Multiple Instance Learning for Training Neural Networks under Label Noise

Stefan Duffner and Christophe Garcia
LIRIS, UMR 5205 CNRS, INSA-Lyon
F-69621 Villeurbanne, France
stefan.duffner@liris.cnrs.fr, christophe.garcia@liris.cnrs.fr

***Abstract*—In this paper, we present an extensive study of different neural network-based approaches and loss functions applied to the Multiple Instance Learning (MIL) problem and binary classification. In the MIL setting, training is performed on small sets of instances called bags, where each positive bag contains at least one positive instance and each negative bag contains only negative instances. We propose a new loss function based on the generalised mean and an effective training strategy particularly suited to this setting and to problems where the instances of one class contain a considerable amount of label noise. Furthermore, we present a probabilistic approach to dynamically estimate the label noise in this unbalanced binary classification setting and utilise it to automatically modulate the hyper-parameter of our proposed loss function. We experimentally evaluated our approach on a number of standard benchmarks for binary classification and showed that it outperforms standard neural network optimisation algorithms as well as most state-of-the-art MIL methods, both on numerical/categorical vector data with MLP architectures and images with Convolutional Neural Networks.**

***Index Terms*—Multiple Instance Learning, Label Noise, Neural Networks**

I. Introduction

Multiple Instance Learning (MIL) denotes a specific type of supervised learning, where instances are grouped into sets, called bags, and labels are only available at the *bag level* and not for each individual instance. Traditionally, MIL has been used for binary classification problems where a bag is labelled positive if it contains *at least one* positive instance and labelled negative otherwise, *i.e.* if *all* instances are negative. This type of unbalanced scenario has numerous applications in different fields, where the labels for one of the classes are uncertain or missing. For example, for spam e-mail classification, credit card fraud detection, medical data classification, image classification and video analysis.

The MIL setting is challenging for many machine learning algorithms, as label information is unbalanced across classes, and it is not known which features and which particular instances in the positive (and negative) bags are discriminant and relevant. Note that this problem is different from "simple" classification with unbalanced data [1], [2], as in addition to an under- or over-represented class in the training set, the labels of one of the classes are noisy or not assigned to every instance. Traditional approaches tackle this problem with specific decision tree-based or feature selection and projection algorithms or with classical methods like k-Nearest Neighbours, Support Vector Machines or Neural Networks that have been modified to cope with bags of multiple instances. However, approaches that iteratively minimise an objective function, like standard Neural Networks through gradient descent-based algorithms, face particular challenges since many cost functions suitable for MIL are not differentiable and non-convex. Nevertheless, neural network models have some properties that are desirable in many machine learning settings and practical applications. For example, their robustness to noise, their modularity and layer-wise structure leading to a wide variety of training strategies and architectures adapted to specific machine learning problems, and, finally, their capacity to automatically extract, select and combine features from data at different levels of abstraction. Also, recent advances on deep neural network models with a wide variety of different architectures have led to state-of-the-art performance on numerous applications. However, their robustness to label noise and their applicability to the MIL setting has not been fully studied yet.

In this paper, we will introduce a new algorithm for training neural network for binary classification with label noise using a MIL approach. We will present different loss functions that have been proposed for MIL, and investigate their behaviour and performance on standard binary classification benchmarks where different amounts of unbalanced label noise is introduced. To conduct a controlled and extensive study, we modified the training sets such that the labels of a certain proportion of *positive* instances is switched to *negative*. Then, we will propose a new loss function based on the generalised mean computed on the bags. The advantage of this function is that it is differentiable everywhere, and, compared to functions proposed in the literature, it has a parameter $p > 0 \in \mathbb{R}$ that controls the way the individual instances in the bag contribute to the bag label prediction, ranging from the standard arithmetic mean ($p = 1$) to the $\max$ function ($p \to \infty$). We further propose an algorithm to automatically determine an appropriate p depending on the estimated label noise in the training dataset. We experimentally show on different datasets and neural network architectures that our generalised mean-based training approach outperforms other loss functions and MIL algorithms in most cases. We evaluated its performance in terms of MIL classification rate using standard benchmarks and compared it to the state of the art, and its robustness to label noise on public datasets where we introduced a considerable amount of noise.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

II. Related Work

The MIL problem has been first formulated by Dietterich *et al*. [3] who proposed an iterative algorithm to select discriminative features defined by so-called Axis-Parallel Rectangles (APR). They evaluated their approach on the drug prediction dataset "MUSK" which has been widely used as a benchmark since then. Auer [4] presented a theoretical complexity analysis and developed an efficient APR-based algorithm called MULTINST.

Maron *et al*. [5] proposed the *Diverse Density* (DD) algorithm that considers each bag as a manifold and identifies positive instances by searching intersection points of positive manifolds that are far from the negative ones. An extension called EM-DD has been proposed by Zhang and Goldman [6] modelling the instance labels as latent variables estimated by the EM algorithm. Wang and Zucker [7] developed a method called Citation-kNN based on the k-Nearest Neighbour algorithm using a reciprocal distance measure between bags based on the Hausdorff distance. Two approaches based on Support Vector Machines (SVM) have been proposed by Andrews *et al*. [8]. The first one, called mi-SVM, takes into account the unknown instance labels of the bags, whereas the second one, called MI-SVM, tries to maximise the margin at the bag level. The optimisation is performed using mixed-integer quadratic programming. Another instance-level kernel-based method has been proposed by Gärtner *et al*. [9] using a linear, polynomial or Gaussian kernel measuring the similarity between bags. Since then, many other kernel-based MIL algorithms have been proposed [10]–[15].

