A Multiple Labeling-based Optimum-Path Forest for Video Content Classification

Luís A. M. Pereira, J. Paulo Papa
Department of Computing
São Paulo State University
Bauru, Brazil
{luis.pereira,papa}@fc.unesp.br

Jurandy Almeida, Ricardo da S. Torres
Institute of Computing
University of Campinas
Campinas, Brazil
{jurandy.almeida,rtorres}@ic.unicamp.br

Willian Paraguassu Amorim
Institute of Computing
Federal University of Mato Grosso do Sul
Campo Grande, Brazil
paraguassuec@gmail.com

***Abstract*—Multiple-labeling classification approaches attempt to handle applications that associate more than one label to a given sample. Since we have an increasing number of systems that are guided by such assumption, in this paper we have presented a multiple-labeling approach for the Optimum-Path Forest (OPF) classifier based on the problem transformation method. In order to validate our proposal, a multi-labeled video classification dataset has been used to compare OPF against three other classifiers and another variant of the OPF classifier based on a k-neighborhood. The results have shown the validity of the OPF-based classifiers for multi-labeling classification problems.**

***Keywords*-Image motion analysis, Video signal classification, multi-label learning, Optimum-Path Forest**

I. Introduction

Traditional pattern recognition techniques employ a training set S in order to learn a function $h \in \mathcal{H}$ that maps a given feature vector $s \in \mathcal{S}$ to a label $l \in \mathcal{L}$, obtained from a set of labels $\mathcal{L}$ [1]. In this context, we have $h : \mathcal{S} \rightarrow \mathcal{L}$, where $\mathcal{H}$ is the set of all functions. If we have $|\mathcal{L}| = 2$, the learning problem is often referred as a binary classification problem, while we have a multi-class problem when $|\mathcal{L}| > 2$.

In the context of multi-label classification, each sample s can be associated with a label set $\mathcal{L}' \subseteq \mathcal{L}$. One of the main reasons for using techniques that support multiple labels concerns with applications that associate with a given sample more than one label, e.g., systems to aid medical diagnosis and text categorization. In the first case, a patient may be affected by more than one disease, while in the case of categorizing documents, a newspaper article, for instance, may be categorized, at the same time, as belonging with two different areas (e.g., religion and arts).

Currently, new applications that require methods with support to multiple labels have grown widely. Ogihara and Li [2], for example, employed such approaches to categorize songs. Boutell et al. [3] performed semantic classification of scenes, since a picture can be labeled as being a beach and also can have buildings at the same time (city). The idea of their work is to reduce the problem of learning with multiple labels in various sorts of binary problems, where each image i was associated with a set $\mathcal{B}_i$, such that $|\mathcal{B}_i| = |\mathcal{L}|$. In this case, each element $b_j \in \mathcal{B}_i$ had the value 1 if the image was associated with a class j, and 0 otherwise.

McCallum [4] proposed a Bayesian approach to the problem of multi-label document classification, where a probabilistic mixture model was assumed to generate each document, and an Expectation-Maximization [5] strategy was used to learn the mixture weights and word distribution in each one of them. Additionally, Zhou and Zhang [6] adapted the algorithm of the k-nearest neighbour classification in the context of multi-labeling, which is called ML-kNN (Multi-label-k-NN).

However, in many cases, the employed learning mechanism produces a ranking function $f' : \mathcal{S} \times \mathcal{L} \rightarrow \mathbb{R}$, such that, for a given instance $s \in \mathcal{S}$, the label set in $\mathcal{L}$ should be ordered according to $f'(x, \cdot)$. Thus, a label l_1 is considered better ranked than another label l_2 if $f'(x, r_1) > f'(x, r_2)$. In this fashion, a ranking of labels requires post processing in order to provide a set of labels with the highest score according to some loss function. When the labels in a dataset belong to a hierarchical structure, i.e., the set of labels for a given sample can be represented as a tree, where each node indicates a possible class, then we have a multi-label hierarchical classification task. Jin and Ghahramani [7] define a problem of multiple labels as an unsupervised classification problem, where each instance is associated with more than one class, but only one of them is the true class.

In this work, we propose to evaluate the Optimum-Path Forest (OPF) classifier [8], [9], [10] for multi-label learning tasks, since OPF has never been studied in this context. OPF is a graph-based approach widely used in several applications [8], [9], [10]. The use of OPF is motivated by its fast training and classification procedures, as well its good recognition rates. Another point is that OPF can easily handle different distance metrics, which is very important in the context of classification tasks.

