方法 BPMLL，以及基于转换的学习方法 LP 相比较，最后给出了一些比较之后的总结。

2 多标记分类简介

多标记学习问题可以描述如下：设样本的特征属性$X = \{x_i : i = 1 \dots m\}$，有限标记集合$Y = \{\lambda_j : j = 1 \dots q\}$，给定学习样本集$S = \{(x_i, L_i) : i = 1 \dots k\}$，其中$x_i \in X, L_i \subseteq Y$。要求构造分类器 h，能够对未知样本集$T = \{(x_{k+i}, ?) : i = 1 \dots p\}$进行标记。

2.1 分类算法描述

解决多标记学习的思路主要有两种：一是算法独立，亦即通过对样本集进行分解，将多标记学习转化为多个单标记学习问题来处理；二是算法依赖，亦即通过对原有算法进行改造，使其能够处理多标记问题。下面将分别回顾基于这两种思想的一些常见方法。

2.1.1 问题转换方法

（1） 基于标记转换方法

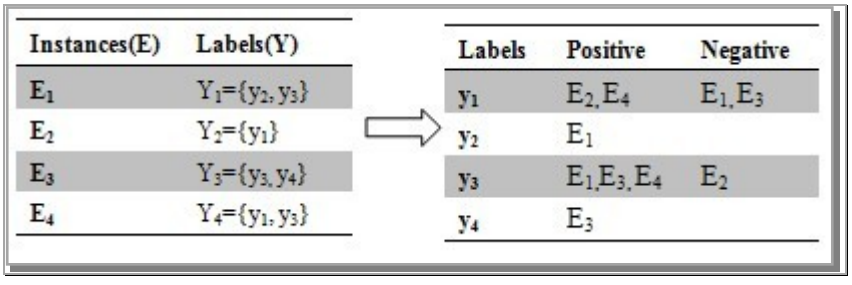
假设样本实例的标记总数为 q，针对每个标记分类出属于这个标记的为一类，不属于的为另一类。这样的二分转换思想可以直观的参考图 1 所示。

典型的利用这种思想的多标记分类方法是 Binary Relevance (BR)。 BR 方法将原来的数据集分成了 q 个二分数据集$D_{\lambda_j}, j = 1 \dots q$，其中每个数据集包含了所有原数据集中的样本实例。但是这每个数据集都属于单个标记集并且标记仅为 Positive 和 Negative，根据原数据集得出的D_{λ_j}数据集中的每个实例要么标记为 Positive，要么为 Negative。分类一个新的实例，BR 输出一个合集，这个合集是由 q 个基分类器输出中包含的 Positive 例组成的。从上文分析看来 BR 算法的好处是它的计算复杂度相对其他算法而言较小。对于实例固定的样本而言，BR 的时间复杂度和样本标记集 L 的标记数量 q 成正比，它的复杂度为$q \times O(C)$，其中$O(C)$为基础分类算法的复杂度。因此，BR 算法针对标记数量 q 比较小的情况下适用。然而，在很多领域中是存在大量标记的，甚至这些标记是有树状的层次的关联的。对于这种情况，BR 算法的局限性就比较大，因为它没有考虑到这些标记之间的关联性。

Classifier Chain(CC)[4]方法成功克服了 BR 没考虑标记之间的关联性这一缺点。CC 方法依然使用 BR 所使用的二叉分类。与 BR 不同的是，它将这些基分类器$C_j, j = 1 \dots q$串联起来形成一条链。CC 方法可以大致描述如下：一个分类器C_j对应一个标记λ_j。假设一个新的实例 x 需要分类，分类器C_1判断x是否属于标记λ_1 ，设其值为$y \in \{0,1\}$，得出$Pr(\lambda_1|x)$。分类器 C_2判断 x 是否属于标记λ_2，但是此时会将上 y_1作为输入得到$Pr(\lambda_2|x,\lambda_1)$。以此类推，当$C_j$判断 x 是否属于标记$\lambda_j$时，会将$y_1, \dots, y_{j-1}$作为额外的信息输入得到$Pr(\lambda_2|x, \lambda_1, \dots, \lambda_{j-1})$。这种链的方式使得标记信息在分类器之间传递，考虑到了标记之间的关联性，克服了 BR 的缺点，并且仍然保持了 BR 的计算复杂度低的优点。

