Collective Classification using Semantic Based Regularization

Claudio Saccà, Michelangelo Diligenti, Marco Gori
Dipartimento di Ingegneria dell'Informazione
University of Siena, Via Roma 54, Siena, Italy
Email: {claudiosacca, diligmic, marco}@dii.unisi.it

***Abstract*—Semantic Based Regularization (SBR) is a framework for injecting prior knowledge expressed as FOL clauses into a semi-supervised learning problem. The prior knowledge is converted into a set of continuous constraints, which are enforced during training. SBR employs the prior knowledge only at training time, hoping that the learning process is able to encode the knowledge via the training data into its parameters. This paper defines a collective classification approach employing the prior knowledge at test time, naturally reusing most of the mathematical apparatus developed for standard SBR. The experimental results show that the presented method outperforms state-of-the-art classification methods on multiple text categorization tasks.**

I. Introduction

Traditional supervised machine learning approaches learn classification functions from a set of examples in the form of a pair (pattern, desired output). It is generally assumed that patterns are independent and identically distributed (*i.i.d. assumption*). However, in many real-world applications, there are relational dependencies between the patterns that violates the i.i.d. assumption. Neglecting these dependencies will generally negatively affect the performances of the resulting classifier.

Statistical Relational Learning (SRL)[1] approaches instead exploit the dependencies among the instances. Collective classification (CC) [2], [3] is a SRL task attempting at simultaneously inferring the class labels of a set of interlinked test patterns. Several studies show the efficacy of CC over independent classification, especially for rich multi-label datasets [4]. Markov Logic Networks (MLNs) can combine first order logic with probabilistic reasoning[5] and are able to express complex rules about related objects, making them a natural candidate for CC tasks.

Semantic Based Regularization (SBR) [6] generalizes kernel machines to learn from examples and prior knowledge , which is assumed to be available in form of arbitrary First Order logic clauses. Semantic Based Regularization (SBR) converts the external knowledge into a set of real value constraints, which are enforced over the values assumed by the learned classification functions. SBR employs the prior knowledge only at training time, assuming that the learning process encodes the knowledge into the parameters via the training set. However, if the number of examples is small or if the feature representation of the patterns is poor, there is no guarantee that the prior knowledge will be respected by outputs on test patterns.

This paper significantly extends SBR by defining a novel collective classification methodology to be used to enforce the constraints also on the test data. One of the main advantages of the proposed method is that it is by no means limited in the type of relationships that can be modeled. Indeed, arbitrary relationships between patterns, labels, set of features and labels, etc. can be represented, as the full expressiveness of FOL is inherited from SBR. The experimental results show a clear improvement over state-of-the-art classification methods on various text categorization tasks.

II. Semantic Based Regularization

Let us consider a multitask learning problem, where each task works on an input domain where labeled and unlabeled examples are sampled from. Tasks can also be n-ary relations where the input is a tuple of patterns, each one from a specific domain. Each input pattern is described by a vector of features that are relevant to solve the tasks at hand. Let $\mathcal{D}_k$ and $f_k : \mathcal{D}_k \to I\!R$ be the input domain and the function implementing task k, respectively. We indicate as $\boldsymbol{x} \in \mathcal{D}_k$ a generic input vector for the k-th task. Task k is implemented by an function f_k, which may be known a priori for *evidence* tasks, or it must be inferred (*query* task). In this latter case it is assumed that each task function lives in an appropriate Reproducing Kernel Hilbert Space $\mathcal{H}_k$. Let us indicate with $\mathcal{T}$ the total number of tasks, of which the first T are assumed to be the query tasks. $\mathcal{L}_k$, $\mathcal{U}_k$ and $\mathcal{S}_k = \mathcal{L}_k \cup \mathcal{U}_k$, $k = 1, \ldots, \mathcal{T}$ be the available supervised, unsupervised and supervised plus unsupervised examples for task k. For evidence tasks it will hold that all sampled data is supervised as the output of the task is known in all data points.

