Multi-label Classification with Error-correcting Codes

Chun-Sung Ferng r99922054@csie.ntu.edu.tw
Hsuan-Tien Lin htlin@csie.ntu.edu.tw
Department of Computer Science and Information Engineering, National Taiwan University

Editor: Chun-Nan Hsu and Wee Sun Lee

Abstract

We formulate a framework for applying error-correcting codes (ECC) on multi-label classification problems. The framework treats some base learners as noisy channels and uses ECC to correct the prediction errors made by the learners. An immediate use of the framework is a novel ECC-based explanation of the popular random k-label-sets (RAKEL) algorithm using a simple repetition ECC. Using the framework, we empirically compare a broad spectrum of ECC designs for multi-label classification. The results not only demonstrate that RAKEL can be improved by applying some stronger ECC, but also show that the traditional Binary Relevance approach can be enhanced by learning more parity-checking labels. In addition, our study on different ECC helps understand the trade-off between the strength of ECC and the hardness of the base learning tasks.

Keywords: Multi-label Classification, Error-correcting Codes

1. Introduction

Multi-label classification is an extension of traditional multi-class classification. In particular, the latter aims at accurately associating one single label with an instance while the former aims at associating a label-set. Because of the increasing application needs in domains like text and music categorization, scene analysis and genomics, multi-label classification is attracting much research attention in recent years.

Error-correcting code (ECC) roots from the information theoretic pursuit of communication (Shannon, 1948). In particular, ECC studies how to accurately recover a desired signal block after transmitting the block's encoding through a noisy communication channel. When the desired signal block is the single-label (of some instances) and the noisy channel consists of some binary classifiers, it has been shown that a suitable use of ECC could improve the association (prediction) accuracy of multi-class classification (Dietterich and Bakiri, 1995). In particular, with the help of ECC, we can reduce multi-class classification to several binary classification tasks. Then, following the foundation of ECC in information theory (Shannon, 1948; Mackay, 2003), a suitable ECC can correct a small portion of binary classification errors during the prediction stage and thus improve the prediction accuracy. Several designs, including some classic ECC (Dietterich and Bakiri, 1995) and some adaptively-constructed ECC (Schapire, 1997; Li, 2006), have reached promising empirical performance for multi-class classification.

While the benefits of ECC are well-established for multi-class classification, the corresponding use for multi-label classification remains an ongoing research direction. Kouzani and Nasireding (2009) take the first step on the direction by proposing a multi-label classification approach that applies a classic ECC, the Bose-Chaudhuri-Hocquenghem (BCH) code, using a batch of binary classifiers as the noisy channel. The work is followed by some extensions to the convolution code (Kouzani, 2010). Although the approach shows some good experimental results over existing multi-label classification approaches, a more rigorous study remains needed to understand the advantages and disadvantages of different ECC designs for multi-label classification and will be the main focus of this paper.

In this work, we formalize the framework for applying ECC on multi-label classification. The framework is more general than both existing ECC studies for multi-class classification (Dietterich and Bakiri, 1995) and for multi-label classification (Kouzani and Nasireding, 2009). Then, we conduct a thorough study with a broad spectrum of classic ECC designs: repetition code, Hamming code, BCH code and low-density parity-check code. The four designs cover the simplest ECC idea to the state-of-the-art ECC in communication systems. Interestingly, such a framework allows us to give a novel ECC-based explanation to the random k-label-sets (RAKEL) algorithm, which is popular for multi-label classification. In particular, RAKEL can be viewed as a special type of repetition code coupled with a batch of simple multi-label classifiers.

We empirically demonstrate that RAKEL can be improved by replacing its repetition code with the Hamming code, a slightly stronger ECC. Furthermore, even better performance can be achieved when replacing the repetition code with the BCH code. When compared with the traditional Binary Relevance approach without ECC, multi-label classification with ECC can perform significantly better. The empirical results justify the validity of the ECC framework.

The paper is organized as follows. First, we introduce the multi-label classification problem and present related works in Section 2. Section 3 formalizes the framework for applying ECC on multi-label classification; Section 4 reviews the four ECC designs that we study. Then, in Section 5, we describe the ECC view of RAKEL. Finally, we discuss the results from experiments in Section 6 and conclude in Section 7.