为了提高整体的精确度，并且实现并行，研究者们还为 BR、CC 等提出了 Ensemble 的框架，得到了 EBR、ECC 等的算法。这些算法表现出了很好的性能，在此不作详述。

图 1 基于标记转换方法示例



Instances(E)	Labels(Y)
E_1	Y_1={y_2, y_3}
E_2	Y_2={y_1}
E_3	Y_3={y_3, y_4}
E_4	Y_4={y_1, y_3}

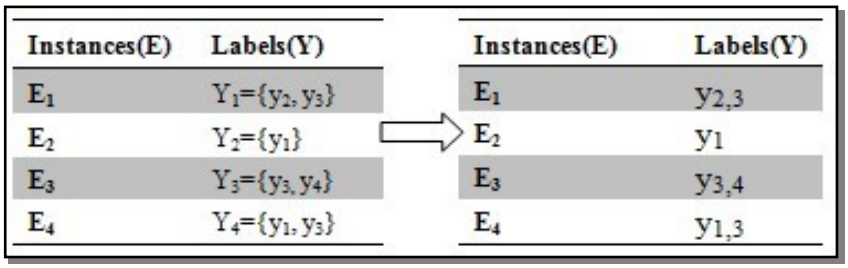
Labels	Positive	Negative
y_1	E_2, E_4	E_1, E_3
y_2	E_1	
y_3	E_1, E_3, E_4	E_2
y_4	E_3	

（2） 基于样本实例转换方法

基于样本实例转换的方法是对于每个实例将其所属的标记进行重新定义使得问题转换为一个或多个单标记的问题去处理。这种转换有三种不同的思路：

1） 创建新的标记。这种方法的思路是将每个多标记实例的所属标记联合起来创建新的标记。如图 2 所示，实例 E_1 属于标记 y_1 和 y_2 ，为此就创建新的标记 $y_{1,2}$。这种方法的实现命名为 Label-Powerset[5]，简称 LP。诚然，这样做的代价是标记的数量就会增加，并且一些标记只有很少的实例。但是 LP 的优点是考虑到了标记之间的关联性。

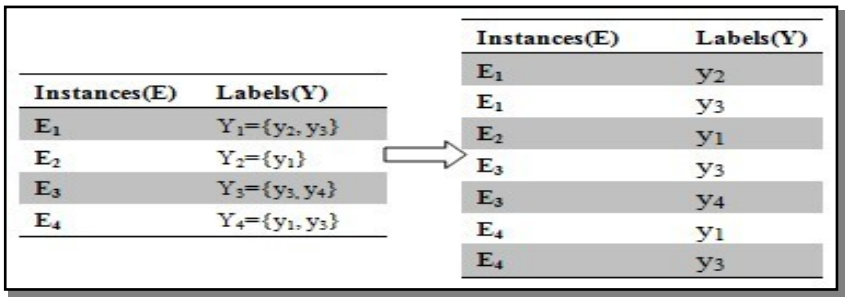
图 2 创建标记转换方法



2） RAndom k-LabELsets。Random k-labelsets[5] 简称 RAKEL 是建立了一个 LP 分类器的 Ensemble，用标签集合的一小部分随机标签子集的数据集作为每一个 LP 分类器的训练集训练。RAKEL 通过这种方式去考虑标签之间的相互关系，同时也就避免了 LP 的缺陷。标签的 ranking 通过每一个基分类器的 0 或 1 预测结果来获得。通过设置阈值也可以产生二值的分类结果。

