Integrating Logic Knowledge into Graph Regularization: 
an application to image tagging

Claudio Saccà
Information Engineering
Department
Via Roma 54
Siena, Italy
claudiosac@gmail.com

Michelangelo Diligenti
Information Engineering
Department
Via Roma 54
Siena, Italy
diligmic@dii.unisi.it

Marco Gori
Information Engineering
Department
Via Roma 54
Siena, Italy
marco@dii.unisi.it

Marco Maggini
Information Engineering
Department
Via Roma 54
Siena, Italy
maggini@dii.unisi.it

ABSTRACT
This paper studies how prior knowledge in form of First Order Logic (FOL) clauses can be converted into a set of continuous constraints. These constraints can be directly integrated into a learning framework allowing to jointly learn from examples and semantic knowledge. In particular, in this paper we show how the constraints can be integrated into a regularization schema working over discrete domains. We consider tasks in which items are connected to each other by given relationships, thus yielding a graph, whose nodes correspond to the available objects. It is required to estimate a set of functions defined on each node of the graph, given a small set of labeled nodes for each function. The FOL constraints enforce dependencies, resulting from the FOL knowledge, among the values that the functions assume over the nodes. The experimental results evaluate the proposed technique on an image tagging task, showing how the proposed approach provides a significantly higher tagging accuracy than simple graph regularization. The experimental results show how the selection of a proper conversion process of the FOL clauses is fundamental in order to achieve good results.

1. INTRODUCTION
Tagging of resources like images, textual documents and videos is an important functionality in social networks and it is emerging as an effective technique to enrich shared resources with semantic meta-information. A high precision tagging allows the deployment of more sophisticated information retrieval mechanisms that are currently provided by search engines. In particular, tags associated to images usually summarize their semantic content that would be difficult to interpret automatically with state-of-the-art image understanding techniques. However, a manual collective tagging process has many limitations, as it is not suited for very large or dynamic collections and it does not provide high consistency of the tags across the images, creating many issues for their subsequent use [8]. In fact, in the context of social networks and folksonomies the tag dictionary is freely built by the users without an actual agreement on a common semantic interpretation or knowledge model.

Automatic image tagging can be formulated as a classification problem, where the number of tags is typically high and the classes are not mutually exclusive, thus yielding a multi-label classification task. In this paper we take the approach of using the manually inserted tags as training data for a tagger, that generalizes the supervisions to new images. However, when an automatic image tagger is trained over the tags inserted by the users of a folksonomy, it may inherit the same inconsistencies of the training data.

The approach presented in this paper exploits a multi-relational representation of the data as proposed in [10], where each image corresponds to a node in a graph and different semantic relations like content similarity, friendship in the social network and authorship can be used to establish connections between the corresponding nodes. A few nodes are labeled with explicit tags inserted by one or more users. Like other transductive approaches [1, 14, 13], the proposed solution generalizes the tags to the unlabeled data via a diffusion process. However, the proposed solution is also able to take into account the available prior knowledge defined by FOL clauses, that enforce the consistency of the assigned tags without depending on specially trained human taggers.

In particular, the FOL clauses are built on variables ranging over the nodes of the graph (the images) and task predicates, each one associated to a tag [3]. The FOL clauses are compiled into a set of equivalent continuous constraints, and the integration between logic and learning is implemented via a novel multi-task learning scheme, which combines the loss on the supervised examples and a penalty term resulting from the conversion of the logic knowledge. This paper proposes...
and evaluates different approaches for the conversion of the logic knowledge, using t-norms or different functions based on mixture of Gaussians.

Other specific tagging methods have been devised for classifying networked items. In particular, some approaches considering the images in their isolation perform image annotation and tagging using generative or latent models [7, 9]. Other approaches perform tag propagation on the network using random walks or semi-supervised learning [2]. Other more recent methods take advantage of the semantic relations among tags [11, 12] exploit tag co-occurrence in order to improve their predicted ranking.

