A Network Architecture for Multi-Multi-Instance Learning

Alessandro Tibo[1], Paolo Frasconi[1], and Manfred Jaeger[2]

[1] Department of Information Engineering, University of Florence, Via di Santa Marta 3, I-50139 Firenze, Italy, {alessandro.tibo, paolo.frasconi}@unifi.it
[2] Department of Computer Science, Aalborg University, Denmark, jaeger@cs.aau.dk

Abstract. We study an extension of the multi-instance learning problem where examples are organized as nested bags of instances (e.g., a document could be represented as a bag of sentences, which in turn are bags of words). This framework can be useful in various scenarios, such as graph classification, image classification and translation-invariant pooling in convolutional neural network. In order to learn multi-multi instance data, we introduce a special neural network layer, called bag-layer, whose units aggregate sets of inputs of arbitrary size. We prove that the associated class of functions contains all Boolean functions over sets of sets of instances. We present empirical results on semi-synthetic data showing that such class of functions can be actually learned from data. We also present experiments on citation graphs datasets where our model obtains competitive results.

Keywords: Classification, Deep learning, Neural networks, Relational and structured data

1 Introduction

In the multi-instance (MI) setting, data is organized in bags of instances rather than single instances. Only labels of bags are observed, while labels of individual instances are assumed to be unknown. Since the original introduction of this setting in the seminal paper of Dietterich [4], several algorithmic solutions have been proposed in the literature, including diverse density [13] and its expectation-maximization extension [23], and adaptations of various learning algorithms such as kNN [21], neural networks [17,24], and support vector machines [1]. Most methods are based on the classic assumptions that instances are either positive or negative, while bags are positive if and only if they contain at least one positive instance. Several other kinds of underlying assumptions have been formulated (see [6] for a review). In this paper, we extend the MI learning setting by considering data consisting of several nested levels of bags of instances. We call this framework multi-multi-instance (MMI) learning, referring specifically to the case of bags-of-bags (the generalization to deeper levels of nesting is however immediate). We focus on the setting where the task is to classify whole bags-of-bags (a setting where instances or sub-bags are to be classified is also conceivable but not addressed in this paper). In our approach, we relax the classic MI assumption of binary instance labels, allowing categorical labels lying in an unspecified alphabet. For MMI learning we propose a method based on neural networks with a special layer called bag-layer. Unlike previous neural network approaches to MI learning, where predicted instance labels are aggregated by the maximum operator, bag-layers aggregate internal representations of instances (or bags of instances) and can be naturally intermixed with other layers commonly used in deep learning. Bag-layers can be in fact interpreted as a generalization of pooling layers commonly used in convolutional neural networks.

The paper is organized as follows. In Section 2 we formally introduce the MMI learning setting. In Section 3 we present the proposed neural network architecture. In 4 we discuss some related works. In Section 5 we offer a theoretical analysis of the expressiveness of the proposed bag-layer architecture. In Section 6 we report an experimental evaluation of our method on some semi-synthetic datasets and on real citation network data. Finally, in Section 7 we summarize our findings and discuss possible directions of future research. The code we used for the experiments can be downloaded from `https://github.com/alessandro-t/mmi`.

2 The Multi-Multi-Instance Learning Setting

For the sake of simplicity, we introduce our setting with data consisting of bags-of-bags of instances. The extension to deeper levels of nesting (i.e., multi^K-instance learning) is immediate. We will also informally use the expression "bag-of-bags" to describe sets with two or more levels of nesting.

Our setting is supervised and the dataset consists of pairs $\{(X^{(i)}, y^{(i)}), i = 1, \dots, n\}$ where $y^{(i)}$ is a discrete category label and $X^{(i)}$ is a set of sets: $X^{(i)} = \{S_j^{(i)}, j = 1, \dots, |X^{(i)}|\}$ where $|X|$ denotes the cardinality of X. In turn, $S_j^{(i)} = \{x_{j,\ell}^{(i)}, \ell = 1, \dots, |S_j^{(i)}|\}$ where $x_{j,\ell}^{(i)} \in \mathcal{X}$ is an *instance* and $\mathcal{X}$ the instance space. We call $X^{(i)}$ *top-bags* and $S_j^{(i)}$ *sub-bags*. We assume that examples are uniformly and independently drawn from a distribution $p(X, y) = p(y|X)p(X)$. Since we take a discriminative approach to supervised learning, we do not make any specific assumptions on $p(X)$. We further assume that *latent* labels are attached to both instances and sub-bags (i.e., only labels attached to top-bags are observed). To simplify notation, we drop the example index (superscript i) when not necessary. We further assume that the label of instance $x_{j,\ell}$, denoted $y_{j,\ell}$, is drawn from a conditional distribution $p(y_{j,\ell}|x_{j,\ell})$, that the label of sub-bag S_j, denoted y_j, is drawn from a conditional distribution $p(y_j|y_{j,1}, \dots, y_{j,|S_j|})$, and that the label of top-bag X, denoted y, is drawn from a conditional distribution $p(y|y_1, \dots, y_{|X|})$.