Another effective approach for MIL is to train decision trees. Ruffo [16], for example, in his RELIC method defined multiple-instance concepts that are used as decision rules. Chevaleyre and Zucker [17] modified the ID3 algorithm by introducing a new bag-level split criterion called multi-instance entropy. Similarly, Blockeel *et al*. [18], in their MITI algorithm, proposed a modified Gini impurity measure for the tree construction. A Random Forest approach has been proposed by Leistner *et al*. [19], where the unknown instance labels are retrieved by several training runs in an deterministic annealing procedure. Straehle *et al*. [20] extended the MITI method with a non-linear splitting rule and an optimal tree output combination strategy in a random forest framework. Weidmann *et al*. [21] developed a *generalised multiple instance learning* framework by defining three different problems: presence-based, threshold-based and count-based MIL, and they proposed a two-level classification method for solving these problems using a decision tree approach.

Recently, Komárek and Somol [22], presented Bag-Level Randomised Trees (BLRT), where extremely randomised trees [23] are trained using a bag-level splitting criterion based on the number of positive instances in a bag.

Also neural network-based MIL approaches have been presented in the literature. Ramon and De Raedt [24] were one of the first to propose a neural network for MIL, trained with backpropagation and a smooth *max* function, *i.e*. log-exp-sum, to compute instance probabilities. Zhou and Zhang [25] proposed the BP-MIP algorithm training a 2-layer Multi-Layer Perceptron with sigmoid activation functions and an objective function computed on the instance with maximum value in each bag. Later, they presented two extensions of their approach [26] by using DD for feature scaling or Principal Component Analysis for feature selection before doing the BP-MIP training. In [27], the same authors proposed a neural network-based method using Radial Basis Functions (RBF), where the centres of the RBFs are determined by a clustering algorithm operating on bags and using the Hausdorff distance.

Recently, deep neural network models have been used for MIL. Wang *et al*. [28] proposed two approaches, mi-Net and MI-Net, performing the pooling within bags respectively at the instance level or at the embedding level. They also employed deep supervision and skip connections to further improve the performance. Ilse *et al*. [29] in their approach rendered this pooling operation trainable using an attention-based mechanism, thus learning also the contribution of each instance within a bag. Finally, Tu *et al*. [30] used a Graph Neural Network (GNN) to model the structural relationship of instances in the bags, and trained this neural network end to end.

We propose to use the MIL formulation for learning neural networks with instance label noise, *i.e*. bags are formed artificially by assuming that the positive bags contain at least one correctly labelled instance, and negative bags contain only negative instances.

Recently, several approaches for training neural network models under label noise have been presented in the literature. For example, Gosh *et al*. [31], [32] showed that the loss function based on the Mean Absolute Error (MAE) is robust to label noise. Zhang *et al*. [33] generalised their approach and proposed a more powerful loss function based on the negative Box-Cox transform [34]. Another approach proposed by Patrini *et al*. [35] is to correct the loss during training based on the estimated label noise. Reed *et al*. [36] proposed to gradually replace the noisy labels of instances with their neural network predictions. And Ren *et al*. [37] re-weighted the individual training instances based on their gradient, but an additional small training set with clean labels is required.

In this paper, we propose a new MIL approach using a neural network model trained with a specific loss function based on the generalised mean function, and we apply our method to two different types of scenarios: first, the standard MIL problem, where the data are organised in bags that are either labelled positive or negative. And second, binary classification problems where labels are available at the instance level but the labels of one class are noisy. In the latter setting, our MIL algorithm is applied on random bags that group instances of the same class.

We propose the following contributions:

- a neural network-based framework for binary classification with (non-symmetric) label noise using a MIL approach,

- a new loss function based on the generalised mean that can be modulated smoothly between the *mean* and the *max* function by varying the exponential parameter p,
- a probabilistic method to dynamically estimate the amount of noise in the training data and to automatically determine the hyper-parameter p accordingly,
- a comprehensive study of noise robustness of the most commonly used loss functions for MIL with neural networks.

In the following, we will first describe the classical MIL problem and how it can be tackled with neural network models in general. Then, we will present our approach that has two main components: a new loss function and an automatic hyper-parameter adaptation scheme based on a dynamic label noise estimation. Finally, we will present experimental results and draw some conclusions.

III. Multiple Instance Learning with Neural Networks

In classical supervised learning for binary classification, the parameters θ of a function $f(\theta, x_i)$ are trained to predict the labels $Y_i \in \{0, 1\}$ for an input vector $x_i \in \mathbb{R}^D$. In the MIL setting, as formulated by Dietterich *et al*. [3], sets of instances $X_k = \{x_{k1}, \dots x_{kN_k}\}$ are grouped into K "bags" and labels Y_k are assigned at the bag level ($k = 1 \dots K$), N_k being the number of instances in bag k. Thus the individual instance labels are not available. Moreover, the negative bags contain only negative instances, whereas a positive bag contains at least one positive instance.