The experimental results show that the employed conversion schema has a very strong impact on the precision of the tagging and that an appropriate conversion can determine a significant improvement of the tagging accuracy with respect to a standard classifier not incorporating the prior knowledge.

The paper is organized as follows. The next section presents the proposed approach for multi-task learning on a graph of entities when a set of known constraints is to be enforced. Then, section 3 describes how the available FOL knowledge base can be converted into a set of continuous constraints involving the tagging functions. Finally, section 4 reports the evaluation on a dataset collected from Flickr1, where the tag dictionary is large and not centrally controlled, resulting in noisy and inconsistent training data.

2. MULTI-TASK TRANSDUCTIVE LEARNING WITH CONSTRAINTS

We consider a given set of entities and relationships among them. Hence, the input can be modeled by a graph where each node represents an entity and an edge encodes some semantic relation between the connected nodes. We assume that the pertinence of each node to each tag has to be determined given a partial labeling of all the nodes in the graph. Therefore, it is possible to approach this task using regularization over discrete domains, performing a separate diffusion over the graph for each tag.

2.1 Regularization in Discrete Domains

We consider a general formulation of transductive learning in discrete domains based on a regularization principle [14, 13]. This class of algorithms exploits a set of examples labeled nodes and the connections (edges) among the objects. Entity classification is performed by computing a function that is defined over the graph nodes (each node representing a labeled node) and the connections (edges) among the objects. Hence, the input can be modeled by a graph where each node represents an entity and an edge encodes some semantic relation between the connected nodes. We assume that the pertinence of each node to each tag has to be determined given a partial labeling of all the nodes in the graph. Therefore, it is possible to approach this task using regularization over discrete domains, performing a separate diffusion over the graph for each tag.

We consider a general formulation of transductive learning in discrete domains based on a regularization principle [14, 13]. This class of algorithms exploits a set of examples labeled nodes and the connections (edges) among the objects. Entity classification is performed by computing a function that is defined over the graph nodes (each node representing a labeled node) and the connections (edges) among the objects. Hence, the input can be modeled by a graph where each node represents an entity and an edge encodes some semantic relation between the connected nodes. We assume that the pertinence of each node to each tag has to be determined given a partial labeling of all the nodes in the graph. Therefore, it is possible to approach this task using regularization over discrete domains, performing a separate diffusion over the graph for each tag.

We consider a general formulation of transductive learning in discrete domains based on a regularization principle [14, 13]. This class of algorithms exploits a set of examples labeled nodes and the connections (edges) among the objects. Entity classification is performed by computing a function that is defined over the graph nodes (each node representing a labeled node) and the connections (edges) among the objects. Hence, the input can be modeled by a graph where each node represents an entity and an edge encodes some semantic relation between the connected nodes. We assume that the pertinence of each node to each tag has to be determined given a partial labeling of all the nodes in the graph. Therefore, it is possible to approach this task using regularization over discrete domains, performing a separate diffusion over the graph for each tag.

The optimal solution minimizing equation (1) can be computed by finding its stationary points, obtained by solving

\[ \nabla C_G[f] = (f - y) + \lambda R_G f = 0. \]

If \((I + \lambda R_G)\) is invertible, \(f^*\) exists, it is unique and equal to

\[ f^* = (I + \lambda R_G)^{-1} y. \]

Equation (2) requires the inversion of a square matrix, which has size equal to the number of nodes in the input graph. When the graph is large, direct inversion is not feasible. However, if the largest eigenvalue of \(\lambda R_G\) lays inside the unit circle and \(R_G\) is sparse, the solution can be efficiently found by solving the following iterative equation:

\[ f^{n+1} = y - \lambda R_G f^n. \]

Interestingly, this iterative equation represents a diffusion process of the labels through the graph.