In most cases, both top-bags and sub-bags will actually be sets, i.e., two instances $x_{j,\ell}, x_{j,\ell'}$ contained in a sub-bag S_j will typically be distinct objects, and similarly for the sub-bags contained in a top-bag. However, when considering only the labels of the instances, we obtain proper bags (multi-sets)

$\{y_{j,1}, \ldots, y_{j,|S_j|}\}$. Our bag-of-bag terminology is motivated by the fact that we view the data mostly as bags of latent labels, rather than sets of raw data points. By a slight abuse of notation, we also use $\{\}$ to denote multisets. In a context where we speak about multisets, then, for instance, $\{0, 0, 1\} \neq \{0, 1\}$.

Example 1. In this example we consider bags-of-bags of handwritten digits (as in the MNIST dataset). Each instance (a digit) has attached its own category in $\{0, \ldots, 9\}$, which is not observed. We consider binary sub-bag and top-bag labels. Only the top-bag labels are observed. In particular, a sub-bag is positive if it contains an instance of category 7 and does not contain an instance of category 3. A top-bag is positive if it contains at least one positive sub-bag. Figure 1 shows a positive top-bag and a negative top-bag.

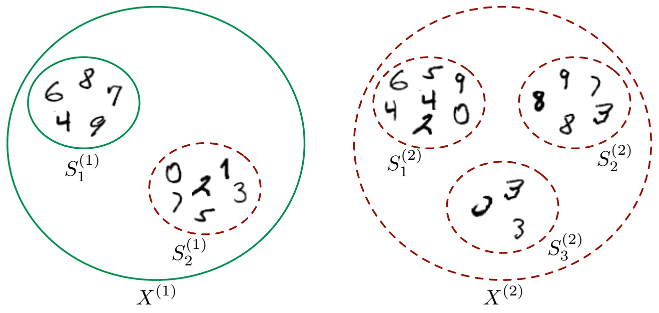

Fig. 1: Two top-bags: on the left a positive top-bag $X^{(1)}$, while on the right a negative top-bag $X^{(2)}$. Solid green lines represent positive bags while dashed red lines represent negative bags.

3 Model

We model the conditional distribution $p(y|X)$ with a neural network architecture that handles sets of sets of variable sizes by aggregating intermediate internal representations. For this purpose, we introduce a new layer called *bag-layer*. A bag-layer takes as input a bag of vectors $\{\phi_1, \ldots, \phi_n\}$, $\phi_i \in \mathbb{R}^m$ and outputs a $k-$dimensional representation of the bag computed as follows:

$$g(\{\phi_1, \ldots, \phi_n\}; w, b) = \Xi_{i=1}^{n} \alpha\left(w\phi_i + b\right) \tag{1}$$

where Ξ is component-wise aggregation operator (such as max or average), $w \in \mathbb{R}^{k\times m}$ is the weight matrix, $b \in \mathbb{R}^k$ is the bias, and α is any component-wise activation function (such as ReLU, tanh, or linear). Both w and b are tunable parameters. Note that equation 1 works with bags of arbitrary cardinality. A bag-layer is illustrated in Figure 2. Networks with a single bag-layer can process

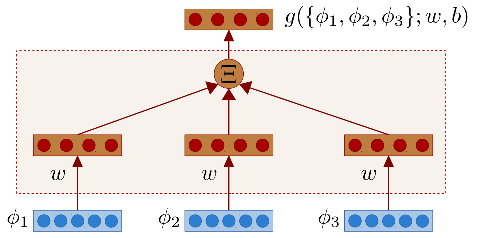

Fig. 2: A bag-layer receiving a bag of cardinality $n = 3$. In this example $k = 4$ and $m = 5$.

sets of instances (as in standard multi-instance learning). In order to work in the multi-multi instance setting, two bag-layers are required. The bottom bag-layer aggregates over internal representations of instances, and the top sub-layer aggregates over internal representations of sub-bags, yielding a representation for the entire top-bag. In this case, the representation of each sub-bag $S_j = \{x_{j,1}, \ldots, x_{j,|S_j|}\}$ would be obtained as

$$\phi_j = g(x_{j,1}, \ldots, x_{j,|S_j|}; w_s, b_s) \quad j = 1, \ldots, |X| \tag{2}$$

and the representation of a top-bag $X = \{S_1, \ldots, S_{|X|}\}$ would be obtained as

$$\phi = g(\phi_1, \ldots, \phi_{|X|}; w_t, b_t) \tag{3}$$

where (w_s, b_s) and (w_t, b_t) denote the parameters used to construct sub-bag and top-bag representations.

Note that nothing prevents us from intermixing bag-layers with standard neural network layers, thereby forming networks of arbitrary depth. In this case, each $x_{j,\ell}$ in Eq. (2) would be simply replaced by the last layer activation of a deep network taking $x_{j,\ell}$ as input. Denoting by θ_s the parameters of such network and by $N_s(x; \theta_s)$ its last layer activation when fed with instance x, Eq. (2) becomes

$$\phi_j = g(N_s(x_{j,1}; \theta_s), \ldots, N(x_{j,|S_j|}; \theta_s); w_s, b_s) \quad j = 1, \ldots, |X|. \tag{4}$$

Similarly, we may use a network N_t, with parameters θ_t, to transform sub-bag representations. As a result, Eq. (3) becomes

$$\phi = g(N_t(\phi_1; \theta_t), \ldots, N_t(\phi_{|X|}; \theta_t); w_t, b_t). \tag{5}$$

Of course the top-bag representation can be itself further processed by other layers. An example of the overall architecture is shown in Figure 3.