The learning procedure can be cast as an optimization problem that aims at computing the optimal query functions $f_k \in \mathcal{H}_k : \mathcal{D}_k \to I\!R, k = 1, \ldots, T$, where $\mathcal{D}_k$ is the input domain for the k-th function. In the following, $\boldsymbol{f} = [f_1, \ldots, f_{\mathcal{T}}]'$ indicates the vector collecting all query and evidence functions. Tasks functions are correlated by a set of constraints that can be expressed by the functionals $\phi_h(\boldsymbol{f}, \mathcal{S}) = 0 \quad h = 1, \ldots, H$, where $\mathcal{S} = \cup_{k=1,\ldots,\mathcal{T}} \mathcal{S}_k$ is the set of data points where the constraints are evaluated. Let the parameters $\lambda_h > 0$ weight the contribution of each constraint, then objective function can be expressed as,

$$E_{emp}[\boldsymbol{f}] \;=\; \sum_{k=1}^{T} ||f_k||^2 + \sum_{h=1}^{H} \lambda_h \cdot \phi_h(\boldsymbol{f}, \mathcal{S}) \qquad (1)$$

As explained in the next sections, SBR can get advantage of the full expressiveness of First Order Logic to represent logic knowledge about the tasks via the constraints ϕ_h.

A. *Translation of first-order logic clauses into constraints*

Let's consider knowledge-based descriptions given by first-order logic (FOL), where $\mathcal{P}$ is a set of predicates used in the KB. The clauses will be built from the set of atoms $\mathcal{A}$ obtained by grounding the variables passed as input to the predicates.

With no loss of generality, we restrict our attention to FOL clauses in the *Prenex Normal Form* (PNF) form, where all the quantifiers $(\forall, \exists)$ and their associated quantified variables are placed at the beginning of the clause. The quantifier-free part of the expression is equivalent to an assertion in propositional logic for any given assignment of the quantified variables. Task functions $\boldsymbol{f}$ are exploited to implement the query predicates in $\mathcal{P}$ and each variable grounding corresponds to an input $\boldsymbol{x}$ represented via a set of real-valued features.

Since the function values are continuous, the clause is relaxed like in fuzzy logic by generalizing the atoms to assume values in the $[0,1]$ range. This converts the clauses into a set of constraints that can be enforced during the kernel based learning process. In particular, a clause is converted into a constraint in three steps. PREDICATE SUBSTITUTION: substitution of the k-predicate p_k with its continuous implementation realized by the function f_k. f_k is typically composed with a squash function, mapping the output values into the interval $[0,1]$ such that the value 0 is associated with *false* and 1 with *true*. In particular, the atom $a_i(\boldsymbol{v}_{a_i})$ is mapped to $\sigma(f_{k(i)}(\boldsymbol{v}_{a_i}))$, where $\sigma : I\!R \rightarrow [0,1]$ is a monotonically increasing squashing function. A natural choice for the squash function is the piecewise linear mapping $\sigma(y) = \min(1, \max(y, 0))$, this is indeed the function that was employed in the experimental results. CONVERSION OF THE PROPOSITIONAL EXPRESSION where all variables are grounded and their feature representations are passed to the real-valued function (see section II-B). QUANTIFIER CONVERSION: conversion of the quantifiers as shown in section II-C.

B. *Logic expressions and their continuous representation*

T-norms [7] are commonly used in fuzzy logic to generalize propositional logic expressions to real valued functions of continuos variables. A *t-norm fuzzy logic* is defined by its t-norm $t(a_1, a_2)$ that models the logic AND, while the negation of a variable is computed as $1 - a$. 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 into a continuous function. Many different t-norm logics have been proposed in the literature. For example, the *product t-norm* $t(a_1, a_2) = a_1 \cdot a_2$. Another commonly used t-norm is the *minimum t-norm* defined as $t(v_1, v_2) = \min(a_1, a_2)$.

The *residuum* of a t-norm generalizes modus ponens to fuzzy logic. Indeed, any t-norm there has a corresponding binary operator $\Rightarrow$ called *residuum*. In particular, for a minimum t-norm, it holds that:

$$(a_1 \Rightarrow a_2) \rightarrow \begin{cases} 1 & a_1 \leq a_2 \\ a_2 & a_1 > a_2 \end{cases}$$

The residuum allows to relax the condition of satisfaction for the implication: an implication is satisfied if the right end side of the implication is more verified than the pre-condition on the left side. Please note that the equivalence $a_1 \Rightarrow a_2 \equiv \neg a_1 \vee a_2$ (*modus ponens*) was used to represent implications in previous work in the SBR literature.