In order to provide a meaningful definition of \(R_G\), we start from a regularization functional \(C_G[f]\) that penalizes the distance between the values computed for pairs of connected nodes, weighted by the strength of the connection,

\[ C_G[f] = \frac{1}{2} \sum_{u,z=1}^{V_G} w_{uz} (f(u) - f(z))^2. \]

This functional favors functions assigning close values on nodes that are strongly connected. Equation (3) can be rearranged as

\[ C_G[f] = \frac{1}{2} \sum_{u,z=1}^{V_G} \left( w_{uz}^2 + w_{zu}^2 \right) \frac{1}{2} (f(u) - f(z))^2. \]
If we define $\bar{w}_{ux} = \frac{(w_{ux} + w_{yu})}{2}$, equation (4) becomes

$$C^R_G[f] = \frac{1}{2} \sum_{u=1}^{\left|V_G\right|} \sum_{z=1}^{\left|V_G\right|} \bar{w}_{uz} (f(u) - f(z))^2 =$$

$$= \frac{1}{2} \sum_{u=1}^{\left|V_G\right|} \sum_{z=1}^{\left|V_G\right|} \bar{w}_{uz} f^2(u) + \frac{1}{2} \sum_{u=1}^{\left|V_G\right|} \sum_{z=1}^{\left|V_G\right|} \bar{w}_{uz} f^2(z) +$$

$$- \sum_{u=1}^{\left|V_G\right|} \bar{w}_{uz} f(u)f(z).$$

The weights $\bar{w}_{uz}$ are symmetric ($\bar{w}_{uz} = \bar{w}_{zu}$ $\forall u, z$), therefore it holds that

$$\sum_{u=1}^{\left|V_G\right|} \sum_{z=1}^{\left|V_G\right|} \bar{w}_{uz} f(u)^2 = \sum_{u=1}^{\left|V_G\right|} \sum_{z=1}^{\left|V_G\right|} \bar{w}_{uz} f(z)^2.$$

Thus, we obtain

$$C^R_G[f] = \sum_{u=1}^{\left|V_G\right|} \sum_{z=1}^{\left|V_G\right|} \bar{w}_{uz} f^2(u) - \sum_{u=1}^{\left|V_G\right|} \sum_{z=1}^{\left|V_G\right|} \bar{w}_{uz} f(u)f(z).$$

Let $W$ be a symmetric square matrix having $\bar{w}_{uz}$ as the $(u, z)$-th element and $D$ be a diagonal matrix with its $u$-th element $d_u$ equal to $\sum_{z=1}^{\left|V_G\right|} \bar{w}_{uz}$, then the two terms on the right side of equation (5) can be expressed as $f^T D f$ and $f^T W f$, respectively.

Therefore, $C^R_G[f]$ can be compactly rewritten as

$$C^R_G[f] = f^T (D - W) f$$

which expresses the regularizer in the form required by equation (1). Now, setting $R_G = D - W$ into equation 2 allows us to compute the optimal score vector as

$$f^* = \left( I + \lambda_D - \lambda_W \right)^{-1} y.$$

Since $D - W$ is diagonally dominant, $I + \lambda_D - \lambda_W$ is also diagonally dominant and, therefore, invertible. Thus, the optimal solution $f^*$ exists and it is uniquely defined by the graph and the supervised vector of the targets.

### 2.2 Learning with constraints

We assume to have available a dictionary $T$ of tags (categories), whose size we indicate with $|T|$. We consider a multitask learning problem in which we must decide which tags are to be assigned to each node. Therefore, a set of multitask learning problem in which we must decide which categories, whose size we indicate with $\{\Phi\}$.

Let $W$ be a symmetric square matrix having $\bar{w}_{uz}$ as the $(u, z)$-th element and $D$ be a diagonal matrix with its $u$-th element $d_u$ equal to $\sum_{z=1}^{\left|V_G\right|} \bar{w}_{uz}$, then the two terms on the right side of equation (5) can be expressed as $f^T D f$ and $f^T W f$, respectively.