4 Related Works

4.1 Multi-instance neural networks

The first algorithm for MI learning was based on axis-parallel rectangles [4]. Shortly after, Ramon & De Raedt [17] proposed a neural network solution

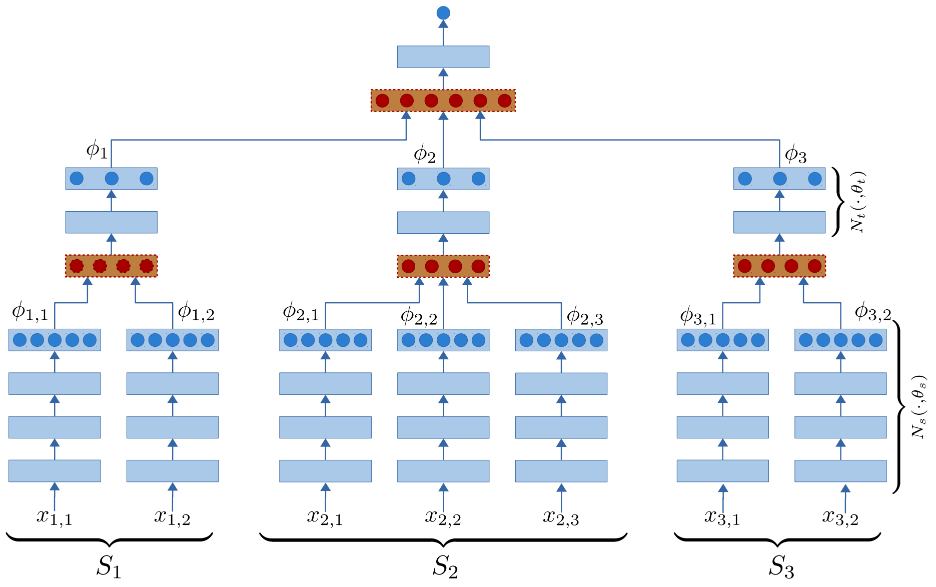

Fig. 3: Network for multi-multi instance learning applied to the bag-of-bags $\{\{x_{1,1}, x_{1,2}\}, \{x_{2,1}, x_{2,2}, x_{2,2}\}, \{x_{3,1}, x_{3,2}\}\}$. Bag-layers are depicted in red with dashed borders. Blue boxes are standard (e.g., dense) neural network layers. Note that parameters θ_s in each of the seven bottom vertical columns are shared, as well parameters θ_t in the middle three columns.

where each instance x_j in a bag $X = \{x_1, \dots, x_{|X|}\}$ is first processed by a replica of a neural network f with weights w. In this way, a bag of output values $\{f(x_1; w), \dots, f(x_{|X|}; w)\}$ computed for each bag of instances. These values are then aggregated by a smooth version of the max function:

$$F(X) = \frac{1}{M} \log\left(\sum_j e^{Mf(x_j;w)}\right)$$

where M is a constant controlling the smoothness of the aggregation. For large enough M, the exact maximum is computed. Note that a single bag-layer (as defined in Section 3) can be easily used to learn in the MI setting. Still, a major difference compared to the work of [17] is that bag-layers perform aggregation at the representation level rather than at the output level. In this way, more layers can be added on the top of the aggregated representation, allowing for more expressiveness. In the classic MI setting (where a bag is positive iff at least one instance is positive) this additional expressiveness is not required. However, it allows to work in slightly more complicated MI settings. For example, suppose each instance has a latent variable $y_j \in 0, 1, 2$, and suppose that a bag is positive iff it contains at least one instance with label 0 and no instance with label 2. In this case, a bag-layer with two units can distinguish positive and negative bags, provided that instance representations can separate instances belonging to the classes $0, 1$ and 2. The network proposed in [17] would not be able to separate positive from negative bags. Indeed, as proved in Section 5, networks with bag-layers can represent any Boolean function over sets of instances.

4.2 Convolutional neural networks

Convolutional neural networks (CNN) [7,12] are the state-of-the-art method for image classification (see, e.g., [20]). It is easy to see that the representation computed by one convolutional layer followed by max-pooling can be emulated with one bag-layer by just creating bags of adjacent image patches. The representation size k corresponds to the number of convolutional filters. The major difference is that the outputs of a convolutional layer are spatially ordered, whereas a bag-layer outputs a set of vectors (without any ordering). This difference may become significant when two or more layers are sequentially stacked. Figure 4

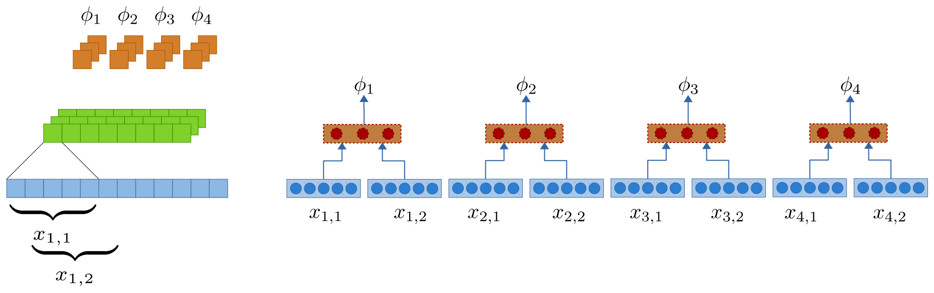

Fig. 4: One convolutional layer with subsampling (left) and the corresponding bag-layer (right). Note that the convolutional layer outputs $[\phi_1, \phi_2, \phi_3, \phi_4]$ whereas the bag-layer outputs $\{\phi_1, \phi_2, \phi_3, \phi_4\}$.

illustrates the relationship between a convolutional layer and a bag-layer, for simplicity assuming a one-dimensional signal (i.e., a sequence). When applied to signals, a bag-layer essentially correspond to a disordered convolutional layer and its output needs further aggregation before it can be fed into a classifier. The simplest option would be to stack one additional bag-layer before the classification layer. Interestingly, a network of this kind would be able to detect the presence of a short subsequence regardless of its position within the whole sequence, achieving invariance to arbitrarily large translations (an experiment related to translational invariance is presented in Section 6.2).

