with reliability weights. For the sake of simplicity, in this paper the number of annotators is fixed, i.e. all the instances are annotated by all the annotators.

3.1 Our Structural EM Strategy

A MBC [1] is a Bayesian network $\mathbb{M} = (G, \boldsymbol{\theta})$ defined over a set $\mathcal{V} = \{V_1, \ldots, V_v\}$ of random variables, where $G = (\mathcal{V}, \mathcal{R})$ is an acyclic directed graph and $\boldsymbol{\theta}$ its parameters. As a classifier, the set of variables can be divided in class variables, $\mathcal{V}_C = \{C_1, \ldots, C_d\}$, and predictive variables, $\mathcal{V}_X = \{X_1, \ldots, X_n\}$, where $v = n + d$. The graph of a MBC cannot contain arcs in $\mathcal{R}$ from the predictive ($\mathcal{V}_X$) to the class variables ($\mathcal{V}_C$).

The Structural EM strategy (SEM), proposed by Friedman [5], provides a suitable framework to infer both the graph structure and the model parameters of a Bayesian network model from missing data. The EM strategy, proposed by Dempster et al. [4], is used in our framework to obtain the maximum likelihood parameters from multiple weighted annotations. Iteratively, the method estimates the reliability weights of the annotators given the current fit of the model, and re-estimates the model parameters. Under fairly general conditions, the iterative increment of the likelihood has been proved to converge to a stationary value (most of the times, a local maximum) [7]. Additionally, the SEM strategy incorporates an outer loop to the parametric-convergence loop of the classical EM, and iteratively improves an initially-proposed structure.

In Algorithm 1, a pseudo-code of the SEM method developed in this paper is shown. In the following subsections, the different tasks of this method are explained in detail: the initialization of the reliability weights (line 3 in Algorithm 1) and their improvement (line 10); the structural learning (line 4) and structural improvement (line 14); and the parametric learning (line 9).

3.2 Reliability Weights of the Annotators

As previously mentioned, we use weights w_{ac} to indicate the reliability of the predictions of the annotator L_a for the class variable C_c. These weights are initialized in the first stage of the SEM method and updated iteratively.

Initializing Weights. Similar to [13], our SEM method initializes the weight w_{ac} as the ability of the annotator L_a to agree with other annotators (consensus) in the label assigned to class variable C_c, averaging over all the instances of the dataset. That is,

$$w_{ac} = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{t-1} \sum_{a' \neq a} \mathbb{I}[A^i_{a'c} = A^i_{ac}] \tag{1}$$

where m is the number of instances, t is the number of annotators and $\mathbb{I}[condition]$ is a function that returns 1 when the *condition* is true, and 0 otherwise.

Algorithm 1 Pseudo-code of our Structural EM method.

procedure StructuralEM($D, maxIt, \epsilon$) ▷ D: dataset
  $i = 0$ ▷ $maxIt$: max. number of iterations
  $\mathbf{W} \leftarrow$ initializeWeights(D) ▷ ϵ: threshold (stop condition)
  $G_i \leftarrow$ structuralLearning($D, \mathbf{W}$)
  repeat
    $j = 0$
    repeat
      $\hat{\mathbf{W}} \leftarrow \mathbf{W}$
      $\boldsymbol{\theta_j} \leftarrow$ parametricLearning($D, \mathbf{W}, G_i$)
      $\mathbf{W} \leftarrow$ estimateWeights($D, G_i, \boldsymbol{\theta_j}$)
      $j = j + 1$
    until (diff($\mathbf{W}, \hat{\mathbf{W}}$) $< \epsilon$) Or ($j = maxIt$)
    $i = i + 1$
    $G_i \leftarrow$ findMaxNeighborStructure($D, \mathbf{W}, G_{i-1}$)
  until ($G_i = G_{i-1}$) Or ($i = maxIt$)
  return $\mathbb{M} \equiv (G_i, \boldsymbol{\theta_j})$
end procedure

Weights Updating. To update the reliability weights, four alternative procedures has been developed: two model-based procedures (using the most probable label configuration; or using the probabilities of all the possible label configurations), both of them combined or not with the consensus concept.