Therefore, $C^R_G[f]$ can be compactly rewritten as

$$C^R_G[f] = f^T (D - W) f$$

which expresses the regularizer in the form required by equation (1). Now, setting $R_G = D - W$ into equation 2 allows us to compute the optimal score vector as

$$f^* = \left( I + \lambda_D - \lambda_W \right)^{-1} y.$$

Since $D - W$ is diagonally dominant, $I + \lambda_D - \lambda_W$ is also diagonally dominant and, therefore, invertible. Thus, the optimal solution $f^*$ exists and it is uniquely defined by the graph and the supervised vector of the targets.

### 3. TRANSLATION OF FOL CLAUSES INTO REAL-VALUED CONSTRAINTS

The learning from constraints approach described in the previous section requires to devise a conversion process to translate a set of logic formula into real-valued functions. We focus our attention on knowledge-based descriptions given by first-order logic (FOL–KB). While the framework can be easily extended to arbitrary FOL predicates, in this paper we will consider only unary predicates to keep the notation simple. In the following, we indicate by $V = \{v_1, \ldots, v_N\}$ the set of the variables used in the KB, with $v_i \in V_G$ denoting a node in the graph. A predicate in the KB takes a node (or more nodes for n-ary predicates) in the graph and associates it with a true/false value depending whether a given property applies to the node. Therefore, given the set of predicates used in the KB $P = \{p_k : v \in V_G \rightarrow \{true, false\}, k = 1, \ldots, |T|\}$, the clauses will be built from the set of atoms $p(v) : p \in P, v \in V$.

Any FOL clause has an equivalent version in *Prenex Normal form* (PNF), that has all the quantifiers ($\forall, \exists$) and their associated quantified variables at the beginning of the clause. Standard methods exist to convert a generic FOL clause into its corresponding PNF and the conversion can be easily automated. Therefore, without loss of generality, we restrict our attention to FOL clauses in the PNF form. We assume that the task functions $f_k, k = 1, \ldots, |T|$ are exploited to implement the predicates in $P$. In this framework, the predicates yield a continuous real value that can be interpreted as a *truth degree*. The FOL–KB will contain a set of
3.1 Conversion of the Propositional Expression

Different methods can be used for the conversion of the propositional expression into a continuous function. In particular, two families of conversion approaches have been proposed in the literature: t-norms and mixtures of Gaussians.

T-norms.

In the context of fuzzy logic, t-norms [6] are commonly used as a generalization of logic clauses to continuous variables.

A t-norm is a function $t: [0,1] \times [0,1] \rightarrow [0,1]$ that is commutative (i.e. $t(x,y) = t(y,x)$), associative (i.e. $t(x,t(y,z)) = t(t(x,y),z)$), monotonic (i.e. $y \leq z \Rightarrow t(x,y) \leq t(x,z)$), and featuring a neutral element 1 (i.e. $t(x,1) = x$). A t-norm fuzzy logic is defined by its t-norm $t(x,y)$ that models the logic AND, while the negation of a variable $\neg x$ is computed as $1 - x$. The t-conorm, modeling the logical OR, is defined as $1 - t((1 - x), (1 - y))$, as a generalization of the De Morgan's law $x \lor y = \neg (\neg x \land \neg y)$. Many different t-norm/ t-conorm functions have been proposed in the literature. In the following we will mainly focus on the product t-norm $t(x,y) = x \cdot y$, for which the t-conorm is computed as $1 - (1 - x)(1 - y) = x + y - xy$. Another commonly used t-norm is the minimum t-norm defined as $t(x,y) = \min(x,y)$. In this case, the t-conorm corresponds to the function $\max(x,y)$. It is clear from their definition that both the product and minimum t-norms are of class $C^1$ (differentiable functions whose derivative is continuous). Once defined the t-norm functions corresponding to the logical AND, OR and NOT, these functions can be composed to convert any arbitrary logic proposition. A t-norm is continuous if $t(x,y)$ is continuous and, when using a continuous t-norm, an arbitrary proposition is converted into a continuous function.