We finally note that it is possible to emulate a CNN with two layers by properly defining the structure of bags-of-bags. For example, a second layer with filter size 3 on the top of the CNN shown in Figure 4 could be emulated with two bag-layers fed by the bag-of-bags

$$\{\{\{x_{1,1}, x_{1,2}\}, \{x_{2,1}, x_{2,2}\}, \{x_{3,1}, x_{3,2}\}\}, \{\{x_{2,1}, x_{2,2}\}, \{x_{3,1}, x_{3,2}\}, \{x_{4,1}, x_{4,2}\}\}\}.$$

A bag-layer, however, is not limited to pooling adjacent elements in a feature map. One could for example segment the image first (e.g., using a hierarchical strategy [2]) and then create bags-of-bags by following the segmented regions.

The convolutional approach has been also recently employed for learning with graph data. For example the technique proposed in [15] casts graphs into a sequential format suitable for representation learning via CNNs. In [5], CNNs are applied to molecular fingerprint vectors. In [3] a diffusion process across general graph structures generalizes the CNN strategy of scanning a regular grid of pixels or voxels. Finally, [11] propose a neural network model based on spectral graph convolutions. In some settings, the architecture presented in this paper can solve graph learning problems where it is natural to organize graph data as bags-of-bags. Indeed, we present in Section 6.3 an application of MMI learning showing that our approach is suitable for citation networks.

4.3 Nested SRL Models

In Statistical Relational Learning (SRL) a great number of approaches have been proposed for constructing probabilistic models for relational data. Relational data has an inherent bag-of-bag structure: each object o in a relational domain can be interpreted as a bag whose elements are all the other objects linked to o via a specific relation. These linked objects, in turn, also are bags containing the objects linked via some relation. A key component of SRL models are the tools employed for aggregating (or combining) information from the bag of linked objects. In many types of SRL models, such an aggregation only is defined for a single level. However, a few proposals have included models for nested combination [9,14]. Like most SRL approaches, these models employ concepts from first-order predicate logic for syntax and semantics, and [9] contains an expressivity result similar in spirit to the one we present in the following section 5.

A key difference between SRL models with nested combination constructs and our MMI network models is that the former build models based on rules for conditional dependencies which are expressed in first-order logic and typically only contain a very small number of numerical parameters (such as a single parameter quantifying a noisy-or combination function for modeling multiple causal influences). MMI network models, in contrast, make use of the high-dimensional parameter spaces of (deep) neural network architectures. Roughly speaking, MMI network models combine the flexibility of SRL models to recursively aggregate over sets of arbitrary cardinalities with the power derived from high-dimensional parameterisations of neural networks.

5 Model expressiveness

In this section, we focus on a restricted (noiseless) version of the MMI setting described in Section 2 where labels are deterministically assigned and no form of counting is involved. We show that under these assumptions, the model of

Section 3 has enough expressivity to represent the solution to the MMI learning problem. Our approach relies on classic universal interpolation results for neural networks [8]. Note that existing results hold for vector data, and this section shows that they can be leveraged to bag-of-bag data when using the model of Section 3.

Definition 1. *We say that data is generated under the restricted MMI setting if the following conditions hold true:*

1. *instance labels are generated by an unknown function $\hat{f} : \mathcal{X} \mapsto C = \{c_1, \dots, c_M\}$, i.e., $y_{j,\ell} = \hat{f}(x_{j,\ell})$, for $j = 1, \dots, |X|$, $\ell = 1, \dots |S_j|$;*
2. *sub-bag labels are generated by an unknown function $\hat{g} : \mathcal{M}(C) \mapsto D$, i.e., $y_j = \hat{g}(\{y_{j,1}, \dots, y_{j,|S_j|}\})$, where $\mathcal{M}(C)$ is the set of all multisets of C;*
3. *the top-bag label is generated by an unknown function $\hat{h} : \mathcal{M}(D) \mapsto \{0, 1\}$, i.e., $y = \hat{h}(\{y_1, \dots, y_{|X|}\})$.*

When data is generated in the restricted setting, then the label of a bag-of-bags

$$X = \{\{x_{1,1}, \dots, x_{1,S_1}\}, \dots, \{x_{|X|,1}, \dots, x_{|X|,|S_{|X|}|}\}\},$$

would be produced as

$$y = \hat{h}\left(\left\{\hat{g}\left(\{\hat{f}(x_{1,1}), \dots, \hat{f}(x_{1,|S_1|})\}\right), \dots, \hat{g}\left(\{\hat{f}(x_{|X|,1}), \dots, \hat{f}(x_{|X|,|S_{|X|}|})\}\right)\right\}\right)$$

Note that the classic MI learning formulation [13] is recovered when examples are sub-bags, $C = \{0, 1\}$, and $\hat{g}(\{y_1, \dots, y_{|S|}\}) = \bigvee_{\ell=1}^{|S|} y_\ell$. Other generalized MI learning formulations [6,18,22] can be similarly captured in this restricted setting.