Therefore, given an expression $E(\mathcal{P}, \boldsymbol{v})$ where the variables $\boldsymbol{v}$ are grounded and set as input to the predicates $\mathcal{P}$ to form the atoms, it is possible to build a t-norm generalization $t_E(\boldsymbol{f}, \boldsymbol{x})$, where the predicates have been substituted by the corresponding functions and the grounded variables have been substituted by the corresponding feature-based representations. For example, $A(v) \vee B(v) \equiv \neg(\neg A(v) \wedge \neg B(v))$ is mapped to $1 - (1 - f_A(\boldsymbol{x}))(1 - f_B(\boldsymbol{x}))$ using a product t-norm.

C. *Quantifier conversion*

Let us consider a universally quantified FOL formula $\forall v \; E(\mathcal{P}, v)$. The universal quantifier expresses the fact that the expression must hold for any realization of the quantified variable. When considering the real–valued mapping $t_E(\boldsymbol{f}, \boldsymbol{x})$ of the original boolean expression where $\boldsymbol{x}$ is the feature-based representation of v, the universal quantifier can be converted measuring the degree of non-satisfaction of the expression over the domain $\mathcal{D}$ of $\boldsymbol{x}$. This measure can be implemented by computing the overall degree of violation, for example using the infinity norm:

$$\forall v \; E(\mathcal{P}, v) \quad \rightarrow \quad \phi(\boldsymbol{f}, \mathcal{S}) = \max_{\boldsymbol{x} \in \mathcal{D}} 1 - t_E(\boldsymbol{f}, \boldsymbol{x})$$

When multiple universally quantified variables are present, the conversion is performed recursively from the outer to the inner variable. For example, $\forall v_1 \ldots \forall v_n \; E(\mathcal{P}, v_1, \ldots, v_n)$: $\phi(\boldsymbol{f}, \mathcal{S}) = \max\limits_{\boldsymbol{x}_1 \in \mathcal{D}_1} \ldots \max\limits_{\boldsymbol{x}_n \in \mathcal{D}_n} 1 - t_E(\boldsymbol{f}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$. The existential quantifier can be implemented by enforcing the De Morgan law, yielding

$$\exists v \; E(\mathcal{P}, v) \quad \rightarrow \quad \phi(\boldsymbol{f}, \mathcal{S}) = \min_{\boldsymbol{x}} 1 - t_E(\boldsymbol{f}, \boldsymbol{x})$$

It is also possible to use a different norm to convert the universal or existential quantifier, for example using norm-1:

$$\forall v \; E(\mathcal{P}, v) \quad \rightarrow \quad \phi(\boldsymbol{f}, \mathcal{S}) = \sum_{\boldsymbol{x} \in \mathcal{D}} 1 - t_E(\boldsymbol{f}, \boldsymbol{x})$$

The description logic $\exists_n$ operator can be conveniently translated as the sum over the n-th min values over the set when using norm-1. We will use this operator in the experimental section. As a final example involving both quantifiers, the clause $\forall v_1 \exists v_2 \; A(v_1) \wedge B(v_2)$ is converted into the constraint: $\max\limits_{\boldsymbol{x}_1 \in \mathcal{S}_1} \min\limits_{\boldsymbol{x}_2 \in \mathcal{S}_2} f_A(\boldsymbol{x}_1) f_B(\boldsymbol{x}_2)$ where $\boldsymbol{x}_1, \boldsymbol{x}_2$ are the feature vectors representing the grounded variables v_1, v_2, respectively.

There is a duality between an universally quantified expression and its continuous generalization: the constraint resulting from the generalization is verified iff the the logic expression is verified as well. It is possible to extend the Representer Theorem for Kernel Machines [8], showing that the optimal solution of equation 1 can be expressed as a kernel expansion over the data points [6].

Therefore, assuming that the functions are kernel expansions, equation 1 becomes:

$$\sum_{k=1}^{T} \boldsymbol{w}_k' \boldsymbol{G}_k \boldsymbol{w}_k + \sum_{h=1}^{H} \lambda_h \cdot \phi_h(\boldsymbol{G}_1 \boldsymbol{w}_1, \ldots, \boldsymbol{G}_T \boldsymbol{w}_T, \boldsymbol{f}_{T+1}, \boldsymbol{f}_{\mathcal{T}}) \; ,$$

where $\boldsymbol{G}_k$, $\boldsymbol{w}_k$, $\boldsymbol{f}_k = \boldsymbol{G}_k \boldsymbol{w}_k$ and $\boldsymbol{y}_k$ are the the gram matrix, the weights, the function values and the desired output column vectors for the patterns in the domain of the k-th task. Evidence tasks do not need to be approximated as they are fully known.