On the one hand, the information provided by the model $\mathbb{M}$ (learnt in the previous EM iteration) is used in two ways. In a first approach, the label configuration of maximum joint probability $\breve{\mathbf{c}}$ given the instance is calculated. Then, each weight w_{ac} is updated as the mean accuracy of the annotator L_a over the class variable C_c, using each maximal configuration $\breve{\mathbf{c}}_i$ as the golden truth:

$$w_{ac} = \frac{1}{m} \sum_{i=1}^{m} \mathbb{I}[\breve{\mathbf{c}}_{ic} = A^i_{ac}] \tag{2}$$

In the second approach, for each instance the marginal probability of each class variable is calculated using the model $\mathbb{M}$. Subsequently, these probabilities are used to update each weight of an annotator by averaging the probability of their predictions for the given class variable over the whole dataset,

$$w_{ac} = \frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{|\mathcal{C}|} p^{\mathbb{M}}(\overline{\mathbf{c}}^j | \mathbf{c}_i) \cdot \mathbb{I}[\overline{c}^j_c = A^i_{ac}] \tag{3}$$

where $\mathcal{C}$ is the label space (set of all the label configurations) and $\overline{\mathbf{c}}^j \in \mathcal{C}$.

On the other hand, the weight-updating process can *remember* the mean degree of consensus. Thus, the reliability weights are updated according to the function, $w_{ac} = (w^{\text{Cons}}_{ac} + w^{\mathbb{M}}_{ac})/2$, where w^{Cons}_{ac} is the consensus weight (calculated by means of Eq. 1) and $w^{\mathbb{M}}_{ac}$ is the model-based weight (calculated with either Eq. 2 or Eq. 3). Therefore, as both model-based functions can be extended with the consensus idea, we finally have four weight-updating techniques.

3.3 Estimating the Model Parameters from Crowds in Multi-dimensional Domains

In this paper, the parameters of the MBC are estimated by frequency counts, as usual. In order to cope with the weighted and multi-labeled class information provided by the crowds, we have adapted the procedure to collect frequency counts. Thus, given an instantiation $(u_1, \ldots, u_j)$ of a set of variables $\mathcal{U}_{\mathbf{u}} = \{U_1, \ldots, U_j\} \subseteq \mathcal{V} = (\mathcal{V}_X, \mathcal{V}_C)$, the posterior probability is defined as,

$$p(u_1, \ldots, u_i | u_{i+1}, \ldots, u_j) = N(u_1, \ldots, u_i, u_{i+1}, \ldots, u_j) / N(u_{i+1}, \ldots, u_j)$$

where $N(\cdot)$ represents the counts obtained from the provided dataset. In this problem, they are calculated as follows:

$$N(\mathbf{u}) = \frac{1}{\sum_{a=1}^{t} \mathbf{W}_a^{\downarrow \mathbf{u}}} \sum_{a=1}^{t} \mathbf{W}_a^{\downarrow \mathbf{u}} \sum_{\mathbf{y} \in \mathcal{X}(D, \mathbf{A}_a)} \mathbb{I}[y_{[U_1]} = u_1, \ldots, y_{[U_{|\mathbf{u}|}]} = u_{|\mathbf{u}|}]$$

where $[U_j]$ indicates the index of the variable $U_j \in \mathcal{U}_{\mathbf{u}}$ in the original set of variables $\mathcal{V}$ and $\mathcal{X}(D, \mathbf{A}_a)$ is the set of instances D labeled according to the annotations $\mathbf{A}_a$ of annotator L_a. In the specific count, the weight assigned to annotator L_a ($\mathbf{W}_a^{\downarrow \mathbf{u}}$) is calculated as the product of the weights per variable, taking into account only those variables in $\mathcal{U}_{\mathbf{u}}$:

$$\mathbf{W}_a^{\downarrow \mathbf{u}} = \prod_{U \in \mathcal{U}_{\mathbf{u}}} w_{a[U]} \tag{4}$$

As previously shown, our SEM method only estimates the weights w_{ac} of the class variables ($C_c \in \mathcal{V}_C$). Consequently, regarding Eq. 4, the weights of the predictive variables ($X_x \in \mathcal{V}_X$) are considered constant, $w_{ix} = 1$. In practice, the estimator implements the Laplacian correction in order to avoid zero counts.