For a multiset s let $\mathrm{set}(s)$ denote the set of elements occurring in s. E.g. $\mathrm{set}(\{0, 0, 1\}) = \{0, 1\}$.

Definition 2. *We say that data is generated under the non-counting restricted MMI setting if, in addition to the conditions of Definition 1, both $\hat{g}(s)$ and $\hat{h}(s)$ only depend on* $\mathrm{set}(s)$.

The following result indicates that a network containing a bag-layer with max aggregation is sufficient to compute the functions that label both sub-bags and top-bags.

Lemma 1. *Let $C = \{c_1, \dots, c_M\}, D = \{d_1, \dots, d_L\}$ be sets of labels, and let $\hat{g} : \mathcal{M}(C) \mapsto D$ be a labeling function for which $\hat{g}(s) = \hat{g}(s')$ whenever $set(s) = set(s')$. Then there exist a network with one bag-layer that computes $\hat{g}$.*

Proof. We construct a network N where first a bag-layer maps the multiset input s to a bit-vector representation of $set(s)$, on top of which we can then compute $\hat{g}(s)$ using a standard architecture for Boolean functions.

In detail, N is constructed as follows: the input $s = \{y_1, \dots, y_{|s|}\}$ is encoded by $|s|$ M-dimensional vectors ϕ_i containing the one-hot representations of the y_i. We construct a bag-layer with $k = m = M$, w is the $M \times M$ identity matrix, b is zero, α is the identity function, and Ξ is *max*. The output of the bag-layer then is an M-dimensional vector ψ whose i'th component is the indicator function $\mathbb{I}[c_i \in s]$.

For each $j \in \{1, \dots, L\}$ we can write the indicator function $\mathbb{I}[\hat{g}(set(s)) = d_j]$ as a Boolean function of the indicator functions $\mathbb{I}[c_i \in s]$. Using standard universal approximation results (see, e.g., [8], Corollary 2.5) we can construct a network that on input ψ computes $\mathbb{I}[\hat{g}(set(s)) = d_j]$. L such networks in parallel then produce an L-dimensional output vector containing the one-hot representation of $\hat{g}(s)$.

Theorem 1. *Given a dataset of examples generated under the non-counting restricted MMI setting, there exist a network with two bag-layers that can correctly label all examples in the dataset.*

Proof. We first note that the universal interpolation result of [8] can be applied to a network taking as input an instance x and generating the desired label $\hat{f}(x)$. We then use Lemma 1 twice, first to form a network that computes the sub-bag labeling function $\hat{g}$, and then to form a network that computes the top-bag labeling function $\hat{h}$.

6 Experiments

We evaluated our model on three experimental setups:

1. we constructed a multi-multi instance semi-synthetic dataset from MNIST, in which digits were organized in bags-of-bags of arbitrary cardinality. This setup is a complex version of the example shown in Section 3. The aim of this experiment is to show the ability of the network to learn functions that have generated the data according to Theorem 1 in Section 5;
2. we constructed semi-synthetic dataset from MNIST, placing digits randomly into a background images of black pixels. The aim of this experiment is to show the disordered convolution property discussed in Section 4.2;
3. we focused on real citation network datasets where data can be naturally decomposed into bags-of-bags (MMI data) or bags (MI data). The goal is to understand whether MMI and MI decompositions are reasonable representations for citation networks and whether MMI representation is more suitable than MI representation. Finally we compared our approach with the state-of-art architecture.

6.1 Learning in the Restricted Setting

The results of Section 5 show that MMI networks can represent any labelling function in the non-counting restricted case. We show here that MMI networks trained by gradient descent can actually *learn* such functions from MMI data. The setup is similar to Example 1 in Section 2, but the classification rule is more complicated. Starting from digit images extracted from the MNIST dataset, we constructed an MMI datasets where each top-bag X is a set of sets of images. For each sub-bag S_j, we denote by $Y_j = \{y_{j,1} \ldots, y_{j,|S_j|}\}$ the set of instance labels in sub-bag S_j, where $y_{j,\ell}$ are derived from the MNIST dataset labels. We then labeled each sub-bag according to the following rules:

– sub-bag S_j has label 0 iff one of the following conditions is satisfied: $\{1,3,5,7\} \subset Y_j$, $\{2,1,3\} \subset Y_j$, $\{3,2,7,9\} \subset Y_j$;
– sub-bag S_j has label 1 iff S_j has not label 0 and one of the following conditions is satisfied: $\{8,9\} \subset Y_j$, $\{4,5,6\} \subset Y_j$, $\{7,2,1\} \subset Y_j$;
– sub-bag S_j has label 2 iff S_j has not label 0 and S_j has not label 1.

Finally each top-bags is positive iff it contains at least a sub-bag of class 1. Observe that data generated according to those rules satisfied the conditions of Definitions 1 and 2. Using these rules, we generated a balanced training set of 5000 top-bags and a balanced test set of 5000 top-bags. Sub-bag and top-bag cardinalities were uniformly sampled in the interval $[2, 6]$. Instances in the training set and in the test set were randomly sampled with replacement from the 60,000 MNIST training images the 10,000 MNIST test images, respectively.