A common approach is *instance-level* MIL, *i.e*. to learn the parameters θ of $f(\theta, x_{ki})$ such that $Y_k = \sigma(f(\theta, x_{ki}))$, where $f(\cdot)$ classifies instances of a bag k and σ is a permutation-invariant aggregation function. Classically, the $\max$ function is chosen for σ, *i.e*. $Y_k = \max_i f(\theta, x_{ki})$, as in the BP-MIP algorithm [25]. However, in gradient-based learning this is not practical as the $\max$ function is not differentiable w.r.t. its arguments leading to a non-convex loss function. Several alternatives have been proposed, most of them modelling a type of "smooth", continuous $\max$ function. Let $O_k = \{o_{ki}\}\,(i = 1 \dots N_k)$ be the outputs of the model for each instance of a bag, that is $o_{ki} = f(\theta, x_{ki})$. Common choices are:

$$\text{Probabilistic:} \quad \sigma(O_k) = 1 - \exp\left(-\sum_{i=1}^{N_k} o_{ki}\right) \tag{1}$$

$$\text{Noisy Or:} \quad \sigma(O_k) = 1 - \prod_{i=1}^{N_k} (1 - o_{ki}) \tag{2}$$

$$\text{Log-Sum-Exp [24]:} \quad \sigma(O_k) = \frac{1}{M} \ln \sum_{i=1}^{N_k} e^{M o_{ki}}, \tag{3}$$

where M is a constant controlling the precision of the $\max$ approximation.

The weights θ of a neural network can then be trained by minimising a loss function computed on all bags X_k using the bag label Y_k:

$$L_\theta = \sum_{k=1}^{K} L_\theta^k(X_k, Y_k)\ , \tag{4}$$

and the instances that compose the bag x_{ki}. For example, the Mean Squared Error (MSE):

$$L_\theta^k(X_k, Y_k) = \frac{1}{2}\left(Y_k - \sigma(f(\theta, x_{ki}))\right)^2\ . \tag{5}$$

Also, the well-known binary cross entropy loss function together with the softmax function is commonly used for MIL.

IV. Proposed approach

As described in the previous section, we propose an instance-level MIL approach that is based on a neural network that estimates instance labels o_{ki} which are then aggregated by the function $\sigma(O_k)$ to predict the bag labels. Our method is independent from the architecture of the neural network model and the loss function. Here, we will use the loss function defined in Eq. 4 and the MSE computed with the target labels Y_k and the outputs o_{ki} from a single scalar $o_{ki} \in (0, 1)$ (*i.e*. one output neuron).

A. Loss function

We propose to use the generalised mean function as a smooth aggregation function:

$$\sigma(O_k) = \left(\frac{1}{N_k}\sum_{i=1}^{N_k} o_{ki}^p\right)^{\frac{1}{p}}\ . \tag{6}$$

This function has several positive properties. Firstly, it is continuous and differentiable. And secondly, the parameter $p \in \mathbb{R}$ modulates the influence of each individual instance in the aggregation. For $p = 1$, σ computes the arithmetic mean of its arguments. For $p \to \infty$, it tends to the $\max$ function. This property is particularly interesting when we have no estimate of the proportion or the number of positive instances in the positive bags, *i.e*. the amount of noise.

During training, instead of setting p to a fixed value, we propose a strategy that starts with $p = 1$ and linearly increases until $p = p_{max}$. In that way, in the beginning, each instance is considered equally within a bag and, at the end, only the most dominant one will be used to compute the output and update the parameters.

B. Noise level adaptation

As mentioned above, the hyper-parameter p can be used to adapt to different amounts of label noise. Our proposed strategy of linearly increasing p until p_{max} effectively tries to extract relevant information from all instances in a bag. However, when p approaches p_{max}, for large p_{max}, only one instance per batch is used for training, and with few training data, there is a risk of overfitting to these instances. Thus, when the amount of label noise is small, a smaller p_{max} is preferable

(and vice-versa) because several examples contribute to the output of a batch and can be used to update the model.

We propose a strategy to dynamically estimate the label noise N in the training data set and set p_{max} accordingly. Let L^* be the true label, L the actual (noisy) label and T the (binary) prediction of the model. We want to estimate the proportion of negative instances in the positive bags, *i.e.*:

$$p(L^*{=}0|L{=}1,T) = 1 - p(L^*{=}1|L{=}1,T) \quad (7)$$

$$= 1 - \frac{p(T,L{=}1|L^*{=}1)p(L^*{=}1)}{p(T,L{=}1)} \quad (8)$$

$$= 1 - \frac{p(T|L{=}1|L^*{=}1)p(L^*{=}1)}{p(T|L{=}1)p(L{=}1)} \quad (9)$$

$$= 1 - \frac{p(T|L{=}1|L^*{=}1)}{p(T|L{=}1)}(1-N) \quad (10)$$

$$= 1 - [\underbrace{p(T{=}1|L^*{=}1,L{=}1)}_{TP} + \underbrace{p(T{=}0|L^*{=}1,L{=}1)}_{FN}]\frac{1-N}{p(T|L{=}1)} \quad (11)$$

where TP and FN are the true positive and false negative rates in a positive bag, respectively, which can be coarsely approximated by:

$$TP \approx p(T{=}1|L{=}1) \quad (12)$$

$$FN \approx p(T{=}1|L^*{=}0,L{=}0)\frac{p(L{=}0)}{p(L{=}1)}\,, \quad (13)$$

assuming that the false negatives are proportional to the false positives according to the proportion of positive and negative instances in the training set.