In our study concerning the use of OPF in multi-label tasks, we have employed a public multi-label video dataset composed of human actions and scenes, and two different video descriptors were used to extract the video contents. We also used two OPF variants for comparison purposes together with a decision three, a Bayesian classifier and a Support Vector Machines (SVM) classifier. It is also important to highlight that OPF has never been applied for video content classification.

The remainder of the paper is organized as follows. In

Section II and Section III we revisit the Optimum-Path Forest theory background. Section IV and V introduce the video content dataset and the video descriptors employed in this work, respectively. Experimental results are discussed in Section VI. Finally, conclusions are stated in Section VII.

II. Multi-label Classification

Multi-class classification aims to associate a sample $s_i \in \mathcal{S} = \{s_1, ...s_m\}$ with a single label $l_j \in \mathcal{L} = \{l_1, ..., l_n\}$, for $m = |\mathcal{S}|$ and $n = |\mathcal{L}|$. Unlike, the multi-label classification task may associate s_i with a set of labels $\mathcal{L}' \subseteq \mathcal{L}$. Mathematically, a multi-label classifier aims to build a function $h : s_i \rightarrow \mathcal{L}'_i$. In this context, multi-label classification algorithms can be categorized into two different groups [1]: i) *problem transformation methods*, and ii) *algorithm adaptation methods*.

Problem transformation methods handle with a multi-label problem by transforming it into one or more single-label classification task. Binary Relevance (BR) is a popular transformation method that maps the original multi-labeled dataset into $|\mathcal{L}|$ binary classification problems. However, BR suffers from not considering the dependency between the labels. Label Powerset (LP) is also a transformation method, which considers a set of labels $\mathcal{L}'$ as a unique label, allowing the dataset to be handled as multi-classes problem. As such, consider a set of labels as a unique class, LP takes into account the dependency between the labels. However, LP may build a large amount of classes with a small number of samples by each class, which may increase the classification complexity.

Algorithm adaptation methods extend and customize existing machine learning algorithm for the multi-label task [11]. Boosting, k-nearest neighbors, decision trees and neural networks are examples of machine learning algorithms extended to multi-label domain. The most famous adaptation method is the Multi-label k-Nearest Neighbours [6], which is derived from the traditional k-NN algorithm and uses maximum a posteriori (MAP) principle to determine the label set for a testing instance.

A. Multi-label evaluation measures

In order to evaluate experiments using multi-label classification methods, the reader can face several different and contrastive measures [11]. In what follows, we introduce some of the most common evaluation measures, which are used in our experiments. The following formulations briefly describe the measures:

$$\textit{Hamming loss}(h) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{|\mathcal{L}|}\left|h(s_i)\Delta\mathcal{L}'_i\right|, \quad (1)$$

where Δ denotes the symmetric difference between two set of labels. Thus, the lower hamming loss the higher classifier's efficacy. For the following measures, greater values indicate better performance:

$$Accuracy(h) = \frac{1}{m}\sum_{i=1}^{m}\frac{|h(s_i)\cap\mathcal{L}'_i|}{|h(s_i)\cup\mathcal{L}'_i|}, \quad (2)$$

$$Precision(h) = \frac{1}{m}\sum_{i=1}^{m}\frac{|h(s_i)\cap\mathcal{L}'_i|}{|\mathcal{L}'_i|}, \quad (3)$$

$$Recall(h) = \frac{1}{m}\sum_{i=1}^{m}\frac{|h(s_i)\cap\mathcal{L}'_i|}{|h(s_i)|} \quad (4)$$

$$\textit{F-measure}(h) = \frac{1}{m}\sum_{i=1}^{m}2\times\frac{|h(s_i)\cap\mathcal{L}'_i|}{|h(s_i)|+|\mathcal{L}'_i|}, \quad (5)$$

and

$$\textit{Subset accuracy}(h) = \frac{1}{m}\sum_{i=1}^{m}I(h(s_i)=\mathcal{L}'_i), \quad (6)$$

where, $I(true) = 1$ e $I(false) = 0$. This is a very strict evaluation measure as it requires an exact match of the predicted and true set of labels.

III. Optimum-Path Forest classifiers

The OPF classifier works by modeling the problem of pattern recognition as a graph partition in a given feature space, where the graph nodes are represented by feature vectors. The partition of the graph is carried out by a competition process between some key samples (prototypes), which offer optimum paths to the remaining nodes of the graph. Each prototype sample defines its optimum-path tree (OPT), and the collection of all OPTs defines an optimum-path forest, which gives the name to the classifier [8], [9].

The OPF can be seen as a generalization of the well-known Dijkstra's algorithm to compute optimum paths from a source node to the remaining ones [12]. The main difference relies on the fact that OPF uses a set of source nodes (prototypes) with any smooth path-cost function [13]. In case of Dijkstra's algorithm, a function that summed the arc-weights along a path was applied.