The structure of neural network we used in this experiment is summarized in Table 1. The model was trained by minimizing the L2-regularized binary cross-

Table 1: Neural network structure for multi-multi instance MNIST dataset

Layer	Parameters
Convolutional Layer	kernel size 5×5 with 32 channels
Batch Normalization	
ReLU	
Max Pooling	kernel size 2×2
Convolutional Layer	kernel size 5×5 with 64 channels
Batch Normalization	
ReLU	
Max Pooling	kernel size 2×2
Dense	1024 units
ReLU	
Dropout	probability 0.5
BagLayer (linear activation)	100 units
ReLU	
BagLayer (linear activation)	100 units
ReLU	
Dense	1 unit

entropy loss (with L2 penalty $5 \cdot 10^{-4}$). We stress the fact that instance and sub-bag labels were not used to form the training objective. We ran 200 epochs of the Adam optimizer [10] with learning rate 0.001. Results in Table 2 confirm that the network is able to recover the latent logic function that was used in the data generation process with a reasonably high accuracy.

Table 2: Accuracies on multi-multi instance MNIST dataset

set	loss	accuracy
training	0.063	98.60%
test	0.108	96.44%

6.2 Multi-multi instance as Disordered Pooling

As discussed in Section 4.2, bag-layers can be used as disordered convolutional layers. The aim of the following experiment is to show this property on a simple scenario, and to compare results with a standard convolutional approach. We first constructed a dataset MNIST^{EXT} by placing the 28×28 pixels MNIST digit images on a black background of size $w \times w$ (we generated datasets for several values of w, i.e. $w \in \{105, 135, 165, 195\}$). MNIST^{EXT} images are labelled with the digit they contain. The goal is to classify digits regardless of their location within the images. Our purpose here is to show how the location-invariance of the bag-layer function can be exploited to obtain robust classification results for raw image data without centering or cropping preprocessing. As MNIST, the MNIST^{EXT} dataset, is split in training and test sets containing respectively 60,000 images and 10,000 images. Digits of training and test images are placed in the top and bottom half of the background, respectively. On the same datasets we trained a MMI network and a standard convolutional network.

Concerning the MMI network we constructed MMI data as follows: each top-bag X represents an image in MNIST^{EXT}. For each we extracted *macro-patches* of size 15×15 and consecutive macro-patches are overlapped by 5 pixels. Each macro-patch represents a sub-bag $S_j \in X$. For each macro-patch we extracted *micro-patches* of size 5×5 and consecutive micro-patches are overlapped by 3 pixels. Each micro-patch represents an instance $x_{j,\ell} \in S_j$. Note that the choice of the micro-patches, macro-patches and w sizes is in general arbitrary. Here we chose those specific values for avoiding, as far as possibile, the zero padding for adapting the sizes and then waste of computational resources. Furthermore choosing w fixed for each experiment is crucial in order to compare the MMI network with the convolutional neural network. Indeed while our approach can handle bags of different size the convolutional model requires that input images have fixed size. The structure of the MMI network we used in this experiment is summarized in Table 3. The model was trained by minimizing the categorical cross-entropy loss. We ran 20 epochs of the Adam optimizer with learning rate 0.0001.

Concerning the convolutional network we use a model composed of 4 convolutional blocks, a dense layer of 1024 units with ReLU activation, a Dropout layer with probability 0.5 and a Softmax Layer of 10 units. The first convolutional block contains a convolutional layer with a kernel of size 7×7 and 32 channels while the other blocks contain convolutional layers of size 5×5 and 64 channels. Each convolutional layer is followed by a ReLU activation, MaxPooling of size 2×2 and a Dropout layer with probability 0.5. The model was trained by minimizing the categorical cross-entropy loss. We ran 20 epochs of the Adam optimizer with learning rate 0.001.

Table 3: Structure of MMI network for discovering spatial-independent features

Layer	Parameters
BagLayer (linear activation)	100 units
ReLU	
BagLayer (linear activation)	200 units
ReLU	
Dense	1024 units
ReLU	
Dropout	probability 0.5
Dense	10 units

In Figure 5 we report the accuracy of MMI network and CNNs as a function of w. We note that the accuracy of the CNN decreases as w grows large, while accuracy the MMI network remains stable. Those results confirm that MMI network is able to learn location-invariant features.

6.3 Citation Network Datasets

In Section 4.2 we discussed the possible benefits that graphs can obtain through bag-layers and then MMI networks. We show here a natural way for decomposing graphs into MMI data. We also show a way for decomposing graphs into multi-instance data (which we abbreviate with MI data) which is still a reasonable representation for graphs. We considered three graph citation datasets: Citeseer, Cora and PubMed [19]. Nodes represent papers described by titles and abstracts and edges are citation links. We treat the citation links as undirected edges. The goal is to classify each node of the graph. In Table 4 are reported the statistics for each dataset.