3） 分解多标记。这种方法是通过将标记多标记分解，这样所有的多标记的实例被分解成多个单标记的实例。在训练过程中，多个标记的实例被利用多次，如图 3 所示。例如，实例 E_1 属于标记 y_2 和 y_3，当 y_2 的分类器被训练之后，其他所有的多标记并且属于 y_2 的实例对于 y_2 的分类 2 器来说都是属于 y_2 的单标记，对于其他的标记也是如此。这种方法被称作 Cross-Training，在此不详述细节，读者可参考文献[6]。

图 3 分解多标记



2.1.2 算法改造方法

顾名思义，这些算法是针对特殊的算法改造而来。改造方法的优点在于，通常在特殊的实际问题中，聚焦特殊的算法要比那些算法独立的方法要优越。下面介绍一些基于改造的算法，由于研究本课题时间较短，所以在此简略介绍其中四种。

（1） 决策树（Decision Trees）

交替决策树学习算法[7]引进了交替决策树，它是决策树的一种派生，并且它的引进原则是 boosting。对于多标记分类问题是基于 AdaBoost[8]和 ADTBoost[9]。在 DTs 运用中，还有研究者改造 C4.5[10]算法并运

用于基因分类，在此不作详述。

（2） 支持向量机（Support Vector Machines）

Support Vector Machines （SVMs）[11]是建立在统计学习理论基础上的机器学习方法。通过学习算法，SVMs 可以自动寻找出那些对分类有较好区分能力的支持向量，由此构造出的分类器可以最大化类与类的间隔，因而有较好的适应能力和较高的分准率。该方法只需要由各类域的边界样本的类别来决定最后的分类结果。由于待分样本集中的大部分样本不是支持向量，移去或者减少这些样本对分类结果没有影响，所以 SVM 法对小样本情况下的自动分类有着较好的分类结果。

（3） K-Nearest Neighbor (KNN)

KNN[12] 方法的思路是：如果一个样本在特征空间中的 k 个最相似(最邻近)的样本中的大多数属于某一个类别，则该样本也属于这个类别。该方法在定类决策上只依据最邻近的一个或者几个样本的类别来决定待分样本所属的类别。由于 KNN 方法主要靠周围有限的邻近的样本，因此对于类域的交叉或重叠较多的待分样本集来说，KNN 方法较其他方法更为适合。该方法的不足之处是计算量较大，因为对每一个待分类的文本都要计算它到全体已知样本的距离，才能求得它的 K 个最近邻点。

Multi-Label k-Nearest Neighbor[13] 简称 MLkNN 是从熟悉的 KNN 算法派生而来。由于针对每个测试样本，它的 KNN 都已经在学习样本中确定，所以根据这些已经获取的近邻标记集的统计信息，用最大后验概率原则（MAP）去决定测试样本的标签集合，最大后验概率是基于 KNN 对每个标签的前验和后验概率。

Binary Relevance k-Nearest Neighbor[14] 简称 BRkNN 是结合 Binary Relevance (BR)的 KNN 算法的改造。由于将 BR 和 KNN 算法结合配对的时间需要比计算相同规模的 KNN 多花 L 倍时间，因此为了避免这些冗余的时间密集型计算，BRkNN 扩展了原来的 KNN 算法，在搜索一遍 k 个近邻标记后，将每个标记作单独预测。为此，BRkNN 比单纯的 BR 加 KNN 快了 L 倍。

（4） 神经网络

神经网络分类算法的重点是构造阈值逻辑单元，一个值逻辑单元是一个对象，它可以输入一组加权系数的量，对它们进行求和，如果这个和达到或者超过了某个阈值，输出一个量。神经网络是基于经验风险最小化原则的学习算法，有一些固有的缺陷，比如层数和神经元个数难以确定，容易陷入局部极小，还有过学习现象等。

Back-propagation Multi-label Learning 简称 BPMLL[15] 是修改流行的反传(Back-propagation)算法来适应多标签数据的一个算法。这个算法主要的修改是引入了新的误差公式来考虑多标签。