3.4 Local Search for Structural Learning

Our method to learn the structure of a MBC B from the data (line 4 in Algorithm 1) is based on the wrapper algorithm of Larrañaga et al. [1]. Following their proposal, at each iteration of the local search, the arc inclusion/deletion (candidate change) that, respecting a fixed ancestral order, most improves the score of the current structure is chosen. The candidate changes are evaluated using the log K2 score:

$$\log P(B, D) = \sum_{i=1}^{v} \sum_{j=1}^{q_i} \log \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \sum_{k=1}^{r_i} \log N_{ijk}!$$

where v is the number of variables, r_i is the number of values that the variable V_i can take, and q_i is the number of possible configurations of the parents of V_i. As the log K2 score is decomposable, the arc inclusion/deletion can be evaluated only taking into account the arc-destination variable and its parents.

Structural MBC Improvement. The function at line 14 of the Algorithm 1 performs a single local-search step in the MBC structure space in order to find a better fit of the model. In practice, the structural improvement is chosen using the same procedure as the structural learning method presented before, but restricted to a single step.

4 Experiments

In this section, the two factors that we have used to describe the amount of information provided in the learning from crowds problem are tested. Due to the lack of time, we have not managed to obtain real crowd datasets[1]. However, we have designed a strategy to simulate multiple annotators controlling the noise rate and the consensus between them. Thus, three real multi-label datasets[2] have been adapted to simulate multiple-annotated datasets.

Generation of Annotators. We have implemented a strategy for generating annotators from the real class labels of the ML datasets. For each class variable, starting from the true labels, a user-specified percentage of these labels —randomly selected— are fixed (well-labeled instances). The rest of labels are swapped with probability 0.5 in order to introduce the characteristic noise of this kind of data.

The degree of consensus is controlled by sharing the same fixed set of well-labeled instances between a user-specified number of annotators. Then, an extra (small) rate of changes is applied to each annotator individually in order to generate low divergence between them.

In both experimental settings, 10 annotators have been generated (this selection is based on the discussion of Snow et al. [14]), and all of them annotate all the class variables and instances. By default, the method uses the provided indexation of variables as ancestral order (always respecting that the class variables appear before the predictive variables). Regarding the learned models, the MBC have been restricted to a maximum of $K = 3$ parents per variable.

4.1 Influence of the Noise Rate

The first set of experiments has been designed in order to test the ability of our learning method to cope with an increasing amount of noise in the annotations. In this way, the consensus degree has been fixed to four annotators [14] and different values (four) of mean noise rate have been tested for each dataset. Three real ML datasets (emotions, scene and yeast) have been used to simulate the information of crowds. Moreover, for each designed test, ten datasets have been generated, summing up to the total number of datasets, 120 (4 error rates, 3 datasets, 10 repetitions).

[1] For future work, Mechanical Turk (`http://www.mturk.com`) is an online platform that allows to easily collect data from crowds.

[2] Multi-label datasets available at: `http://mulan.sourceforge.net/datasets.html`

Table 1. Experiments developed to test the noise rate influence. The three datasets are evaluated in a 10×5-fold CV according to four measures [1], and the results are shown in terms of the mean value and the corresponding standard deviation (each experiment is repeated over 10 equal-generated crowd datasets).

Noise rate	10%	20%	30%	40%	10%	20%	30%	40%
microf1	0.59 ± 0.01	0.56 ± 0.02	0.50 ± 0.03	0.41 ± 0.03	0.44 ± 0.02	0.35 ± 0.02	0.24 ± 0.03	0.13 ± 0.03
macrof1	0.58 ± 0.02	0.54 ± 0.02	0.48 ± 0.03	0.41 ± 0.03	0.44 ± 0.02	0.35 ± 0.01	– – –	– – –
globalAcc	0.24 ± 0.01	0.22 ± 0.01	0.18 ± 0.02	0.11 ± 0.02	0.34 ± 0.02	0.26 ± 0.02	0.15 ± 0.02	0.07 ± 0.02
meanAcc	0.72 ± 0.01	0.70 ± 0.01	0.69 ± 0.02	0.69 ± 0.01	0.83 ± 0.01	0.82 ± 0.01	0.82 ± 0.00	0.83 ± 0.00
	emotions				scene			