This equation can be optimized via gradient descent, where the gradient of the constraints is computed using the chain rule:

$$\frac{\partial \phi_h}{\partial \boldsymbol{w}_k} = \frac{\partial \phi_h}{\partial \boldsymbol{f}_k} \cdot \frac{\partial \boldsymbol{f}_k}{\partial \boldsymbol{w}_k} \ . \tag{2}$$

D. Supervised data

In a classical supervised setting, a set of supervised examples $\mathcal{L}_k \in \mathcal{S}_k$ is available for the k-th predicate that must be learned. In SBR, supervised data can be expressed using the following clause: $\forall v \ S_k(v) \Rightarrow p_k(v), \quad \forall v \ N_k(v) \Rightarrow \neg p_k(v)$ where $S_k(v)$ and $N_k(v)$ are evidence predicates that hold true iff v is a positive or negative example for the query predicate p_k, respectively. Using the norm-1 conversion of the universal quantifier and adding also the negative examples, this results in:

$$\phi(\mathcal{L}_k, f_k) = \sum_{\boldsymbol{x}^j \in \mathcal{L}_k} h(f_k(\boldsymbol{x}^j), y_k^j)$$

where $h(\cdot)$ is the hinge function and y_k^j is the target for the j-th example of function k. This is the term of the cost function used for the fitting of supervised data in SVMs.

III. Collective classification for SBR

Let us assume that the function parameters have been learned during the training phase using examples and constraints deriving from logic knowledge. When a test set is provided, which is composed by a set of data points for each function domain, together with the prior knowledge expressed by FOL. The continuous constraints can be built using the same procedure as described before, yielding the constraints grounded over the test data $\mathcal{S}'$: $\phi_h(\mathcal{S}', \boldsymbol{f}) = 0, \quad h = 1, \ldots, H$. Let $\boldsymbol{f}_k$ indicate the vector of values obtained by evaluating f_k over the data points of the test set $\mathcal{S}'_k$ belonging to the domain $\mathcal{D}_k$. Collective classification can been formulated as a minimization problem by searching for a new set of $\boldsymbol{f}'_k \quad k = 1, \ldots, T$ that are close to the learned function values $\boldsymbol{f}_k$, while respecting the constraints on the test data. This means that the learned function values will be used as a prior for a refinement of the values on the test data using the constraints. In particular, the following optimization problem is considered:

$$\underset{\boldsymbol{f}'_1, \ldots, \boldsymbol{f}'_T}{\operatorname{argmin}} \ \sum_{k=1}^{T} L_k^{cc}\left(\boldsymbol{f}'_k, \boldsymbol{f}_k\right) + \sum_{h=1}^{H'} \lambda_h^{cc} \cdot \phi_h(\boldsymbol{f}'_1, \ldots, \boldsymbol{f}'_T)$$

where each λ_h^{cc} is a parameter weighting the contribution of the h-th constraint and balancing the trade off between the minimization of the regularization term, which penalizes function values moving away from the learnt values using the loss L_k^{cc}, and the second term which penalizes solutions not respecting the constraints.

The collective classification objective function defined can be optimized using gradient descent by computing the derivative with respect to the vector of functions $\boldsymbol{f}'_k, k$ for each query predicate $k = 1, \ldots, T$. SBR collective classification reuses the same schema to compute the gradients of the constraints as shown in equation 2. However, while in the training phase the gradient were computed with respect to the functions weights $(\frac{\partial \phi_h}{\partial \boldsymbol{f}'_k} \cdot \frac{\partial \boldsymbol{f}'_k}{\partial \boldsymbol{w}_k})$, here only the first term of the derivative chain rule has to be taken into account. Indeed, since the weights are now fixed, no back-propagation of the derivative of the error down to the weights is needed. This provides a very elegant solution to collective classification as there is no additional complexity in the implementation. For example, our software simulator reuses the same routine for optimizing the constraints in model training and collective classification.