2.2 算法评价指标

在文献[16] 中定义了目前多标记学习中的几种常用评价指标，分别简介如下，具体公式参见原文，其中指标(1)是基于算法输出的预测标记矩阵来计算的，指标(2)~(5)则基于算法所输出的实值矩阵计算。(1)~(4)的结果越小越好，而(5)的值则越大越好。

(1) 汉明损失(hamming loss)：该指标衡量了预测所得标记与样本实际标记之间的不一致程度，即样本具有标记 y_i 但未被识别出，或不具有标记 y_i 却别误判的可能性。

(2) 1-错误率(one-error)：该指标描述了样本所具体的隶属度最高的标记不是其实际标记的可能性，在单标记学习中，就演化成普通的分类错误率。

(3) 覆盖率(coverage)：该指标衡量了在排序队列中，从隶属度最高的类别开始，平均需要跨越多少标记才能覆盖样本所拥有的全部标记。

(4) 排序损失(ranking loss)：该指标表明了样本对其所属标记的隶属度低于对其非属标记的隶属度的可

能性。

(5) 平均精度(average precision)：该指标反映了预测类标的平均精确度。

3 实验

3.1 实验数据集

本实验用了三种不同的应用领域的数据集：(1)scene[17]是一个静态场景的语义索引数据集。(2)emotions[18]是一个歌曲的分类数据集。(3) yeast[19] 是一个关于蛋白质功能的数据集。

表 1 反映了一些关于这些数据集的相关统计信息。其中 Domain 表示数据所属应用领域，Instances 表示该数据集中的 Train 和 Test 样本实例总个数，Nominal 和 Numeric 是标记的属性，Labels 表示标签的个数，Label Cardinality 表示每个样本实例的平均标记数，这是个统计值，Label Density 是 Label Cardinality 与标记总数的商值。

表 1 样本数据集及其内部数据相关统计信息

Dataset	Domain	Instances	Nominal	Numeric	Labels	Label Cardinality	Label Density	Distinct Subsets
scene	Images	2407	0	294	6	1.074	0.179	15
emotions	Music	593	0	72	6	1.869	0.311	27
yeast	Biology	2417	0	103	14	4.237	0.303	198

3.2 比较方法和参数说明

首先，选择三种基于问题转换方法的算法进行比较，它们分别是上文提到的 CC、RAkEL 和 BR，选出相对最好的算法。其次，选择三种基于算法改造的算法进行比较，它们分别是 MLkNN、BRkNN 和 BPMLL，选出相对最好的。最后，将上面选出的最好的再进行比较。

本实验的实现代码全部来自 MuLan 开源库。实验针对每个测试数据集运用了 10-fold 交叉验证（cross-validation）方法，从而得出一些度量指标的统计平均值。

训练 CC、RAkEL 和 BR 的参数分别是：BR 和 CC 的基础分类器是 J48(C4.5)。RAkEL 的基础分类算法是上文提到的 LP，其基础分类器也是 J48(C4.5)，其中 k=3，n=2L（标记的数量）。

训练 MLkNN、BRkNN 和 BPMLL 的参数分别是：MLkNN 和 BRkNN 中 k=10，smoothing=1。BPMLL 中 learningRate=0.05，epochs=100，hiddenUnits=0.2。

4 实验结果

4.1 CC、RAkEL和BR 的比较

本节比较 CC、RAkEL 和 BR，这三种算法都是基于问题转换思想，其中 RAkEL 和 BR 都是运用的基础分类器 J48。表 2 中加粗了相对优越的指标，可以看出 RAkEL 的性能要比 BR 和 CC 优越。观察 CC 和 BR 的结果可以看出，这两者的性能差距较小，Training 和 Testing 的时间也相近，但是 CC 的 Average Precision 总体要比 BR 好 。从表 3 中可以看出 RAkEL 算法的 Training 和 Testing 时间比 BR 和 CC 长的多，而且结合表 1 和表 2 可以分析出，RAkEL(LP-based)的 Training 时间跟 Label Cardinality 和 Distinct Subsets 有很大的关系，不排除跟 LP 作为基础算法有关，需要进一步研究。如果要处理大量的数据集，并且反应时间的重要性大于精确度时，RAkEL 可能不是理想的算法。从数据来看，BR 的 Training 时间受 Label Cardinality 以及 Distinct Subsets 的变化影响不大。