Setting $p(T|L{=}1) = 1$ and assuming that the noise is equally distributed among the bags, *i.e.* $N \approx p(L^*{=}0|L{=}1,T)$ we formulate a recursive update equation for N:

$$N_t = 1 - [TP + FN]\,(1 - N_{t-1}) \quad (14)$$

In the learning process described above, instead of a fixed p_{max}, we use $N_t p_{max}$ as the maximum exponent in Eq. 6. This approximation is relatively coarse as it is based on a number of loose assumptions. However, our experiments showed that this is enough to guide the dynamic adaptation of the parameter p_{max} in the loss function.

V. Experiments

We performed several sets of experiments to evaluate the proposed neural network MIL approach. In the first experiment we measured the accuracy binary classification on the following benchmark datasets from the UCI Machine Learning Repository[1]:

Name	#instances	attributes
Credit Approval	690	15
Spambase	4601	57
Census Income	48842	14
Bank Marketing	45211	17

[1]https://archive.ics.uci.edu/ml/datasets.php

Further we created a dataset for image classification containing 16 539 greyscale images of faces extracted from the AFLW dataset[2] and resized to 36×36 pixels and 26 950 non-faces images randomly extracted from landscape or other photographs. We performed a 10-fold cross validation, *i.e.* with 10 random partitions where 90% is used for training and 10% held out for testing. From the training part, in each run, 10% is used for validation and early stopping.

To introduce label noise, we randomly switched positive labels into negative ones with probability p_s varying between 10% and 90%, *i.e.* the positive bags will contain a considerable amount of incorrectly labelled instances (up to ∼50% for Credit Approval, 35% for Spambase, 68% for Census Income, 80% for Bank Marketing and up to 56% for AFLW Faces). The labels of the test set are not altered, and the correct classification rate (accuracy) is computed at the instance level using a trivial output threshold of 0.5.

We used a simple Multi-Layer Perceptron (MLP) architecture with 2 hidden layers of 20 and 10 neurons and an output layer of 1 neuron. The second and third layers use a sigmoid activation function. We experimented with different activation functions, *e.g.* ReLU, but we did not notice a significant difference. Also various different architectures have been compared, and when increasing the number of neurons or layers, the model tends to overfit. This may be reduced by an increased regularisation, but, in this study, we preferred to keep the model complexity as simple as possible. For the face image dataset, we used a Convolutional Neural Network (CNN) with 2 convolutional layers of 8 and 16 channels, respectively, each followed by a 2x2 pooling layer, a fully-connected layer of 100 neurons and a fully-connected output layer. Again, all layers except the first use a sigmoid activation function. The network models are trained using the MSE loss function (Eq. 4 and 5) and Stochastic Gradient Descent (SGD). Each experiment is run 10 times and the mean and standard deviation is reported. The batch or bag size is set to 10 for all experiments (except for the *ONLINE* mode). We also perform the dynamic noise estimation in the beginning of the training and set $p_{max} = 500$ in our experiments.

Tables I–V show the accuracy results for the 5 different datasets and different loss and aggregation functions: *ONLINE* is the standard online backpropagation algorithm, *i.e.* batch size is 1, the *BATCH* algorithm updates the model using the gradient cumulated over mini-batches of size 10, *MAX* uses the simple $\max$ function over the model outputs of the instances of a bag, like in BP-MIL [25] and *PROB*, *LSE* and *NOISY-OR* correspond to the probabilistic, log-sum-exp [24] and noisy-or aggregation functions of Eq. 1-3. The last column, named *GMEAN*, corresponds to our proposed approach using the generalised mean loss function. We highlighted in bold face, for each configuration, the maximum mean values that are statistically significant (using hypotheses testing and the two-tailed t-test with p-value 0.05).