Let $Z = Z_1 \cup Z_2 \cup Z_3 \cup Z_4$ be a dataset labeled with a function λ, in which Z_1, Z_2, Z_3 and Z_4 are, respectively, a training, learning, evaluating, and test sets. Let $S \subseteq Z_1$ a set of prototype samples. Essentially, the OPF classifier creates a discrete optimal partition of the feature space such that any sample $s \in Z_2 \cup Z_3 \cup Z_4$ can be classified according to this partition. This partition is an optimum path forest (OPF) computed in $\Re^n$ by the Image Foresting Transform (IFT) algorithm [13]. The OPF algorithm works with a training and a testing phase. In the former step, the competition process begins with the prototypes computation. The OPF algorithm may be used with any *smooth* path-cost function which can group samples with similar properties [13]. Next, we introduce two examples of path cost functions employed on the OPF supervised classifiers with complete and k-NN graph, respectively:

$$\begin{aligned} f_{\max}(\langle s\rangle) &= \begin{cases} 0 & \text{if } s \in S, \\ +\infty & \text{otherwise} \end{cases} \\ f_{\max}(\pi\cdot\langle s,t\rangle) &= \max\{f_{\max}(\pi), d(s,t)\} \end{aligned} \quad (7)$$

and

$$f_{\min}(\langle s \rangle) = \begin{cases} +\infty & \text{if } s \in S, \\ 0 & \text{otherwise} \end{cases}$$
$$f_{\min}(\pi \cdot \langle s,t \rangle) = \min\{f_{\min}(\pi), d(s,t)\}, \quad (8)$$

in which $d(s,t)$ means the distance between samples s and t, and a path π is defined as a sequence of adjacent samples. In such a way, we have that $f_{\max}(\pi)$ computes the maximum distance between adjacent samples in π, when π is not a trivial path; and $f_{\min}(\pi)$ computes the minimum distance between adjacent samples in π, when π is not a trivial path.

A. OPF with complete graph

We are interested in finding the elements that fall on the boundary of the classes with different labels. For that purpose, we can compute a Minimum Spanning Tree (MST) over the original graph (Figure 1a) and then mark as prototypes the connected elements with different labels. Figure 1b displays the MST with the prototypes at the boundary. After that, we can begin the competition process between prototypes in order to build the optimum-path forest, as displayed in Figure 1c. The classification phase is conducted by taking a sample from the test set (orange diamond in Figure 1d) and connecting it to all training samples. The distance to all training nodes are computed and used to weight the edges. Finally, each training node offers to the test sample a cost given by a path-cost function (maximum arc-weight along a path - Equation 7), and the training node that has offered the minimum path-cost will conquer the test sample. This procedure is shown in Figure 1e.

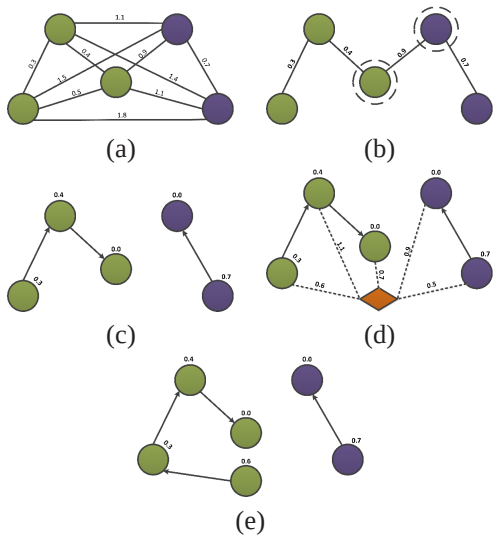

Fig. 1. OPF pipeline: (a) complete graph, (b) MST and prototypes bounded, (c) optimum-path forest generated at the final of training step, (d) classification process and (e) the orange diamond sample is associated to the green circle class. The values above the nodes are their costs after training, and the values above the edges stand for the distance between their corresponding nodes.

B. OPF with k-NN graph

The OPF with k-NN graph (OPFkNN) also models the training samples as graph nodes. However, it makes use of the k-nearest neighborhood as adjacency relation, and both arcs and nodes are weighted [10]. The basic difference between OPFkNN and the standard OPF is the fact the latter estimates the prototypes at the boundaries of the classes, while OPFkNN estimates the prototypes on the regions with high concentration of samples. To fulfill this task, a probability density function is used to estimate the density of each sample. Figure 2a displays a k-NN optimum-path forest with two prototypes (bounded nodes). For the classification phase, the probability density of a testing sample is computed, and then the testing sample is connected with its k nearest neighborhood (Figure 2b). The training node that offered the maximum path-cost will conquer the test sample, as shown in Figure 2c.