2. Setup and Review

Multi-label classification aims at mapping an instance $\mathbf{x} \in \mathbb{R}^d$ to a label-set $Y \subseteq \mathcal{L} = \{1, 2, \ldots, K\}$, where K is the number of classes. Following the hypercube view of Tai and Lin (2010), the label-set Y can be represented as a binary vector $\mathbf{y}$ of length K, where $\mathbf{y}[i]$ is 1 if the i-th label is in Y, and 0 otherwise. Consider a training data set $D = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^{N}$. A multi-label classification algorithm uses D to locate a multi-label classifier $h\colon \mathbb{R}^d \to \{0,1\}^K$ such that $h(\mathbf{x})$ predicts $\mathbf{y}$ well on future test examples $(\mathbf{x}, \mathbf{y})$.

There are several loss functions for evaluating whether $h(\mathbf{x})$ predicts $\mathbf{y}$ well. Two common ones are:

- **subset** 0/1 **loss**: $\Delta_{0/1}(\tilde{\boldsymbol{y}}, \mathbf{y}) = [\![\tilde{\boldsymbol{y}} \neq \mathbf{y}]\!]$, which is arguably one of the most challenging loss functions because zero (small) loss occurs only when every bit of the prediction is correct.
- **Hamming loss**: $\Delta_{HL}(\tilde{\boldsymbol{y}}, \mathbf{y}) = \frac{1}{K}\sum_{i=1}^{K} [\![\tilde{\boldsymbol{y}}[i] \neq \mathbf{y}[i]]\!]$, which considers individual bit differences.

Dembczyński et al. (2010) show that the two loss functions focus on different statistics of the underlying probability distribution from a Bayesian perspective. While a wide range of other loss functions exist (Tsoumakas and Vlahavas, 2007), in this paper we only focus on 0/1 and Hamming because they connect tightly with the ECC framework that will be discussed.[1] Note that the subset 0/1 loss is also conventionally listed in its complement form $A(\tilde{\boldsymbol{y}}, \mathbf{y}) = 1 - \Delta_{0/1}(\tilde{\boldsymbol{y}}, \mathbf{y})$, which is called subset accuracy (Tsoumakas and Vlahavas, 2007). We take such a convention and report both accuracy and Δ_{HL} in this paper.

The hypercube view (Tai and Lin, 2010) unifies many existing problem transformation approaches (Tsoumakas and Vlahavas, 2007) for multi-label classification. Problem transformation approaches transform multi-label classification into to one or more reduced learning tasks. For instance, one simple problem transformation approach for multi-label classification is called binary relevance (BR), which learns one binary classifier per each individual label. Another simple problem transformation approach is called label powerset (LP), which transforms multi-label classification to one multi-class classification task with a huge number of extended labels. One popular problem transformation approach that lies between BR and LP is called random k-label-sets (RAKEL; Tsoumakas and Vlahavas, 2007), which transforms multi-label classification to many multi-class classification tasks with a smaller number of extended labels.

Multi-label classification with compressive sensing (Hsu et al., 2009) is a problem transformation approach that encodes the training label-set $\mathbf{y}_n$ to a shorter, real-valued codeword vector using compressive sensing. Tai and Lin (2010) study some different encoding schemes from label-sets to real-valued codewords. Note that those encoding schemes focus on *compression*—removing the redundancy within the binary signals (label-sets) to form the shorter codewords. The compression perspective can lead to not only more efficient training and testing, but also more meaningful codewords.

Compression is a classic task in information theory based on Shannon's first theorem (Shannon, 1948). Another classic task in information theory aims at *expansion*—adding redundancy to the (longer) codewords to ensure robust decoding against noise contamination. The power of expansion is characterized by Shannon's second theorem (Shannon, 1948). ECC targets towards using the power of expansion systematically. In particular, ECC works by encoding a block of signal to a longer codeword $\mathbf{b}$ before passing it through the noisy channel, and then decoding the received codeword $\tilde{\mathbf{b}}$ back to the block appropriately. Then, under some assumptions (Mackay, 2003), the block can be perfectly recovered—resulting in zero block-decoding error; in some cases, the block can only be almost perfectly recovered—resulting in a few bit-decoding errors.