T-norms have been employed in previous work in the literature to integrate prior knowledge into kernel machines [4].

Mixtures of Gaussians.

A different approach based on mixtures of Gaussians has been proposed in [5] in the context of symbolic learning using neural networks. Unlike t-norms, this approach generalizes the logic clause without making any independence assumption among the variables. In particular, let us consider a propositional logic clause involving $n$ logic variables. The logic clause is equivalent to its truth table containing $2^n$ rows, each one corresponding to a configuration of the variables. The continuous function approximating the clause is based on a set of Gaussian functions, each one centered on a configuration corresponding to the true value in the truth table. The mixture function sums all the Gaussians:

$$t(x_1, \ldots, x_n) = \sum_{[c_1, \ldots, c_n] \in \mathcal{T}} \exp \left( -\frac{||[x_1, \ldots, x_n] - [c_1, \ldots, c_n]||^2}{2\sigma^2} \right).$$

where $x_1, \ldots, x_n$ is the set of variables in the propositional clause and $\mathcal{T}$ is the set of all possible configurations of the input variables which correspond to the true value in the table. We indicate as $PGAUSS$ this conversion procedure.

For example, let us consider the clause $x \lor y$, which is verified by the three configurations $[true, true]$, $[true, false]$, and $[false, true]$. The clause is converted as

$$t(x,y) = \exp \left( -\frac{||[x,y] - [1,1]||^2}{2\sigma^2} \right) + \exp \left( -\frac{||[x,y] - [1,0]||^2}{2\sigma^2} \right) + \exp \left( -\frac{||[x,y] - [0,1]||^2}{2\sigma^2} \right).$$

If $[x,y]$ assumes values corresponding to a configuration verifying the clause, $t(x,y) \geq 1$ holds. Otherwise, the value of $t(x,y)$ will decrease depending on the distance from the clos-
est configuration verifying the clause. The variance $\sigma^2$ is a parameter that can be used to determine how quickly $t(x, y)$ decreases when moving away from a configuration verifying the constraint. Please note that each configuration verifying the constraint is always a global maximum of $t$ when using a small enough $\sigma$ value. See [5] for a complete discussion on how to select a suitable $\sigma$ value.

A variant of this approach (MGAUSS) is to consider only the Gaussian centered in the closest true configuration

$$t(x_1, \ldots, x_n) = \max \left[ \exp \left( -\frac{\left\| x_1 - c_{a_1} \right\|^2}{2\sigma^2} - \frac{\left\| x_2 - c_{a_2} \right\|^2}{2\sigma^2} \right) \right].$$

In this case, one negative Gaussian is centered on each configuration of variables yielding a false value of the considered clause. A bias value equal to 1 is introduced to obtain a default true value when distant from a false configuration

$$t(x_1, \ldots, x_n) = 1 - \max \left[ \exp \left( -\frac{\left\| x_1 - c_{a_1} \right\|^2}{2\sigma^2} - \frac{\left\| x_2 - c_{a_2} \right\|^2}{2\sigma^2} \right) \right].$$

where $F$ is the set of input configurations corresponding to a false value in the truth table.

Using this conversion procedure, the clause $x \lor y$ would be converted as

$$t(x, y) = 1 - \exp \left( -\frac{||x||^2}{2\sigma^2} \right).$$

The conversion schema modelling the true or false configurations resemble the duality in the representation of any propositional formula in Disjunctive or Conjunctive Normal Form. Figure 1 shows the functions obtained by converting the clause $a \Rightarrow b$ using PGAUSS and NGAUSS. Any formula can be converted using both forms. However, depending on the formula, one configuration can be more compact. Compact mixtures should be generally preferred as, when integrated into a cost function to be optimized as usually done in machine learning applications, they introduce less local minima. The experimental results, reported in the last section of this paper, confirm this claim.