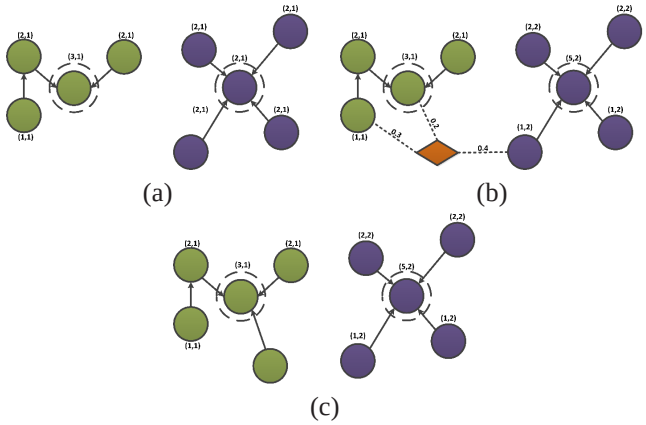

Fig. 2. OPFkNN pipeline: (a) optimum-path forest generated at the final of training step, in which the bounded nodes are the prototypes, i.e. the nodes with the maximum densities (b) classification process (c) the orange diamond sample is associated to the green circle class. The values (x, y) above the nodes are their density value and class label, respectively.

IV. Dataset description

In this work, we use a benchmarking dataset, namely HOLLYWOOD-2 [1], composed of video clips from 69 different movies. The dataset is divided into image videos with 12 classes of human actions and 10 classes of scenes distributed over 3669 video clips and approximately 20.1 hours of video in total. It contains approximately 150 samples per action class and 130 samples per scene class in training and test subsets [14]. Figure 3 shows some samples from HOLLYWOOD-2 dataset.

Action samples were collected by means of automatic script-to-video alignment in combination with text-based script classification [16]. Video samples generated from training movies correspond to the automatic training (AUTO-TRAIN) subset with noisy action labels. Based on this subset a clean training (CLEAN-TRAIN) subset is constructed with action labels manually verified to be correct. The test subset is also composed of action labels manually checked.

Scene classes are selected automatically from scripts such as to maximize co-occurrence with the given action classes and to capture action context as reported in [14]. Scene video

[1] http://www.di.ens.fr/~laptev/actions/hollywood2/

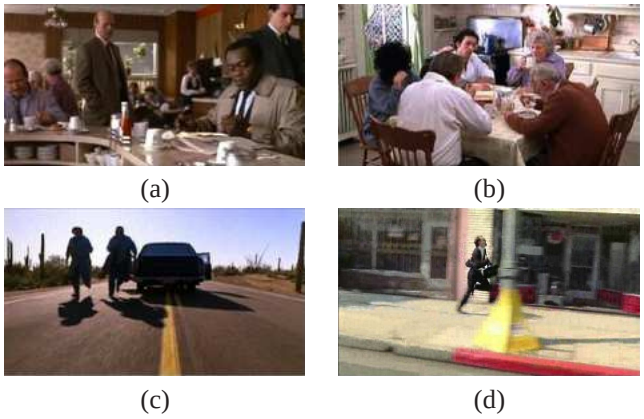

Fig. 3. Examples of samples from HOLLYWOOD-2 dataset: (a) eating-coffee, (b) eating-kitchen, (c) running-road and (d) running-street [15].

samples are then generated using script-to-video alignment. The labels of test scene samples are manually verified to be correct.

Finally, another interesting characteristic of this dataset is that samples may contain instances of several actions. For instance, a sample may belong to both kissing and hugging classes, which make HOLLYWOOD-2 handleable by methods with multi-label support. Tables I and II provide information about each subset, as well as the distribution of the classes.

TABLE I
ACTION DATASET DESCRIPTION.

Feature	# Training subset (clean)	# Training subset (automatic)	# Test subset clean
AnswerPhone	66	59	64
DriveCar	40	44	33
Eat	54	33	70
FightPerson	51	40	57
GetOutCar	32	38	45
HandShake	64	27	66
HugPerson	135	187	103
Kiss	104	87	108
Run	24	26	37
SitDown	132	133	146
SitUp	823	810	884
StandUp	66	59	64
Total samples	823	810	884

V. VIDEO REPRESENTATION

To encode visual properties from the video content, we have used two main approaches. One encodes local spatio-temporal features and is based on the *bag-of-features* approach [14]. The other approach specifically encodes motion information by using *histogram of motion patterns* [17].

A. Bag-of-Features (BoF)

Following previous works on action and scene recognition, we built a Bag-of-Features (BoF) model upon local space-time features, as described in [14]. For that, we extracted local features using the on-line implementation[2] of Spatio-temporal Interest Points (STIP) [18] combined with HOG/HOF descriptors [16].