If we take the "block" as the label-set $\mathbf{y}$ for every example $(\mathbf{x}, \mathbf{y})$ and a batch of base learners as a channel that outputs the contaminated block $\tilde{\boldsymbol{b}}$, the block-decoding error corresponds to $\Delta_{0/1}$ while the bit-decoding error corresponds to a scaled version of Δ_{HL}. Such a correspondence motivates us to study whether suitable ECC designs can be used to improve multi-label classification, which will be formalized in the next section.

3. ECC for Multi-label Classification

We now describe the ECC framework in detail. The main idea is to use an ECC encoder $enc(\cdot)\colon \{0,1\}^K \to \{0,1\}^M$ to expand the original label-set $\boldsymbol{y} \in \{0,1\}^K$ to a codeword

1. We follow the final remark of Dembczyński et al. (2010) to only focus on the loss functions that are related to our algorithmic goals.

$\boldsymbol{b} \in \{0,1\}^M$ that contains redundancy information. Then, instead of learning a multi-label classifier $h(\boldsymbol{x})$ between $\boldsymbol{x}$ and $\boldsymbol{y}$, we learn a multi-label classifier $\tilde{h}(\boldsymbol{x})$ between $\boldsymbol{x}$ and the corresponding $\boldsymbol{b}$. In other words, we transform the original multi-label classification problem to another multi-label classification task. During prediction, we use $h(\boldsymbol{x}) = dec \circ \tilde{h}(\boldsymbol{x})$, where $dec(\cdot)\colon \{0,1\}^M \to \{0,1\}^K$ is the corresponding ECC decoder, to get a multi-label prediction $\tilde{\boldsymbol{y}} \in \{0,1\}^K$. The simple steps of the framework is shown in Algorithm 1.

Algorithm 1: Error-Correcting Framework

```
• Parameter: an ECC with encoder $enc(\cdot)$ and decoder $dec(\cdot)$; a base multi-label learner $\mathcal{A}_b$
• Training: Given $D = \{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^N$,
    1. ECC-encode each $\boldsymbol{y}_n$ to $\boldsymbol{b}_n = enc(\boldsymbol{y}_n)$;
    2. Return $\tilde{h} = \mathcal{A}_b\Big(\big\{(\boldsymbol{x}_n, \boldsymbol{b}_n)\big\}\Big)$.
• Prediction: Given any $\boldsymbol{x}$,
    1. Predict a codeword $\tilde{\boldsymbol{b}} = \tilde{h}(\boldsymbol{x})$;
    2. Return $h(\boldsymbol{x}) = dec(\tilde{\boldsymbol{b}})$ by ECC-decoding.
```

Algorithm 1 is simple and general. It can be coupled with any block-coding ECC and any base learner $\mathcal{A}_b$ to form a new multi-label classification algorithm. For instance, the ML-BCHRF method (Kouzani and Nasireding, 2009) uses the BCH code (see Subsection 4.3) as ECC, and BR on Random Forest as the base learner $\mathcal{A}_b$. Note that Kouzani and Nasireding (2009) did not describe why ML-BCHRF may lead to improvements in multi-label classification. Next, we show a simple theorem that connects the ECC framework with $\Delta_{0/1}$.

Many ECC can guarantee to correct up to m bit flipping errors in a codeword of length M. We will introduce some of those ECC in Section 4. Then, if Δ_{HL} of $\tilde{h}$ is low, the ECC framework guarantees that $\Delta_{0/1}$ of h is low. The guarantee is formalized as follows.