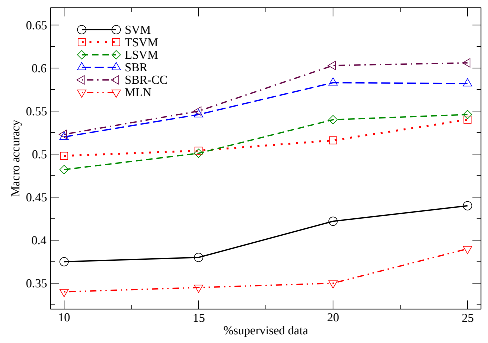

Fig. 1. Macro accuracy on the AAN Movies dataset (average over 5 runs).

IV. Experimental results

For all experiments, the available supervised data for each task has been included adding the clauses described in section II-D. Since this is common to all experiments, we will not explicitly list this knowledge.

AAN dataset. This dataset[1] contains 380 scientific publications manually classified into three research areas. The AAN dataset is a relational dataset as the papers are linked via citations. Each paper is associated to a title represented as bag-of-words. The citation information has been translated into a set of clauses (one per category C_i), stating that papers tend to cite other papers about the same category: $\forall x \forall y \ Cite(x, y) \Rightarrow (C_i(x) \wedge C_i(y)) \vee (\neg C_i(x) \wedge \neg C_i(y))$ where $Cite(x, y)$ is a binary predicate which holds true iff paper x cites paper y.

The fact that classes are mutually exclusive is expressed by the following clause: $\forall x \in \mathcal{P} \quad C_1(x) \oplus C_2(x) \oplus C_3(x)$, Five folds have been generated by randomly selecting a given percentage of the papers for which supervisions are kept as training set. Since classes are strongly unbalanced and mutually exclusive, macro accuracy has been selected as target metric for this task. Results obtained by collective classification for Semantic Based Regularization (SBR-CC) has been compared against standard SVM (using only supervised labels), Transductive SVM (TSVM), Laplacian SVM (LSVM) using the citations to build the manifold of data, standard semantic-based regularization (SBR) and Markov Logic Networks performing

[1]http://clair.si.umich.edu/homepage/downloads/aan_relational/

	10%	15%	20%	25%	30%	35%	40%
SVM	0.241	0.289	0.325	0.362	0.394	0.416	0.428
SBR	0.407	0.447	0.465	0.469	0.478	0.485	0.487
SBR-CC	0.428	0.482	**0.501**	**0.509**	**0.512**	**0.523**	**0.527**
TSVM	0.362	0.407	0.428	0.441	0.457	0.466	0.474
LSVM	0.366	0.395	0.413	0.426	0.436	0.443	0.451

TABLE I. F1 METRIC ON THE CORA DATASET FOR DIFFERENT NUMBERS OF SUPERVISED PATTERNS.

collective classification in their *infer* phase. Markov Logic Networks have been built by adding the same logic knowledge used by SBR. Since MLNs do not directly deal with patterns represented as feature vectors, the bag-of-words have been transformed as suggested by the official Alchemy tutorial for tackling text categorization tasks [9]. In particular, the bag-of-word for a document d is converted into a set of grounding $HasWord(w, d)$ for each word w in the bag-of-word and then by adding the rule $HasWord(w, d) \Rightarrow Topic(c, d)$ for each word appearing in a document d of class c. All classifiers have been trained using different meta-parameters, and the best model has been selected by cross validation on the validation set. Figure 1 reports the classification results as an average over 10 runs. SBR with collective classification (SBR-CC) improves SBR for all number of training patterns. SBR outperforms TSVM and LSVM for all tested configuration, while SVMs and MLN are the worse performers for this task.

CORA dataset. Collective classification for SBR has been evaluated over the CORA research paper dataset [10]. The top 30 categories by the highest number of papers have been selected. 5000 papers belonging to one of these classes have been then randomly sampled and divided in three sets composed by $60\%, 10\%, 30\%$ of the patterns for the training, validation and test set, respectively. Each paper has been associated to a vectorial representation containing its title represented as bag-of-words. For both dataset, five folds have been generated by selecting $n\%$ of the papers for which supervisions are kept as training set.