In the BoF framework, visual words [19] are obtained by quantizing local feature descriptors according to a pre-learned dictionary. Thus, a video sequence is represented as a normalized frequency histogram of visual words associated with each local feature. In this work, we construct a visual dictionary using K-Means with K = 4000 visual words, as suggested in [14], [16].

TABLE II
SCENES DATASET DESCRIPTION.

Feature	# Training subset (automatic)	# Test subset clean
EXT-House	81	140
EXT-Road	81	114
INT-Bedroom	67	69
INT-Car	44	68
INT-Hotel	59	37
INT-Kitchen	38	24
INT-LivingRoom	30	51
INT-Office	114	110
INT-Restaurant	44	36
INT-Shop	47	28
Total samples	570	582

B. Histogram of Motion Patterns (HMP)

Besides encoding visual properties using a bag-of-features model, we also adopted a simple and fast algorithm to compare video sequences described in [17]. It consists of three main steps: (1) partial decoding; (2) feature extraction; and (3) signature generation.

For each frame of an input video, motion features are extracted from the video stream. For that, 2×2 ordinal matrices are obtained by ranking the intensity values of the four luminance (Y) blocks of each macroblock. This strategy is employed for computing both the spatial feature of the 4-blocks of a macroblock and the temporal feature of corresponding blocks in three frames (previous, current, and next). Each possible combination of the ordinal measures is treated as an individual pattern of 16-bits (i.e., 2-bits for each element of the ordinal matrices). Finally, the spatio-temporal pattern of all the macroblocks of the video sequence are accumulated to form a normalized histogram. For a detailed discussion of this procedure, refer to [17].

VI. EXPERIMENTAL RESULTS

In this section we present the experiments to evaluate the robustness of OPF-based classifiers in multi-label tasks.

A. Classification methods

In order to evaluate the performance of OPF-based classifiers, we compare them against with three different base single-label classifiers, being them implemented in Weka Java Framework [3]. We chose the J48 classifier, which is a decision tree learning algorithm and the well-known Naïve Bayes (NB)

[2]http://www.di.ens.fr/~laptev/download.html

[3]http://www.cs.waikato.ac.nz/ml/weka/

TABLE III
PERFORMANCE OVER THE ACTIONS DATA WITH AUTO-TRAIN SUBSET.

Evaluation	J48		NB		OPF		OPFkNN		SMO	
	BR	LP	BR	LP	BR	LP	BR	LP	BR	LP
Accuracy	0.1037	0.1162	0.1312	0.1482	0.1586	0.1642	**0.1604**	0.1604	0.1246	**0.2511**
F2-score	0.1244	0.1241	0.1581	0.1516	**0.1716**	0.1736	0.1696	0.1696	0.1339	**0.2581**
Hamming Loss	0.1593	0.1631	0.1794	0.1486	0.1628	0.1522	0.1524	0.1524	**0.1194**	**0.1304**
Precision	0.1113	0.1243	0.1387	0.1584	0.1714	0.1778	**0.1735**	0.1735	0.1339	**0.2698**
Recall	0.1652	0.1318	**0.2206**	0.1482	0.1844	0.1787	0.1742	0.1742	0.1431	**0.2540**
Subset Accuracy	0.0509	0.0950	0.0679	0.1380	0.1210	0.1369	**0.1335**	0.1335	0.0973	**0.2308**

TABLE IV
PERFORMANCE OVER THE ACTIONS DATA WITH CLEAN-TRAIN SUBSET.

Evaluation	J48		NB		OPF		OPFkNN		SMO	
	BR	LP	BR	LP	BR	LP	BR	LP	BR	LP
Accuracy	0.1315	0.1296	0.1460	0.2521	0.2175	0.2320	**0.2289**	0.2289	0.1833	**0.3273**
F2-score	0.1518	0.1364	0.1703	0.2566	**0.2392**	0.2488	0.2385	0.2385	0.1923	**0.3362**
Hamming Loss	0.1414	0.1580	0.1316	0.1308	0.1554	0.1439	0.1375	0.1375	**0.0916**	**0.1179**
Precision	0.1408	0.1399	0.1502	0.2653	0.2310	0.2475	**0.2459**	0.2459	0.1934	**0.3473**
Recall	0.1861	0.1401	0.2285	0.2526	**0.2713**	0.2668	0.2410	0.2410	0.2002	**0.3337**
Subset Accuracy	0.0769	0.1109	0.0871	0.2387	0.1595	0.1855	**0.2014**	0.2014	0.1561	**0.3009**

TABLE V
PERFORMANCE OVER THE SCENES DATA WITH AUTO-TRAIN SUBSET.