表 2 CC、RAkEL 和 BR 比较结果

Measures / Dataset / Methods		Hamming Loss	One-error	Coverage	Ranking Loss	Average Precision	#wins (#better)	
scene	CC	0.1538±0.0190	0.4214±0.0478	1.3535±0.1575	0.2470±0.0276	0.7028±0.0312	0	RAkEL
	BR	0.1468±0.0089	0.4398±0.0388	1.3376±0.1474	0.2443±0.0266	0.6963±0.0259	0	
	RAkEL	**0.1041±0.0117**	**0.2860±0.0379**	**0.6897±0.1026**	**0.1177±0.0181**	**0.8223±0.0211**	5	
emotions	CC	0.2537±0.0563	0.4162±0.1632	2.6540±0.3750	0.2936±0.0670	0.6963±0.0728	0	RAkEL
	BR	0.2618±0.0342	0.4155±0.0841	2.7102±0.3919	0.2972±0.0601	0.6890±0.0444	0	
	RAkEL	**0.2343±0.0373**	**0.3060±0.0950**	**2.0052±0.3423**	**0.1839±0.0490**	**0.7835±0.0454**	5	
yeast	CC	0.2871±0.0163	0.3761±0.0662	9.3069±0.3789	0.3613±0.0262	0.5996±0.0321	0	RAkEL
	BR	0.2678±0.0140	0.3523±0.0768	9.6079±0.4411	0.3310±0.0257	0.6136±0.0261	0	
	RAkEL	**0.2411±0.0089**	**0.3108±0.0316**	**7.6362±0.2206**	**0.2256±0.0150**	**0.7023±0.0189**	5	

表 3 CC、RAkEL 和 BR 的 Train、Test 时间比较

	scene			emotions			yeast		
	CC	BR	RAkEL	CC	BR	RAkEL	CC	BR	RAkEL
Training Time（ms）	10453	**10405**	43714	**795**	1376	3061	11945	**10323**	89112
Testing Time（ms）	**8567**	8625	37708	**276**	594	1072	**5530**	5650	44592

4.2 MLkNN、BRkNN和BPMLL的比较

从表 4 中的数据可以看出：对于 scene 数据集和 yeast 数据集，MLkNN 的 5 种评价值要比其他两种都要好，BRkNN 其次。对于 emotions 数据集，BPMLL 的 One-error、Coverage、Ranking Ross 和 Average Precision 都比其他两种要好，这似乎跟 emotions 的实例较少，Label Density 较大有一定的关系，但是还需要进一步研究。从表 3 中看出，BRkNN 的 training 时间相对 MLkNN 和 BPMLL 要小很多。权衡来看，当数据量比较大的时候，并且需要少的反应时间时，BRkNN 是个比较适合的选择。