3.2 Quantifier conversion

The quantified portion of the expression is processed recursively by moving backward from the inner quantifier in the PNF expansion. When processing the universal quantifier, the expression must hold for any node of the graph, and it can be naturally converted measuring the degree of non-satisfaction of the expression over the domain $V_G$. We indicate with $v_E$ the vector of variables contained in the expression $E$. The satisfaction measure corresponds to the overall distance of the penalty associated to $E$, i.e. $\varphi_E(v_E, F)$ from the constant function equal to 0 over the domain $V_G$. Using the infinity norm on discrete domains, this measure is

$$\forall v_q \in V_G \quad E(v_E, P) \rightarrow \max_{v_E \in V_G} \left| \varphi_E(v_E, F) \right|,$$

where the resulting expression depends on all the variables in $v_E$ except $v_q$, and $\varphi_E(v_E, F) = \max(1 - t_E(A(v_E, F), 0)$, where $A$ maps the values assumed by the predicate functions $F$ to the grounded variables $v_E$ in order to obtain the correct argument list to the function $t_E(\cdot)$, that implements the propositional expression $E$. Hence, the result of the conversion applied to the expression $E_1(v_E, P) = \left( v_{s_1} \in V_G \quad E(v_E, P) \right)$ is a functional $\varphi(E_1(v_E, F))$, assuming values in $[0,1]$ and depending on the set of variables $v_E = [v_1(v_1, E), \ldots, v_n(v_n, E)]$, such that $n_{s_1} = n_E - 1$ and $v_{s_1}(v_1, E) \in \{v_v \in V \mid \exists v \in s_1(E), v_v \neq v_v\}$. The variables in $v_{s_1}$ need to be quantified or assigned a specific value in order to obtain a constraint functional depending only on the functions $F$.

The existential quantifier can be realized by enforcing the De Morgan law

$$\exists v_q \in V_G \quad E(v_E, P) \iff \neg\forall v_q \in V_G \quad \neg E(v_E, P).$$

It is also possible to select a different norm on the discrete domain to convert the universal quantifier. For example, when using the $\| \cdot \|_1$ norm for discrete domains, yields the conversion rule

$$\forall v_q \in V_G \quad E(v_E, P) \rightarrow \frac{1}{|V_G|} \sum_{v_q \in V_G} |\varphi_E(v_E, F)|.$$

As an example of the conversion procedure, let $a(\cdot), b(\cdot)$ be two predicates, implemented by the function vectors $f_a, f_b$. The clause $\forall v_1 \in V_G \quad v_2 \in V_G \quad (a(v_1) \land \neg b(v_2)) \lor (\neg a(v_1) \land b(v_2))$ is converted starting with the conversion of the quantifier free expression

$$E_0([v_1, v_2], (a(\cdot), b(\cdot)) = (a(v_1) \land \neg b(v_2)) \lor (\neg a(v_1) \land b(v_2)),$$

verified if $a(v_1) = true$, $b(v_2) = false$ or $a(v_1) = false$, $b(v_2) = true$, as:

$$t_{E_0}(f_a(v_1), f_b(v_2)) = \exp \left( -\frac{f_a(v_1)^2 + (f_b(v_2) - 1)^2}{2\sigma^2} \right) + \exp \left( -\frac{(f_a(v_1) - 1)^2 + f_b(v_2)^2}{2\sigma^2} \right).$$

Then, $E_0([v_1, v_2], (a(\cdot), b(\cdot)))$ is converted into the distance measure and the two universal quantifiers are converted using the infinity norm over the set of nodes, yielding the constraint

$$\phi(F) = \varphi_{E_0}(\{[f_a, f_b]\}) = \max_{v_1 \in V_G} \max_{v_2 \in V_G} [\max(1 - t_{E_0}(f_a(v_1), f_b(v_2)), 0)] = 0.$$
The datasets used in the experiments have been constructed by downloading a set of images from the Flicker hosting and online community website. Each image is associated to a textual description and title assigned by the author. The images have also been tagged by the users of the Flicker social network using a set of tags. Typically, most images are tagged by the users with 1 to 6 tags. The dataset has been built by randomly selecting 10,000 images, each one assigned to at least one of the 80 most frequent tags in the dataset.