Evaluation	J48		NB		OPF		OPFkNN		SMO	
	BR	LP	BR	LP	BR	LP	BR	LP	BR	LP
Accuracy	0.1278	0.1690	0.2082	0.2431	0.2148	0.2242	**0.2182**	0.2182	0.0972	**0.2706**
F2-score	0.1507	0.1813	**0.2439**	0.2560	0.2415	0.2360	0.2314	0.2314	0.1089	**0.2847**
Hamming Loss	0.1787	0.1804	0.1753	0.1600	0.1885	0.1680	0.1662	0.1662	**0.1244**	**0.1538**
Precision	0.1436	0.2010	0.2249	0.2818	0.2413	**0.2552**	0.2552	0.2552	0.1148	**0.3127**
Recall	0.1787	0.1735	**0.3153**	0.2431	0.2698	0.2285	0.2208	0.2208	0.1134	**0.2706**
Subset Accuracy	0.0670	0.1323	0.1168	0.2045	0.1409	0.1890	**0.1787**	0.1787	0.0636	**0.2285**

and the support vector machines SMO learning algorithms. Those classifiers are used in most of the works that address multi-label classification tasks with transformation methods. In order to deal with multi-label, we use two transformation methods: Binary Relevance (BR) and Label Powerset (LP). Both methods are implemented in Mulan Java library for multi-label learning[4].

B. Experiments with HMP descriptor

Table III displays the classifier performances over the Actions dataset with AUTO-TRAIN as training set. Considering the LP method, we notice that the SMO classifier achieved the best results for all measures. OPF and OPFkNN achieved the second and third best results, respectively. NB performed slightly better than OPF classifiers considering the Hamming Loss and Subset Accuracy measures. The J48 classifier obtained the lowest results, which can be evidenced specially by the low rate according to Subset Accuracy. For the BR method, the OPF classifiers were the best performers in almost all measures, being them more balanced considering all measures. Although SMO achieved a low Hamming Loss rate, we can note it performed unsatisfactorily according to the remaining measures, performing better only than J48, which was the worst classifier again. NB achieved the highest recall rate, but did not perform well, specially according to Hamming Loss.

From Table IV, we can see the classifiers' results over the Actions dataset, but now with CLEAN-TRAIN as training set. As expected, the CLEAN-TRAIN set provided better classification results than AUTO-TRAIN set. The classifiers presented the same behavior observed over AUTO-TRAIN, but the classification rates were improved for all classifiers. SMO was the best combined to LP method, and the OPF-based classifers were the best performers considering the BR method.

Table V displays the classifiers' performances over the Scenes dataset. As we can note, SMO with LP achieved the best classification rates according all measures, and NB was the second best performer. The OPF-based classifiers were the third best ones, followed by J48. For BR method, OPF and OPFkNN achieved the best classification rates, respectively. NB were the second, with the highest values according to F2-score and Recall. We can see that both SMO and J48 had difficulties to dealing with the binary classifications provided by BR, since they achieved low classification rates, specially SMO, which was the worst performer.

[4]http://mulan.sourceforge.net/

TABLE VI
PERFORMANCE OVER THE ACTIONS DATA WITH AUTO-TRAIN SUBSET.

Evaluation	J48		NB		OPF		OPFkNN		SMO	
	BR	LP	BR	LP	BR	LP	BR	LP	BR	LP
Accuracy	0.1180	0.1713	**0.2112**	**0.3113**	0.1708	0.1719	0.1557	0.1557	0.2099	0.3026
F2-score	0.1359	0.1788	**0.2417**	**0.3186**	0.1831	0.1833	0.1663	0.1663	0.2322	0.3120
Hamming Loss	0.1548	0.1496	**0.1230**	**0.1196**	0.1532	0.1520	0.1580	0.1580	0.1237	0.1212
Precision	0.1221	0.1823	0.2214	**0.3331**	0.1876	0.1887	0.1702	0.1702	**0.2227**	0.3284
Recall	0.1740	0.1827	**0.3092**	**0.3118**	0.1933	0.1910	0.1752	0.1752	0.2656	0.3056
Subset Accuracy	0.0747	0.1493	0.1324	**0.2896**	0.1369	0.1403	0.1267	0.1267	**0.1471**	0.2749

TABLE VII
PERFORMANCE OVER THE ACTIONS DATA WITH TRAIN SUBSET.