The following logic knowledge has been used for the task: TAXONOMY HIERARCHY expressed in FOL as a set of clauses stating that if the predicate associated to a leaf node is true, then all the predicates associated to the nodes up to the root should be true as well. For example, the following rules have been added $\forall x \in \mathcal{P}\ NN(x) \Rightarrow ML(x)$, $\forall x \in \mathcal{P}\ ML(x) \Rightarrow AI(x)$. CORA CITATIONS expressed by a set of clauses (one per category C_i), stating that papers tend to cite other papers about the same category (similarly to what done for the AAN dataset, see above for details). TRANSDUCTIVE SVMS [11] correspond to a special case of the proposed framework, where a logic clause imposes that any predicate should be either true or false. Please note that while this rule is always verified in standard logic, it is not verified in fuzzy logics. Therefore for each category C_i, the following rule can be expressed: $\forall x \in \mathcal{P}\ \ C_i(x) \vee \neg C_i(x)$ to force the function estimating predicate C_i to assume values that are away from the separating surface even on unsupervised data. As typically done in Transductive SVMs, it is possible to avoid unbalanced solutions by imposing that $\exists_n x \in \mathcal{P}\ C_i(x) \wedge \exists_m\ x \in \mathcal{P}\ \neg C_i(x)\ :\ n+m = N$ where N is the total number of patterns and n and m are estimated from the supervised examples: $n = N \cdot n_{pos}^{C_i} / (n_{pos}^{C_i} + n_{neg}^{C_i})$ where $n_{pos}^{C_i}$ and $n_{neg}^{C_i}$ are the number of positive and negative supervised examples for predicate C_i, respectively. EXTERNAL SEMANTIC KNOWLEDGE can be inserted as well using common knowledge about the environment. For example, let $HasWord$ be an evidence predicate holding true iff document x has word "neural", it is possible to associate the presence of the term to either the category Artificial Intelligence or Biology as: $\forall\ x \in \mathcal{P}\ HasWord(x, Neural) \Rightarrow AI(x) \vee BIO(x)$ where AI, BIO indicate the predicate for Artificial Intelligence, and biology, respectively.

Results obtained by collective classification for Semantic Based Regularization (SBR-CC) has been compared against standard SVM, TSVM, LSVM using the citations to build the manifold of data and standard semantic-based regularization (SBR) not enforcing the logic knowledge on the test data. For all classifiers, the meta-parameters have been selected on the validation set and evaluated on the test set. Table I report the results as an average over 5 runs. SVM-CC over-performs in terms of F1 all the other tested methods for all configurations. Metrics in bold represent statistically significant gains (95% t-test) over all the other classifiers.

V. CONCLUSIONS

This paper presented a collective classification approach for Semantic Based Regularization. This approach can get advantage on the test data of tight integration between the kernel machines working on the low leve feature vectors and the high level knowledge, which can be represented with the full expressiveness of FOL. The experimental results show significant gains on multiple text categorization tasks.

Acknowledgments. This research was partially supported by the research grant PRIN2009 Learning Techniques in Relational Domains and Their Applications (2009LNP494) from the Italian MURST.

REFERENCES

[1] L. Getoor and B. Taskar, *Introduction to statistical relational learning*. MIT press, 2007.

[2] D. Jensen, J. Neville, and B. Gallagher, "Why collective inference improves relational classification," in *In Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004, pp. 593–598.

[3] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI magazine*, vol. 29, no. 3, p. 93, 2008.

[4] N. Ghamrawi and A. McCallum, "Collective multi-label classification," in *Proceedings of the 14th ACM international conference on Information and knowledge management (CIKM)*. ACM, 2005, pp. 195–200.

[5] M. Richardson and P. Domingos, "Markov logic networks," *Machine Learning*, vol. 62, no. 1–2, pp. 107–136, 2006.

[6] M. Diligenti, M. Gori, M. Maggini, and L. Rigutini, "Bridging logic and kernel machines," *Machine Learning*, pp. 1–32, 2011.

[7] E. Klement, R. Mesiar, and E. Pap, *Triangular Norms*. Kluwer Academic Publisher, 2000.

[8] B. Scholkopf and A. J. Smola, *Learning with Kernels*. Cambridge, MA, USA: MIT Press, 2001.

[9] P. Domingos, "The alchemy tutorial," http://alchemy.cs.washington.edu/tutorial/tutorial.pdf, 2010, [Online; accessed April 2013].

[10] A. McCallum, K. Nigam, J. Rennie, and K. Seymore, "Automating the construction of internet portals with machine learning," *Information Retrieval Journal*, vol. 3, pp. 127–163, 2000, www.research.whizbang.com/data.

[11] V. N. Vapnik, *Statistical learning theory*. Wiley, NY, 1998.