MMI data was constructed from citation networks in which a top-bag X represents a bag of nodes: the neighborhood of a node (including the node itself). A sub-bag $S_j \in X$ represents the bag of words corresponding to text attached to the node. An instance $x_{j,\ell} \in S_j$ is a word represented with GloVe word vectors [16]. MI data was constructed from citation networks in which bags are

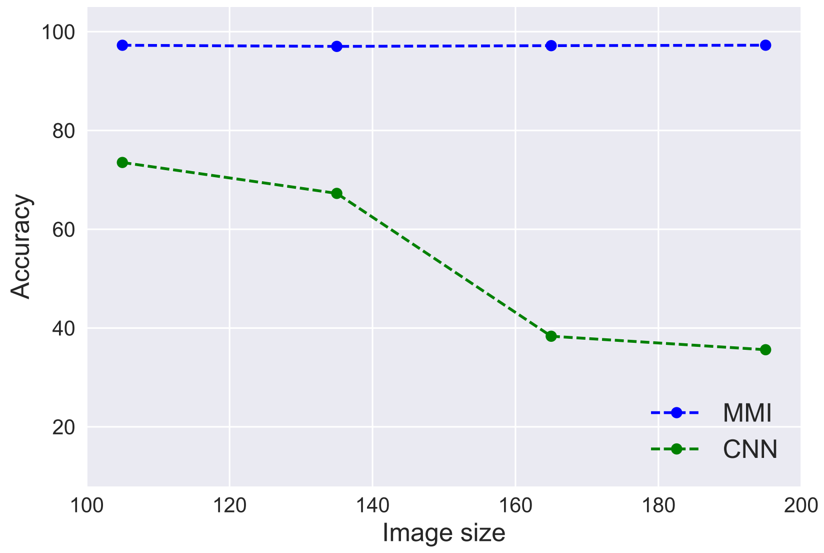

Fig. 5: Accuracies of MMI network (blue) and convolutional neural network (green) in function of window size w.

the set of bag of words of the neighborhood of a node (including the node itself). Note that in general both the cardinality of bags for MMI data and MI data could differ.

The structure of neural networks we used for those experiments is summarized in Table 5 (all bag-layers have linear activation). All models were trained by minimizing the softmax cross-entropy loss. We ran 400 epochs of the Adam optimizer with learning rate 0.0001 on 10 randomly drawn splits: the training set is composed of 20 top-bags for each class and the test set is composed of 1000 top-bags. Results in Table 6 report the average accuracy evaluated on the same training/test splits for MMI network, MI network and the state-of-art architecture [11]. In Table 2 of [11], GCN is compared against several methods, outperforming all of of them. Hence we only compare against GCN. The results confirm that it is reasonable to represent graphs as MI data, the performance being competitive with the current state-of-art approach. Nevertheless MMI decomposition allows to obtain better results than MI decomposition. This fact suggests that in this scenario graphs benefit from the more nested structure of MMI data. Finally MMI networks trained on such MMI data can slightly improve the state-of-art in two out of three datasets.

Table 4: Citations dataset

Dataset	#Classes	#Nodes	#Edges
CITESEER	6	3,327	4,732
CORA	7	2,708	5,429
PUBMED	3	19,717	44,338

Table 5: Structure of MI networks (left) and MMI networks (right); BL stands for BagLayer

MI			MMI		
Citeseer	Cora	Pubmed	Citeseer	Cora	Pubmed
BL(2000)	BL(2000)	BL(3000)	BL(2000)	BL(2000)	BL(3000)
Sigmoid	ReLU	Sigmoid	Sigmoid	ReLU	Sigmoid
Dense(6)	Dense(7)	Dense(3)	BL(2000)	BL(2000)	BL(2000)
			Sigmoid	ReLU	Sigmoid
			Dense(6)	Dense(7)	Dense(3)

Table 6: Averaged accuracies (in percent)

Method	Citeseer	Cora	Pubmed
multi instance	68.5 ± 2.1	77.6 ± 1.3	76.0 ± 2.0
multi-multi instance	**71.2±1.7**	**79.2±1.5**	77.1 ± 2.0
GCN[11]	69.6 ± 1.9	78.8 ± 1.1	**78.2±2.1**

7 Conclusions

We have introduced the MMI framework for handling data organized in nested bags. The MMI setting allows for a natural hierarchical organization of data, where components at different levels of the hierarchy are unconstrained in their cardinality. We have identified several learning problems that can be naturally expressed as MMI problems. For instance, image or graph classification are promising application areas, because here the examples can be objects of varying structure and size, for which a bag-of-bag data representation is quite suitable, and can provide a natural alternative to graph kernels or convolutional network for graphs. The fact that bags do not impose an order on their elements directly leads to useful spatial invariance properties when applied to image data.

We proposed a neural network architecture involving the new construct of bag layers for learning in the MMI setting. Theoretical results show the expressivity of this type of model. In the empirical results we have shown that learning MMI models from data is feasible, and the theoretical capabilities of MMI networks can be exploited in practice, e.g., to learn accurate models for noiseless data, or location invariant models for image classification.

In this paper, we have focused on the setting where whole bags-of-bags are to be classified. In conventional MI learning, it is also possible to define a task where individual instances are to be classified. Such a task is however less clearly defined in our setup since we do not assume to know the label spaces at the instance and sub-bag level, nor the functional relationship between the labels at the different levels. It is an interesting subject of future work to extend our approach also to also identify the unknown labels associated with instances and sub-bags. Interestingly, this direction of research might also connect to the interpretation of the behavior of the network, which at the moment, like all related deep learning techniques, is totally opaque.