Evaluation	J48		NB		OPF		OPFkNN		SMO	
	BR	LP	BR	LP	BR	LP	BR	LP	BR	LP
Accuracy	0.1878	0.2186	0.2544	0.3865	0.2809	0.2836	0.2853	0.2853	**0.3096**	**0.4595**
F2-score	0.2151	0.2280	0.2951	0.3956	0.2988	0.2999	0.3011	0.3011	**0.3261**	**0.4695**
Hamming Loss	0.1352	0.1412	0.1215	0.1062	0.1357	0.1345	0.1354	0.1354	**0.0900**	**0.0952**
Precision	0.1971	0.2336	0.2674	0.4135	0.2964	0.2994	0.3000	0.3000	**0.3217**	**0.4847**
Recall	0.2651	0.2311	**0.3861**	0.3871	0.3181	0.3158	0.3169	0.3169	0.3499	**0.4646**
Subset Accuracy	0.1154	0.1912	0.1527	0.3597	0.2296	0.2364	0.2398	0.2398	**0.2636**	**0.4299**

TABLE VIII
PERFORMANCE OVER THE SCENES DATA WITH AUTO-TRAIN SUBSET.

Evaluation	J48		NB		OPF		OPFkNN		SMO	
	BR	LP	BR	LP	BR	LP	BR	LP	BR	LP
Accuracy	0.1507	0.1604	**0.1932**	0.2543	0.1720	0.1744	0.1589	0.1589	0.1465	**0.2732**
F2-score	0.1740	0.1675	**0.2448**	0.2703	0.1837	0.1850	0.1667	0.1667	0.1660	**0.2858**
Hamming Loss	0.1686	0.1869	0.2332	0.1558	0.1818	0.1794	0.1823	0.1823	**0.1416**	**0.1548**
Precision	0.1682	0.1770	**0.2116**	0.3024	0.1964	0.1993	0.1778	0.1778	0.1635	**0.3110**
Recall	0.2027	0.1649	**0.3952**	0.2543	0.1821	0.1813	0.1632	0.1632	0.1873	**0.2732**
Subset Accuracy	0.0911	0.1392	0.0773	0.2062	**0.1375**	0.1426	0.1357	0.1357	0.0945	**0.2354**

C. Experiments with BoF descriptor

Considering BoF descriptor, all classifiers have improved their classification rates. Table VI displays the results over Actions dataset with AUTO-TRAIN as training set. As one can note, the NB classifier was the best performer, especially when allied with the LP method, achieving the best rates in all measures. NB also kept the highest recall using BR as transformation method. SMO was the second best regarding to LP, and improved considerably its results with BR, compared to those presented with HMP descriptor. The OPF-based classifiers were the third best ones, and they also kept the similarity between them. Finally, J48 was the worst performer.

With the CLEAN-TRAIN as training set, the classification rates were even better, as can be seen in Table VII. However, SMO achieved the best recognition rates with both LP and BR. OPFkNN and OPF achieved similar classification rates, being them the second best using BR. NB was the second best allied to the LP methods, and the best with BR according to the Recall measure.

Finally, Table VIII shows the classification rates over Scenes dataset. Considering the LP method, SMO and NB were the first and second best classifiers, respectively. OPF performed slight better than OPFkNN and J48. In regard to the BR transformation, NB achieved the best rates in four of six measures. However, we can note that NB was the worst classifier according to Hamming Loss and Subset Accuracy, which shows an unbalance, specially whether we consider its low value for Subset Accuracy, which demonstrates some difficulties to predict an exactly set of label to a sample. SMO and J48 achieved the best Hamming Loss rates, but they did not perform well regarding the other measures. The OPF-based classifiers achieved the best Subset Accuracy rate, showing they were good performers to match exactly a set of label.

D. Analysis and considerations

As the reader may have noticed, the OPFkNN perform equally using both LP and BR methods. This may happen due to the fact that OPFkNN generates the same collection of optimum-path trees for both cases, where the same set of prototypes propagates their labels when the training phase occurs. Even when BR is employed, the propagated labels always form the same set of label that compose the Label Powerset for a given training sample.

Unlike, the traditional OPF with BR method may generate different optimum-path trees for each binary classifier required by BR. This behaviour is determined by the prototypes set,

once a sample may be a prototype in a binary training set and may be not in another one. However, both OPF and OPFkNN performed similarly, and presented better results when allied to BR, overcoming traditional classifiers. A possible reason to explain the low classification rates with LP may stand for the high number of classes that this method generates, in the expense of the low number of samples by classes. This may imply in many trivial trees, i.e., trees with only one node, reducing the OPF-based classifiers effectiveness.

VII. Conclusions

In this work we addressed the video content as a multi-label problem. We also evaluated the performance of two OPF-based classifiers to accomplish this task. We employed a public video content dataset composed of human actions and scenes, and two video descriptors were also employed in order to evaluate the classifiers' performances.