[2]https://www.tugraz.at/institute/icg/research/team-bischof/lrs/downloads/aflw/

noise level	**ONLINE**	**BATCH**	**MAX**	**PROB**	**LSE**	**NOISY-OR**	**GMEAN**
0	0.857273 ±0.01	0.861818 ±0.01	0.64972 ±0.02	0.55317 ±0.01	0.658186 ±0.03	0.556908 ±0.01	0.856869 ±0.00
0.2	0.850758 ±0.01	0.857424 ±0.01	0.663982 ±0.03	0.572922 ±0.03	0.667148 ±0.03	0.563271 ±0.02	**0.865629** ±0.01
0.4	0.836344 ±0.02	0.855447 ±0.01	0.684596 ±0.03	0.564794 ±0.02	0.680843 ±0.03	0.555769 ±0.01	0.858182 ±0.01
0.6	0.816939 ±0.03	0.838065 ±0.02	0.705991 ±0.04	0.610831 ±0.05	0.716461 ±0.03	0.570509 ±0.02	0.844367 ±0.01
0.8	0.773147 ±0.04	0.79864 ±0.03	0.723897 ±0.04	0.633283 ±0.06	0.743974 ±0.05	0.630501 ±0.06	**0.819701** ±0.03
0.9	0.720152 ±0.06	0.764184 ±0.05	0.730381 ±0.07	0.659759 ±0.06	0.722292 ±0.06	0.627863 ±0.07	**0.789864** ±0.04

TABLE I
ACCURACY ON THE *Credit Approval* DATASET WITH DIFFERENT LOSS FUNCTIONS AND NOISE LEVELS.

noise level	**ONLINE**	**BATCH**	**MAX**	**PROB**	**LSE**	**NOISY-OR**	**GMEAN**
0	0.922116 ±0.00	0.91515 ±0.01	0.664329 ±0.01	0.577449 ±0.05	0.662674 ±0.01	0.496986 ±0.04	**0.92435** ±0.00
0.2	0.914168 ±0.01	0.913491 ±0.00	0.673056 ±0.01	0.586576 ±0.04	0.674046 ±0.01	0.550452 ±0.05	**0.919783** ±0.00
0.4	0.899447 ±0.01	0.905301 ±0.01	0.671484 ±0.01	0.609203 ±0.03	0.672176 ±0.01	0.566644 ±0.04	**0.912057** ±0.01
0.6	0.867355 ±0.01	0.884877 ±0.01	0.675761 ±0.02	0.627153 ±0.04	0.672806 ±0.01	0.577294 ±0.03	0.891966 ±0.01
0.8	0.816853 ±0.01	0.848576 ±0.01	0.670601 ±0.02	0.659767 ±0.04	0.675696 ±0.02	0.621852 ±0.03	**0.861004** ±0.01
0.9	0.774891 ±0.01	0.81587 ±0.01	0.67142 ±0.03	0.683759 ±0.05	0.676249 ±0.02	0.64239 ±0.03	0.824808 ±0.01

TABLE II
ACCURACY ON THE *Spambase* DATASET WITH DIFFERENT LOSS FUNCTIONS AND NOISE LEVELS.

noise level	**ONLINE**	**BATCH**	**MAX**	**PROB**	**LSE**	**NOISY-OR**	**GMEAN**
0	0.773888 ±0.01	0.778351 ±0.02	0.796122 ±0.00	0.771677 ±0.00	**0.799152** ±0.00	0.765514 ±0.00	0.766231 ±0.01
0.2	0.771697 ±0.01	0.776283 ±0.00	**0.811764** ±0.00	0.787933 ±0.00	0.809655 ±0.00	0.77356 ±0.00	0.755359 ±0.01
0.4	0.765616 ±0.01	0.76752 ±0.00	0.813525 ±0.00	0.794136 ±0.00	0.814139 ±0.00	0.77702 ±0.00	0.764675 ±0.03
0.6	0.749893 ±0.00	0.74373 ±0.01	0.81764 ±0.00	0.79991 ±0.00	0.819892 ±0.00	0.781688 ±0.00	0.820588 ±0.00
0.8	0.718854 ±0.01	0.723256 ±0.02	0.816023 ±0.00	0.790533 ±0.01	0.816514 ±0.00	0.775382 ±0.01	**0.818582** ±0.00
0.9	0.68106 ±0.03	0.685687 ±0.02	0.812972 ±0.00	0.769302 ±0.00	0.811478 ±0.00	0.765821 ±0.00	**0.814262** ±0.00

TABLE III
ACCURACY ON THE *Census Income* DATASET WITH DIFFERENT LOSS FUNCTIONS AND NOISE LEVELS.

noise level	**ONLINE**	**BATCH**	**MAX**	**PROB**	**LSE**	**NOISY-OR**	**GMEAN**
0	0.808448 ±0.01	0.796924 ±0.01	**0.899089** ±0.00	0.895359 ±0.00	0.89875 ±0.00	0.889782 ±0.00	0.874549 ±0.01
0.2	0.775211 ±0.01	0.773402 ±0.01	0.897317 ±0.00	0.890616 ±0.00	0.89739 ±0.00	0.89654 ±0.00	0.826638 ±0.01
0.4	0.722206 ±0.02	0.741464 ±0.02	0.891781 ±0.00	0.885945 ±0.01	0.89251 ±0.00	**0.894735** ±0.00	0.892038 ±0.00
0.6	0.683772 ±0.02	0.706092 ±0.03	0.886085 ±0.00	0.888042 ±0.00	0.884976 ±0.00	0.887346 ±0.00	**0.890502** ±0.00
0.8	0.641384 ±0.02	0.672054 ±0.02	0.879304 ±0.01	**0.887346** ±0.00	0.879943 ±0.00	**0.887346** ±0.00	0.884904 ±0.00
0.9	0.602134 ±0.02	0.645634 ±0.01	0.868477 ±0.01	**0.888827** ±0.00	0.880451 ±0.01	0.887346 ±0.00	0.879337 ±0.00