表 4 MLkNN、BRkNN 和 BPMLL 比较结果

Measures / Dataset / Methods		Hamming Loss	One-error	Coverage	Ranking Loss	Average Precision	#wins (#better)	
scene	MLkNN	**0.0911±0.0063**	**0.2416±0.0281**	**0.5267±0.0696**	**0.0849±0.0126**	**0.8542±0.0180**	5	MLkNN
	BRkNN	0.0982±0.0076	0.2642±0.0304	0.6071±0.0853	0.1013±0.0145	0.8377±0.0191	0	
	BPMLL	0.2479±0.0475	0.5044±0.0760	0.9118±0.1388	0.1622±0.0282	0.7060±0.0465	0	
emotions	MLkNN	0.2077±0.0396	0.2629±0.0609	1.8731±0.3271	0.1614±0.0529	0.8096±0.0441	0	BPMLL
	BRkNN	**0.1980±0.0438**	0.2717±0.0844	1.8179±0.3184	0.1579±0.0500	0.8114±0.0467	1	
	BPMLL	0.2049±0.0510	**0.2583±0.114**	**1.8095±0.4689**	**0.1514±0.0773**	**0.8211±0.0720**	4	
yeast	MLkNN	**0.1971±0.0106**	0.2333±0.0592	**6.3563±0.2142**	**0.1703±0.0185**	**0.7595±0.0290**	4	MLkNN
	BRkNN	0.2036±0.0085	**0.2322±0.0518**	6.6098±0.2370	0.1795±0.0145	0.7567±0.0235	1	
	BPMLL	0.2430±0.0172	0.2430±0.0172	7.0196±0.1979	0.2168±0.0152	0.7195±0.0217	0	

表 5 MLkNN、BRkNN 和 BPMLL 的 Train、Test 时间比较

	scene			emotions			yeast		
	MLkNN	BRkNN	BPMLL	MLkNN	BRkNN	BPMLL	MLkNN	BRkNN	BPMLL
Training Time(ms)	15100	**10**	27427	440	**2**	1376	9741	**1**	10323
Testing Time（ms）	13773	**1393**	24716	134	**22**	594	3490	**393**	5650

4.3 RAkEL和MLkNN的比较

从表 4 和表 5 可以看出，针对数据集 scene、emotions 和 yeast，MLkNN 的性能要比 RAkEL(LP-based)的性能要好。很自然可以想到，如果 RAkEL 将 MLkNN 作为基础算法，那么它的性能会不会要比 MLkNN 本身要好呢，这将是下一步工作需要做的。

表 6 RAkEL 和 MLkNN 的比较结果

Measures / Dataset / Methods		Hamming Loss	One-error	Coverage	Ranking Loss	Average Precision	#wins (#better)	
scene	MLkNN	**0.0911±0.0063**	**0.2416±0.0281**	**0.5267±0.0696**	**0.0849±0.0126**	**0.8542±0.0180**	5	MLkNN
	RAkEL	0.1041±0.0117	0.2860±0.0379	0.6897±0.1026	0.1177±0.0181	0.8223±0.0211	0	
emotions	MLkNN	**0.2077±0.0396**	**0.2629±0.0609**	**1.8731±0.3271**	**0.1614±0.0529**	**0.8096±0.0441**	5	MLkNN
	RAkEL	0.2343±0.0373	0.3060±0.0950	2.0052±0.3423	0.1839±0.0490	0.7835±0.0454	0	
yeast	MLkNN	**0.1971±0.0106**	**0.2333±0.0592**	**6.3563±0.2142**	**0.1703±0.0185**	**0.7595±0.0290**	5	MLkNN
	RAkEL	0.2411±0.0089	0.3108±0.0316	7.6362±0.2206	0.2256±0.0150	0.7023±0.0189	0	

表 7 RAkEL 和 MLkNN 的 Training、Test 时间

	scene		emotions		yeast	
	MLkNN	RAkEL	MLkNN	RAkEL	MLkNN	RAkEL
Training Time(ms)	15100	**43714**	440	**3061**	9741	**89112**
Testing Time（ms）	13773	**37708**	134	**1072**	3490	**44592**

4.4 总结

本篇文章总结了多标记分类的两种常见思路，并且介绍了基于这两种思路的一些主流的算法。另外，本文还选择其中六种算法，针对三种数据源进行了分组比较，通过比较发现 MLkNN 的整体性能最优。由于时间较短的原因，本文的比较只是通过一些常规性指标，以及 Training 和 Testing 时间做了比较。如果有更多时间的话，我想可以对测试集的分类结果进行统计，并画出直方图，来比较说明分类的结果；也可以针对每个算法在训练数据集时的内存使用情况进行比较。

References:

[1] Kazawa H, lzumitani T, Taira H, et al. Maximal margin labeling for multi-topic text categorization. Proceedings of Advances in Neural Information Processing Systems. Canada: Vancouver, 2003, 16:647-656.