Noise rate	10%	20%	30%	40%
microf1	0.59 ± 0.01	0.57 ± 0.01	0.56 ± 0.01	0.54 ± 0.01
macrof1	– – –	– – –	– – –	– – –
globalAcc	0.15 ± 0.01	0.13 ± 0.01	0.11 ± 0.01	0.09 ± 0.02
meanAcc	0.76 ± 0.02	0.75 ± 0.01	0.74 ± 0.01	0.74 ± 0.01
	yeast			

In Table 1, the results obtained from these experimental settings show the expected tendency of an increment of the degradation as the noise rate is larger. However, as a result of the transformation to the multi-dimensional framework, the resulting class variables tend to be strongly unbalanced (a class value is over-represented in the dataset). Among the problems that this generates, note that the accuracy-based evaluation measures become unfair. For example, in the tests with most noise of Table 1, some of the displayed mean accuracy values correspond to the label proportions of the dataset (which have macrof1 = '– – –'), i.e. the method is always predicting the majority class label. In this way, our weight updating procedure based on maximal-probability could be failing to capture the information of supervision as a combination of the multiple annotations.

4.2 Influence of the Consensus

In the second set of experiments, we show the behavior of our method when the consensus between the labelers increases. Thus, following the procedure described before, five groups of ten datasets were generated where the degree of consensus ranges from two to six annotators. The annotators in consensus have been generated with a noise rate of 10%, and the rest with 30%. Due to lack of space, only the ML dataset *emotions* is used in this experimental settings.

As a fundamental parameter in our approach, we wanted to show the reliability weights of the annotators obtained after the training process. As shown in Figure 1, all the weight updating procedures identify the reliable annotators (all of them are shown over 1). However, the weights produced by the approaches that incorporate the consensus idea are those which are most unbalanced. Surprisingly, larger consensus annotators' weights do not imply a notable gain in terms of global accuracy (nor other performance measures, not shown due to lack of space). Our method behaves as expected when it performs better as the degree of consensus is increased. However, the performances do not show

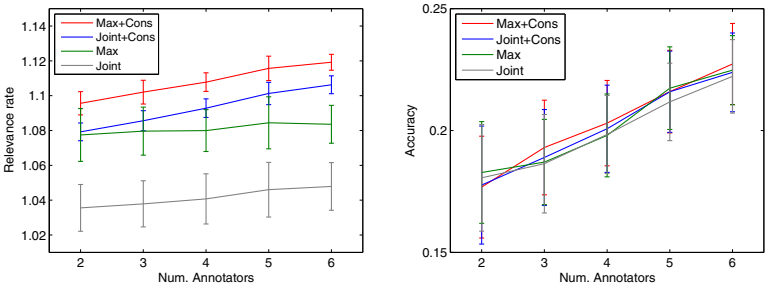

Fig. 1. Experiments developed to test the influence of the consensus degree. In the left figure, the relevance rate of the consensus annotators (mean of the weights of the consensus annotators divided by that of non-consensus annotators) according to different weight-updating approaches. In the right figure, the same experiments are evaluated in terms of global accuracy. All the results are shown by means of mean value and the corresponding standard deviation, evaluated in a 10×5-fold CV (repeated over 10 equal-generated crowd datasets).

notable differences whether the weight-updating approach considers the consensus information or not.

5 Conclusions

As shown, the current method does not seem to make the most of this kind of data, being unable to extract information from the consensus between annotators. It could be worth exploring other paradigms to weight the relevance of the annotators, according to other performance metrics (see, for example, [16]).

As explained before, our method implements four approaches for updating the weights of the annotators. Specifically, two of them only consider the model predictions in the update procedure, and the other two combine the model estimations and the consensus information, both with the same relevance. An interesting idea for future work could be to implement a simulated-annealing based technique that modifies the relevance of both factors (model predictions and consensus) throughout the iterative method every time that the weights are updated. In this way, in the first iterations we could rely more on the consensus information and, in the final iterations, relying on the model predictions.

Moreover, considering that the annotators can choose the instances that they label, we could skip the previous assumption that all the annotators label all the instances. Similarly, it could be also interesting to allow annotators not to assert the membership of every instance to all the classes [15]; that is, to consider a new state for the annotations (member, non-member, *unknown*). Skipping both assumptions would introduce new challenges to the learning process, mainly affecting the way in which we calculate the reliability weights of the annotators.