TABLE IV
ACCURACY ON THE *Bank Marketing* DATASET WITH DIFFERENT LOSS FUNCTIONS AND NOISE LEVELS.

noise level	**ONLINE**	**BATCH**	**MAX**	**PROB**	**LSE**	**NOISY-OR**	**GMEAN**
0	0.976132 ±0.001	0.984847 ±0.001	0.984594 ±0.000	0.982409 ±0.001	0.984916 ±0.005	0.984295 ±0.002	0.963508 ±0.005
0.2	0.946883 ±0.001	0.966474 ±0.002	0.968889 ±0.000	0.959024 ±0.003	0.969532 ±0.009	0.970705 ±0.010	0.965439 ±0.009
0.4	0.935432 ±0.002	0.959898 ±0.001	0.959047 ±0.003	0.916691 ±0.004	0.965049 ±0.015	0.95999 ±0.004	0.958783 ±0.013
0.6	0.940629 ±0.003	0.948401 ±0.005	0.945182 ±0.006	0.858125 ±0.007	0.950608 ±0.003	0.947596 ±0.002	**0.958771** ±0.010
0.8	0.886041 ±0.004	0.914001 ±0.002	0.926004 ±0.002	0.67842 ±0.002	0.923337 ±0.009	0.928602 ±0.009	**0.95293** ±0.011
0.9	0.865851 ±0.005	0.858011 ±0.004	0.883741 ±0.010	0.612039 ±0.010	0.88266 ±0.017	0.881465 ±0.099	**0.939226** ±0.026

TABLE V
ACCURACY ON THE *AFLW Faces* DATASET WITH DIFFERENT LOSS FUNCTIONS AND NOISE LEVELS.

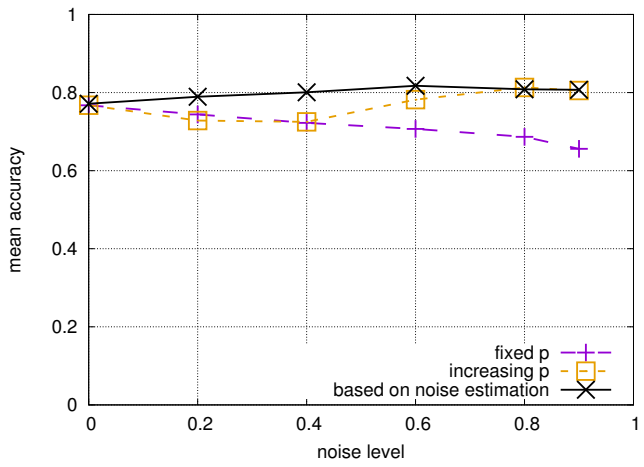

Fig. 1. Mean accuracy on "Census Income" of our proposed approach with varying noise for 1) fixed p, 2) p increasing from 1 to p_{max} and 3) p increasing from 1 to $N_t \cdot p_{max}$ based on the estimated label noise N_t.

It can noticed that for low noise levels most algorithms give similar performance, except *MAX*, *PROB*, *LSE* and *NOISY-OR* for the smaller datasets (Credit Approval, Spambase). This is probably because they concentrate too much on a single, or on very few instances in a bag and do not exploit the information from all relevant instances for training. For larger dataset, there is no clear "winner" among the baseline methods. However, the proposed method (*GMEAN*) outperforms them especially for larger amounts of label noise. Also for smaller datasets our method is superior to (or on par with) the baseline for almost all levels of noise. These results suggest that the adaptive generalised mean function is able to effectively use the information and features from bags with variable amounts of correctly labelled instances. For large datasets, this property may be less important if there is enough redundancy in the training data, *i.e.* other approaches like the standard *BATCH* training may be robust to noise where many instances are very similar but have different labels.

To show the effectiveness of our approach to dynamically estimate the label noise and set the hyper-parameter p in Eq. 6 accordingly, we measured the mean accuracy with three different strategies for varying noise levels: 1) $p = p_{max}$, 2) $p = 1 \dots p_{max}$ increasing linearly during training and 3) $p = 1 \dots N_t \cdot p_{max}$, where N_t is the estimated noise level computed using Eq. 14. Figure 1 shows the results. The approach using the dynamic, noise-dependant p obtains the highest accuracy for all levels of noise, and thus adapts dynamically to different scenarios and datasets. We observed the same behaviour on the other datasets.

Finally, we also evaluated our approach on the standard MIL benchmark datasets: MUSK1, MUSK2, Fox, Tiger and Elephant. Here, labels are only available for bags. Thus, according to the standard evaluation protocol, the classification accuracy is computed at the bag level, and no additional artificial label noise is added. We used an MLP architecture with only 1 hidden layer of 150 neurons and an output layer of one neuron and sigmoid activation functions, as before. A bag is considered positive if the output defined by Eq. 6, *i.e.* the generalised mean over the bag with $p = p_{max}$, is above 0.5, and negative otherwise.