[2] Boutell M R, Luo J, Shen X, et al. Learning multi-label scene classification. Pattern Recognition, 2004, 37(9): 1757-1771.

[3] Diplaris S, Tsoumakas G, Mitkas P, et al. Protein classification with multiple algorithms. Proceedings of the 10th Panhellenic Conference on Informatics(PCI 2005). Greece: Springer, 2005: 448-456.

[4] Read, J., Pfahringer, B., Holmes, G., and Frank, E. Classifier chains for multi-label classification. In ECML/PKDD 2009, 2009，pp. 254–269.

[5] G. Tsoumakas and I. Vlahavas, “Random k-labelsets: An ensemble method for multilabel classification,” in Proc. Of the 18th European Conference on Machine Learning (ECML 2007), Warsaw, Poland, September 17-21 2007, pp. 406–417.

[6] Shen, X., Boutell, M., Luo, J., Brown, C.: Multi-label machine learning and its application to semantic scene clasification. In: International Symposium on Electronic Imaging, San Jose, CA, January 2004, pp. 18–22.

[7] Freund, Y., Mason, L.: The alternating decision tree learning algorithm. In: Proceedings of the Sixteenth International Conference on Machine Learning, ICML, 1999, pp. 124–133.

[8] Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. In: Vitányi, P.M.B. (ed.) EuroCOLT 1995. LNCS, vol. 904, Springer, Heidelberg. 1995, pp. 23–37.

[9] Freund, Y., Mason, L.: The alternating decision tree learning algorithm. In: Proceedings of the Sixteenth International Conference on Machine Learning, ICML, 1999, pp. 124-133.

[10] Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco, 1993.

[11] Vallim, R.M.M., Goldberg, D.E., Llorà, X., Duque, T.S.P.C.: A New Approach for Multi-label Classification Based on Default Hierarchies and Organizational Learning, IWLCS. In: The 11th International Workshop on Learning Classifier Systems, part of the Genetic and Evolutionary Computation 2008 Conference (GECCO 2008), Atlanta, Georgia, USA (accepted), 2008.

[12] Witten, I., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques (2nd Edition). Morgan Kaufmann , 2005.

[13] Zhang, M.-L., Zhou, Z.-H.: A k-nearest neighbor based algorithm for multi-label classification. In: Proceedings of the 1st IEEE International Conference on Granular Computing (GrC 2005), Beijing, China, 2005, pp. 718–721.

[14] E. Spyromitros, G. Tsoumakas, and I. Vlahavas. An empirical study of lazy multilabel classification algorithms. In Proc. 5th Hellenic Conference on Artificial Intelligence (SETN 2008), 2008.

[15] M.-L. Zhang and Z.-H. Zhou, Multi-label neural networks with applications to functional genomics and text categorization, IEEE Transactions on Knowledge and Data Engineering, vol. 18, no. 10, 2009, pp. 1338–1351.

[16] Schapire R E, Singer Y. Boostexter: A boosting-based system for text categorization. Machine Learning, 2000, 39(2/3): 135-168.

[17] M.R. Boutell, J. Luo, X. Shen, and C.M. Brown. Learning multi-label scene classiffication. Pattern Recognition, 2004, 37(9):1757-1771.

[18] K. Trohidis, G. Tsoumakas, G. Kalliris, I. Vlahavas. Multilabel Classification of Music into Emotions. Proc. 2008 International Conference on Music Information Retrieval (ISMIR 2008), Philadelphia, PA, USA, 2008, pp. 325-330.

[19] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In T.G. Dietterich, S. Becker, and Z. Ghahramani, (eds), Advances in Neural Information Processing Systems 14, 2002.