Acknowledgments. This work has been partially supported by the Saiotek and IT609-13 programs (Basque Government), TIN2010-14931 (Spanish Ministry of Science and Innovation), network in computational bio-medicine (Carlos III Health Institute), 2011-CIEN-000060-01 (Diputación Foral de Gipuzkoa) and CRC-Biomarkers project (6-12-TK-2011-014, Diputacióon Foral de Bizkaia). Jerónimo Hernández-González holds a grant (FPU) from the Spanish Ministry of Science and Innovation. Thanks to Aritz Pérez-Martínez for his helpful comments.

References

1. Bielza, C., Li, G., Larrañaga, P.: Multi-dimensional classification with Bayesian networks. International Journal of Approximate Reasoning 52(6), 705–727 (2011)
2. Brodley, C.E., Friedl, M.A.: Identifying mislabeled training data. Journal of Artificial Intelligence Research 11, 131–167 (1999)
3. Cour, T., Sapp, B., Taskar, B.: Learning from partial labels. Journal of Machine Learning Research 12, 1501–1536 (2011)
4. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society. Series B (Methodological) 39(1), 1–38 (1977)
5. Friedman, N.: Learning belief networks in the presence of missing values and hidden variables. In: Proceedings of the 14th ICML, pp. 125–133 (1997)
6. López-Cruz, P.L., Larrañaga, P., DeFelipe, J., Bielza, C.: Bayesian network modeling of the consensus between experts: An application to neuron classification. International Journal of Approximate Reasoning (in press, 2013)
7. McLachlan, G.J., Krishnan, T.: The EM Algorithm and Extensions (Wiley Series in Probability and Statistics). Wiley Interscience (1997)
8. Nguyen, Q., Valizadegan, H., Hauskrecht, M.: Learning classification with auxiliary probabilistic information. In: Proceedings of the 11th IEEE International Conference on Data Mining (ICDM 2011), pp. 477–486 (2011)
9. Raykar, V.C., Yu, S., Zhao, L.H., Valadez, G.H., Florin, C., Bogoni, L., Moy, L.: Learning from crowds. Journal of Machine Learning Research 11, 1297–1322 (2010)
10. Rodríguez, J.D., Martínez, A.P., Arteta, D., Tejedor, D., Lozano, J.A.: Using multidimensional bayesian network classifiers to assist the treatment of multiple sclerosis. IEEE Transactions on Systems, Man, and Cybernetics 42(6), 1705–1715 (2012)
11. Sellamanickam, S., Tiwari, C., Selvaraj, S.K.: Regularized structured output learning with partial labels. In: Proceedings of the 12th SDM, pp. 1059–1070 (2012)
12. Sheng, V.S., Provost, F.J., Ipeirotis, P.G.: Get another label? improving data quality and data mining using multiple, noisy labelers. In: Proceedings of the International Conference on Knowledge Discovery and Data Mining, pp. 614–622 (2008)
13. Smyth, P., Fayyad, U., Burl, M., Perona, P., Baldi, P.: Inferring ground truth from subjective labelling of venus images. In: Proceedings of the Advances in Neural Information Processing Systems (NIPS), pp. 1085–1092 (1994)
14. Snow, R., O'Connor, B., Jurafsky, D., Ng, A.Y.: Cheap and fast - but is it good? evaluating non-expert annotations for natural language tasks. In: Proceedings of the Conference on Empirical Methods in NLP, pp. 254–263 (2008)

15. Sun, Y.Y., Zhang, Y., Zhou, Z.H.: Multi-label learning with weak label. In: Proceedings of the 24th AAAI Conference on Artificial Intelligence, AAAI 2010 (2010)
16. Younes, Z., abdallah, F., Denœux, T.: Evidential multi-label classification approach to learning from data with imprecise labels. In: Hüllermeier, E., Kruse, R., Hoffmann, F. (eds.) IPMU 2010. LNCS, vol. 6178, pp. 119–128. Springer, Heidelberg (2010)
17. Zhang, M.L., Zhou, Z.H.: A review on multi-label learning algorithms. IEEE Transactions on Knowledge and Data Engineering (in press, 2013)
18. Zhu, X., Wu, X., Chen, Q.: Eliminating class noise in large datasets. In: Proceedings of the 20th ICML, pp. 920–927 (2003)