We compared our method to the state of the art in MIL: of of the classical methods: EM-DD [6], the Random Forest-based methods: MIForest [19] and MIOForest [20], the kernel-based methods: MI-Kernel [9], mi-SVM and MI-SVM [8], mi-Graph [14], miVLAD and miFV [15], and the neural network-based methods: MI-Net [28] using a Deep Neural Network (DNN) with some specific tricks, like deep supervision, Attention-MIL [29] based on a DNN with attention mechanism and the GNN proposed by [30]. Table VI shows the mean accuracies and standard errors for the different methods and datasets. Our method compares favorable to the state of the art, although it is instance-based and does not model the relationship between instances in a bag as GNN [30], for example. In fact, it outperforms most of the kernel-based methods except on MUSK2. Also it compares favorable or outperforms other neural network-based approaches. Only the GNN-based method shows slightly better results, but, as mentioned above, it is bag-level classifier and thus cannot predict instance labels. Furthermore, our neural network architecture is extremely light and shallow compared to the other DNNs. For that reason these methods require strong regularisation (*e.g.* with drop-out) and other tricks, like deep supervision and skip connections. Probably, our results can be further improved by optimising the neural architecture and training strategy.

VI. Conclusion

We presented a new approach for training neural networks under label noise based on a Multiple Instance Learning framework. We proposed a new loss function defined by the generalised mean function on the bag instances which is particularly suitable for effectively learning with strong label noise. We further introduced a method to estimate during training the label noise and automatically set the hyper-parameter p of our function. In our experiments we compared our approach to the most common loss functions for MIL with neural networks and showed that our method outperforms them in most cases, especially with few training data and with high label noise. Finally, we also compared our algorithm to state-of-the-art methods in MIL and standard benchmarks, and we obtained comparable results with a much less complex model. In future work, we would like to study the use of more complex (strongly regularised) DNN architectures and extend our approach for semi-supervised learning.

References

[1] R. D. and P. Manikandan, "Imbalanced dataset classification and solutions: a review," *International Journal of Computing and Business Research (IJCBR)*, vol. 5, no. 4, 2014.

[2] S. Cateni, V. Colla, and M. Vannucci, "A method for resampling imbalanced datasets in binary classification tasks for real-world problems," *Neurocomputing*, vol. 135, pp. 32–41, 2014.

[3] R. G. Dietterich, R. H. Lathrop, and T. Lozano-Prez, "Solving the multiple instance problem with axis-parallel rectangles," *Artifical Intelligence*, vol. 89, no. 1, pp. 31–71, 1997.

Algorithms	MUSK1	MUSK2	FOX	TIGER	ELEPHANT
EM-DD [6]	0.849±0.044	0.869±0.048	0.609±0.045	0.730±0.043	0.771±0.043
MIForest [19]	0.85±N/A	0.82±N/A	0.64±N/A	0.82±N/A	0.84±N/A
MIOForest [20]	0.89±N/A	0.87±N/A	0.68±N/A	0.83±N/A	0.86±N/A
MI-Kernel [9]	0.880±0.031	0.893±0.015	0.603±0.028	0.842±0.010	0.843±0.016
mi-SVM [8]	0.74±N/A	0.836±N/A	0.582±N/A	0.784±N/A	0.822±N/A
MI-SVM [8]	0.779±N/A	0.843±N/A	0.578±N/A	0.840±N/A	0.843±N/A
mi-Graph [14]	0.889±0.033	0.903±0.039	0.620±0.044	0.860±0.037	0.869±0.035
miVLAD [15]	0.871±0.043	0.871±0.042	0.620±0.044	0.811±0.039	0.850±0.036
miFV [15]	0.909±0.040	0.884±0.042	0.621±0.049	0.813±0.037	0.852± 0.036
MI-Net [28]	0.887±0.041	0.859±0.046	0.622±0.038	0.830±0.032	0.862±0.034
Attention-MIL [29]	0.892±0.040	0.858±0.048	0.615±0.043	0.839±0.022	0.868±0.022
GNN [30]	0.917±0.048	0.892±0.011	0.679±0.007	0.876±0.015	0.903±0.010
Ours	0.916±0.033	0.841±0.048	0.635±0.026	0.860±0.037	0.858±0.022

TABLE VI
COMPARISON WITH THE STATE OF THE ART ON STANDARD MIL DATASETS (MEAN ACCURACY AND STANDARD ERROR OF THE MEAN).

[4] P. Auer, "On learning from multi-instance examples: empirical evaluation of a theoretical approach," in *Proceedings of the International Conference on Machine Learning (ICML)*, 1997, pp. 21–29.

[5] O. Maron and T. Lozano-Pérez, "A framework for multiple-instance learning," in *Advances in Neural Information Processing Systems (NIPS)*, 1998, pp. 570–576.

[6] Q. Zhang and S. A. Goldman, "EM-DD: an improved multi-instance learning technique," in *Advances in Neural Information Processing Systems (NIPS)*, 2002, pp. 1073–1080.