References

1. Andrews, S., Tsochantaridis, I., Hofmann, T.: Support vector machines for multiple-instance learning. In: Advances in neural information processing systems. pp. 561–568 (2002), `http://machinelearning.wustl.edu/mlpapers/paper_files/AA10.pdf`, 00828
2. Arbeláez, P., Maire, M., Fowlkes, C., Malik, J.: Contour Detection and Hierarchical Image Segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 33(5), 898–916 (May 2011), `http://ieeexplore.ieee.org/document/5557884/`, 00000
3. Atwood, J., Towsley, D.: Diffusion-convolutional neural networks. In: Advances in Neural Information Processing Systems. pp. 1993–2001 (2016), `http://papers.nips.cc/paper/6212-diffusion-convolutional-neural-networks`
4. Dietterich, T.G., Lathrop, R.H., Lozano-Pérez, T.: Solving the multiple instance problem with axis-parallel rectangles. Artificial Intelligence 89(1–2), 31–71 (Jan 1997), `http://www.sciencedirect.com/science/article/pii/S0004370296000343`, 01439
5. Duvenaud, D., Maclaurin, D., Aguilera-Iparraguirre, J., Gómez-Bombarelli, R., Hirzel, T., Aspuru-Guzik, A., Adams, R.P.: Convolutional Networks on Graphs for Learning Molecular Fingerprints. arXiv:1509.09292 [cs, stat] (Sep 2015), `http://arxiv.org/abs/1509.09292`, arXiv: 1509.09292
6. Foulds, J., Frank, E.: A review of multi-instance learning assumptions. The Knowledge Engineering Review 25(01), 1 (Mar 2010), `http://www.journals.cambridge.org/abstract_S026988890999035X`, 00081
7. Fukushima, K.: Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological cybernetics 36(4), 193–202 (1980), `http://link.springer.com/article/10.1007/BF00344251`, 01681
8. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural networks 2(5), 359–366 (1989), `http://www.sciencedirect.com/science/article/pii/0893608089900208`, 12001
9. Jaeger, M.: Relational Bayesian Networks. In: arXiv:1302.1550 [cs] (1997), `http://arxiv.org/abs/1302.1550`, 00252 arXiv: 1302.1550

10. Kingma, D., Ba, J.: Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs] (Dec 2014), `http://arxiv.org/abs/1412.6980`, 00204 arXiv: 1412.6980
11. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016), `https://arxiv.org/abs/1609.02907`, 00020
12. LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Backpropagation applied to handwritten zip code recognition. Neural computation 1(4), 541–551 (1989), `http://www.mitpressjournals.org/doi/abs/10.1162/neco.1989.1.4.541`, 01543
13. Maron, O., Lozano-Pérez, T.: A framework for multiple-instance learning. Advances in neural information processing systems pp. 570–576 (1998), `http://lamda.nju.edu.cn/zhangml/files/NIPS97.pdf`, 00870
14. Natarajan, S., Tadepalli, P., Dietterich, T.G., Fern, A.: Learning first-order probabilistic models with combining rules. Annals of Mathematics and Artificial Intelligence 54(1-3), 223–256 (2008), `http://link.springer.com/article/10.1007/s10472-009-9138-5`, 00069
15. Niepert, M., Ahmed, M., Kutzkov, K.: Learning Convolutional Neural Networks for Graphs. New York, NY, USA (May 2016), `http://arxiv.org/abs/1605.05273`, 00001 arXiv: 1605.05273
16. Pennington, J., Socher, R., Manning, C.D.: Glove: Global Vectors for Word Representation. In: EMNLP. vol. 14, pp. 1532–1543 (2014), `http://llcao.net/cu-deeplearning15/presentation/nn-pres.pdf`, 00365
17. Ramon, J., De Raedt, L.: Multi instance neural networks (2000), `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.682`, 00115
18. Scott, S., Zhang, J., Brown, J.: On generalized multiple-instance learning. International Journal of Computational Intelligence and Applications 5(01), 21–35 (2005), `http://www.worldscientific.com/doi/abs/10.1142/S1469026805001453`, 00059
19. Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., Eliassi-Rad, T.: Collective classification in network data. AI magazine 29(3), 93 (2008), `https://vvvvw.aaai.org/ojs/index.php/aimagazine/article/view/2157`, 00567
20. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. arXiv:1602.07261 [cs] (Feb 2016), `http://arxiv.org/abs/1602.07261`, 00127 arXiv: 1602.07261
21. Wang, J., Zucker, J.D.: Solving multiple-instance problem: A lazy learning approach (2000), `http://cogprints.org/2124`, 00444
22. Weidmann, N., Frank, E., Pfahringer, B.: A two-level learning method for generalized multi-instance problems. In: European Conference on Machine Learning. pp. 468–479. Springer (2003), `http://link.springer.com/chapter/10.1007/978-3-540-39857-8_42`, 00110
23. Zhang, Q., Goldman, S.A., others: EM-DD: An improved multiple-instance learning technique. In: NIPS. vol. 1, pp. 1073–1080 (2001), `https://papers.nips.cc/paper/1959-em-dd-an-improved-multiple-instance-learning-technique.pdf`
24. Zhou, Z.H., Zhang, M.L.: Neural networks for multi-instance learning. In: Proceedings of the International Conference on Intelligent Information Technology, Beijing, China. pp. 455–459 (2002), `http://cs.nju.edu.cn/zhouzh/zhouzh.files/publication/techrep02.pdf`, 00066