The graph of images is built by establishing connections among pairs of images. In particular, let $u$ and $z$ be two nodes of the graph (corresponding to two images in the dataset), the value $w_{uz}$ is attached to the edge $(u, z)$ as $w_{uz} = \text{cos}_\text{sim}(title_u, title_z)$, where $\text{cos}_\text{sim}$ indicates the cosine similarity function and $title_u$ and $title_z$ are the TF-IDF bag-of-words representations of the title of $u$ and $z$, respectively. To reduce the number of edges, only edges with a weight above 0.1 are kept. The resulting graph contains approximately 311,000 edges.

Other similarity metrics could have been exploited like the similarity of the textual descriptions of the images, authorship, friendship between the corresponding authors, etc. We decided to not include any relation based on the visual similarity of the images, as state-of-the-art methods still fail to correlate well with the semantic similarity of the image contents. However, if available, any visual similarly metric could be integrated as the other relations. Once the graph is built, the user-provided tags have been removed from 75% of the nodes. Furthermore, a knowledge base containing a set of rules has been compiled to express the semantic relationships between the tags. In particular, 45 rules have been defined. Table 1 shows a small sample of the rules inserted in the knowledge base.

Each function associated to a tag is obtained by a regularization-based transduction on the graph. The conversion of logic rules into a continuous form has been performed using the different approaches described in section 3. We indicate with MNORM and PNORM the min-max and product t-norms, respectively. PGAUSS, MGAUSS, NGAUSS indicate the mixture-of-gaussians conversion of the logic clause using the definitions in equations (7), (8) and (9), respectively. In the experiments, the $\sigma$ value for the mixture of Gaussians is a critical parameter. A tuning on this parameter has been performed and an optimal value of 0.05, 0.02, 0.1 has been found for MGAUSS, PGAUSS and NGAUSS respectively.

The results are summarized in table 2, where SGR indicates the graph regularization when using logic rules for the specified conversion schema, and GR when no logic rules are used. The first column in the table reports the relative accuracy of the methods that are compared, as the percentage of images for which a rater preferred the SGR generated list over the GR one, given that she/he expressed a preference. The second column, instead, reports the percentage of images for which the raters expressed a preference for one of the two rankings. These latter values measure the impact of the introduction of the logic rules.

PNORM outperforms MNORM by a significant margin both in terms of accuracy and impact. MNORM does not provide a significantly improvement versus not using the logic rules (the raters expressed a preference for the baseline experiment in half of the cases). MGAUSS works similarly
to PNORM and it provides the highest impact among all the conversion schema. NGAUSS performs very well on this task. We think this is related to the specific clauses employed in the experiment. These clauses are implications in the form $A_1 \land \ldots \land A_n \Rightarrow B$ which are always verified except for a single configuration of the input variables. In this cases, NGAUSS converts the clause using a single negative Gaussian, whereas MGAUSS and PGAUSS employ a larger mixture. The more compact representation has the advantage of limiting the number of local minima, thus yielding an easier optimization problem. In a general setting, it is possible to select the most compact representation for each clause, therefore using PGAUSS or NGAUSS for expressions that are false or true for most configurations, respectively.

5. CONCLUSIONS

This paper presents a novel approach to integrate logic prior knowledge in form of FOL clauses into graph regularization. This methodology is a first attempt to bridge pure transductive machine learning approaches to knowledge based annotators based on logic formalisms. In particular, the presented approach directly injects logic knowledge compiled as continuous constraints into a classical graph regularization schema. Different conversion techniques for the FOL clauses have been proposed and tested. The experimental results have been carried out on an image tagging dataset and show that, using a careful selection of the logic knowledge conversion schema, the presented approach outperforms an image annotator based on graph regularization by a significant margin in terms of tagging accuracy.

6. REFERENCES