[7] J. Wang and J.-D. Zucker, "Solving the multiple-instance problem: a lazy learning approach," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2000, pp. 1119–1125.

[8] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support vector machines for multiple-instance learning," in *Advances in Neural Information Processing Systems (NIPS)*, 2003, pp. 561–568.

[9] T. Gärtner, P. Flach, A. Kowalczyk, and A. J. Smola, "Multi-instance kernels," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2002, pp. 179–186.

[10] Y. Chen and J. Z. Wang, "Image categorization by learning and reasoning with regions," *J. of Machine Learning Res.*, vol. 5, pp. 913–939, 2004.

[11] Y. Chen, J. Bi, and J. Z. Wang, "Miles: Multiple-instance learning via embedded instance selection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 1931–1947, 2006.

[12] P.-M. Cheung and J. T. Kwok, "A regularization framework for multiple-instance learning," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2006.

[13] Z.-H. Zhou and J.-M. Xu, "On the relation between multi-instance learning and semi-supervised learning," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2007.

[14] Z.-H. Zhou, Y.-Y. Sun, and Y.-F. Li, "Multi-instance learning by treating instances as non-iid samples," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2009, pp. 1249–1256.

[15] X.-S. Wei, J. Wu, and Z.-H. Zhou, "Scalable algorithms for multi-instance learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 4, pp. 975–987, 2017.

[16] G. Ruffo, "Learning single and multiple instance decision trees for computer security applications," Ph.D. dissertation, Department of Computer Science, University of Turin, Torino, Italy, 2000.

[17] Y. Chevaleyre and J.-D. Zucker, "Solving multiple-instance and multiple-part learning problems with decision trees and decision rules. application to the mutagenesis problem." in *In Proceedings of the 14th Biennial Conference of the Canadian Society for Computational Studies of Intelligence*, 2001, pp. 204–214.

[18] H. Blockeel, D. Page, and A. Srinivasan, "Multi-instance tree learning," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2005, pp. 57–64.

[19] C. Leistner, A. Saffari, and H. Bischof, "MIForests: multiple-instance learning with randomized trees," in *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 29–42.

[20] C. Straehle, M. Kandemir, U. Köthe, and F. Hamprecht, "Multiple instance learning with response-optimized random forests," in *Proc. of the International Conference on International Conference on Pattern Recognition (ICPR)*, 2014.

[21] N. Weidmann, E. Frank, and B. Pfahringer, "A two-level learning method for generalized multi-instance problem," in *Proceedings of the European Conference on Machine Learning (ECML)*, 2003, pp. 468–479.

[22] T. Komárek and P. Somol, "Multiple instance learning with bag-level randomized trees," in *Proceedings of the European Conference on Machine Learning (ECML)*, 2018.

[23] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, no. 1, pp. 3–42, 2006.

[24] J. Ramon and L. De Raedt, "Multi instance neural networks," in *Proceedings of the ICML 2000 workshop on attribute-value and relational learning*, 2000, pp. 53–60.

[25] Z.-H. Zhou and M.-L. Zhang, "Neural networks for multi-instance learning," AI Lab, Computer Science & Technology Department, Nanjing University, Nanjing, China, Tech. Rep., aug 2002.

[26] M.-L. Zhang and Z.-H. Zhou, "Improve multi-instance neural networks through feature selection," *Neural Processing Letters*, vol. 19, no. 1, pp. 1–10, 2004.

[27] ——, "Adapting RBF neural networks to multi-instance learning," *Neural Processing Letters*, vol. 23, no. 1, pp. 1–26, 2006.

[28] X. Wang, Y. Yan, P. Tang, X. Bai, and W. Liu, "Revisiting multiple instance neural networks," *Pattern Recognition*, vol. 74, pp. 15–24, 2018.

[29] M. Ilse, J. Tomczak, and M. Welling, "Attention-based deep multiple instance learning," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2018, pp. 2127–2136.

[30] M. T. Tu, J. Huang, X. He, and B. Zhou, "Multiple instance learning with graph neural networks," in *Proceedings of the ICML 2019 Workshop on Learning and Reasoning with Graph-Structured Data*, 2019.

[31] A. Ghosh, N. Manwani, and P. S. Sastry, "Making risk minimization tolerant to label noise," *Neurocomputing*, vol. 160, pp. 93–107, 2015.

[32] A. Ghosh, H. Kumar, and P. S. Sastry, "Robust loss functions under label noise for deep neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017, pp. 1919–1925.

[33] Z. Zhang and M. R. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Advances in Neural Information Processing Systems (NIPS)*, 2018.

[34] G. E. P. Box and D. R. Cox, "An analysis of transformations," *Journal of the Royal Statistical Society. Series B*, pp. 211–252, 1964.

[35] G. Patrini, A. Rozza, A. K. Menon, R. Nock, and L. Qu, "Making deep neural networks robust to label noise: A loss correction approach," in *Proc. of the International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2233–2241.

[36] S. E. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich, "Training deep neural networks on noisy labels with bootstrapping," in *3rd International Conference on Learning Representations (ICLR)*, 2015.

[37] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2018, pp. 4331–4340.