In order to handle with the multi-label provided by the dataset, we opted to use two different problem transformation methods, which employ traditional classifiers as core. Three multi-classes classifiers were used to compare the OPF-based classifiers, which had not yet been evaluated in the multi-label context.

The results showed that the OPF-based classifiers have potential to be employed in multi-label tasks, specially when the focus are build binary classifiers, where they can overcome traditional classifiers as Naïve Bayes and SVM.

Future work includes the evaluation of other visual features and similarity metrics (e.g., camera motion [20]). In addition, the proposed method can be augmented to consider temporal segmentation [21] and/or summarization methods [22]. Finally, we also plan to apply the OPF-classifiers allied to different multi-label methods, and to explore domains with hundreds of labels, i.e., texts and musical categorization, for instance.

Acknowledgements

The authors would like to thank FAPESP (grants 2009/16206-1, 2011/11171-5, 2011/14058-5, and 2011/14094-1), CNPq (grants 303182/2011-3, 306580/2012-8, and 484254/2012-0), and CAPES for the financial support.

References

[1] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *Int J Data Warehousing and Mining*, vol. 2007, pp. 1–13, 2007.

[2] T. Li and M. Ogihara, "Detecting emotion in music," in *In Proceedings of the Fifth International Symposium on Music Information Retrieval*, 2003, pp. 239–240.

[3] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.

[4] A. K. McCallum, "Multi-label text classification with a mixture model trained by em," in *Proceedings of the AAAI 99 Workshop on Text Learning*, 1999, pp. 1–9.

[5] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[6] M. ling Zhang and Z. hua Zhou, "Ml-knn: A lazy learning approach to multi-label learning," *Pattern Recognition*, vol. 40, p. 2007, 2007.

[7] R. Jin and Z. Ghahramani, "Learning with multiple labels," in *Proceedings of the Neural Information Processing Systems*. MIT Press, 2002, pp. 897–904.

[8] J. P. Papa, A. X. Falcão, and C. T. N. Suzuki, "Supervised pattern classification based on optimum-path forest," *International Journal of Imaging Systems and Technology*, vol. 19, no. 2, pp. 120–131, 2009.

[9] J. P. Papa, A. X. Falcão, V. H. C. Albuquerque, and J. M. R. S. Tavares, "Efficient supervised optimum-path forest classification for large datasets," *Pattern Recognition*, vol. 45, no. 1, pp. 512–520, 2012.

[10] A. X. Papa, João P.; Falcão, "A new variant of the optimum-path forest classifier." *International Symposium on Visual Computing*, vol. 47, no. 1, pp. 935–944, 2008.

[11] G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. Dzeroski, "An extensive experimental comparison of methods for multi-label learning," *Pattern Recognition*, vol. 45, no. 9, pp. 3084 – 3104, 2012.

[12] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.

[13] A. Falcão, J. Stolfi, and R. Lotufo, "The image foresting transform theory, algorithms, and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 19–29, 2004.

[14] M. Marszałek, I. Laptev, and C. Schmid, "Actions in context," in *IEEE Conference on Computer Vision & Pattern Recognition*, 2009.

[15] J. M. Chaquet, E. J. Carmona, and A. Fernndez-Caballero, "A survey of video datasets for human action and activity recognition," *Computer Vision and Image Understanding*, vol. 117, no. 6, pp. 633 – 659, 2013.

[16] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *Conference on Computer Vision & Pattern Recognition*, jun 2008.

[17] J. Almeida, N. J. Leite, and R. da S. Torres, "Comparison of video sequences with histograms of motion patterns," in *IEEE International Conference on Image Processing (ICIP'11)*, 2011, pp. 3673–3676.

[18] I. Laptev, "On space-time interest points," *International Journal of Computer Vision*, vol. 64, no. 2-3, pp. 107–123, 2005.

[19] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Proceedings of the 9th international conference on Computer Vision*, ser. ICCV'03, 2003, pp. 1470–1477.

[20] J. Almeida, R. Minetto, T. A. Almeida, R. da S. Torres, and N. J. Leite, "Estimation of camera parameters in video sequences with a large amount of scene motion," in *International Conference on Systems, Signals and Image Processing (IWSSIP'10)*, 2010, pp. 348–351.

[21] J. Almeida, N. J. Leite, and R. da S. Torres, "Rapid cut detection on compressed video," in *Iberoamerican Congress on Pattern Recognition (CIARP'11)*, 2011, pp. 71–78.

[22] J. Almeida, N. J. Leite, and R. da S. Torres, "VISON: VIdeo Summarization for ONline applications," *Pattern Recognition Letters*, vol. 33, no. 4, pp. 397–409, 2012.