Theorem 1 *Consider an ECC that can correct up to m bit errors in a codeword of length M. Then, for any T test examples $\{(\boldsymbol{x}_t, \boldsymbol{y}_t)\}_{t=1}^T$, let $\boldsymbol{b}_t = enc(\boldsymbol{y}_t)$. If*

$$\Delta_{HL}(\tilde{h}) = \frac{1}{T}\sum_{t=1}^{T} \Delta_{HL}(\tilde{h}(\boldsymbol{x}_t), \boldsymbol{b}_t) \le \epsilon,$$

then $h = dec \circ \tilde{h}$ satisfies

$$\Delta_{0/1}(h) = \frac{1}{T}\sum_{t=1}^{T} \Delta_{0/1}(h(\boldsymbol{x}_t), \boldsymbol{y}_t) \le \frac{M\epsilon}{m+1}\ .$$

Proof When the average Hamming loss of $\tilde{h}$ is at most ϵ, $\tilde{h}$ makes at most ϵTM bits of error on all $\boldsymbol{b}_t$. Since the ECC corrects up to m bits of errors in one $\boldsymbol{b}_t$, an adversarial has to make at least $m+1$ bits of errors on $\boldsymbol{b}_t$ to make $h(\boldsymbol{x}_t)$ different from $\boldsymbol{y}_t$. The number of such $\boldsymbol{b}_t$ can be at most $\frac{\epsilon TM}{m+1}$. Thus $\Delta_{0/1}(h)$ is at most $\frac{\epsilon TM}{T(m+1)}$. ■

From Theorem 1, it appears that we should simply use some stronger ECC, for which m is larger. Nevertheless, note that we are applying ECC in a learning scenario. Thus, ϵ is not a fixed value, but depends on whether $\mathcal{A}_b$ can learn well from $\tilde{D}$. Stronger ECC usually contains redundant bits that come from complicated compositions of the original bits in $\boldsymbol{y}$, and the compositions may not be easy to learn. The trade-off has been revealed when applying ECC to multi-class classification (Li, 2006). In the next section, we study ECC with different strength and empirically verify the trade-off in Section 6.

4. Review of Classic ECC

Next, we review four ECC designs that will be used in the empirical study. The four designs cover a broad spectrum of practical choices in terms of strength: repetition code, Hamming on repetition code, Bose-Chaudhuri-Hocquenghem code, and low-density parity-check code.

4.1. Repetition Code

One of the simplest ECC is repetition code (REP; Mackay, 2003), for which every bit in $\boldsymbol{y}$ is repeated $\lfloor \frac{M}{K} \rfloor$ times in $\boldsymbol{b}$ during encoding. If M is not a multiple of K, then (M mod K) bits are repeated one more time. The decoding takes a majority vote using the received copies of each bit. Thus, repetition code corrects up to $m_{REP} = \frac{1}{2}\lfloor \frac{M}{K} \rfloor - 1$ bit errors in $\boldsymbol{b}$. We will discuss the connection between REP and the RAKEL algorithm in Section 5.

4.2. Hamming on Repetition Code

A slightly more complicated ECC than REP is called the Hamming code (HAM; Hamming, 1950), which can correct $m_{HAM} = 1$ bit error in $\boldsymbol{b}$ by adding some parity check bits (exclusive-or operations of some bits in $\boldsymbol{y}$). One typical choice of HAM is HAM$(7,4)$, which encodes any $\boldsymbol{y}$ with $K = 4$ to $\boldsymbol{b}$ with $M = 7$. Note that $m_{HAM} = 1$ is worse than $m_{REP} = \frac{1}{2}\lfloor \frac{M}{K} \rfloor - 1$ when M is large. Thus, we consider applying HAM$(7,4)$ on every 4 (permuted) bits of REP. That is, to form a codeword $\boldsymbol{b}$ of M bits from a block $\boldsymbol{y}$ of K bits, we first construct an REP of $4\lfloor M/7 \rfloor + (M \bmod 7)$ bits from $\boldsymbol{y}$; then for every 4 bits in the REP, we add 3 parity bits to $\boldsymbol{b}$ using HAM$(7,4)$. The resulting code will be named Hamming on Repetition (HAMR). During decoding, the decoder of HAM$(7,4)$ is first used to recover the 4-bit sub-blocks in the REP. Then, the decoder of REP (majority vote) takes place.

It is not hard to compute m_{HAMR} by analyzing the REP and HAM parts separately. When M is a multiple of 7 and K is a multiple of 4, it can be proved that $m_{HAMR} = \frac{4M}{7K}$, which is generally better than $m_{REP} = \frac{1}{2}\lfloor \frac{M}{K} \rfloor - 1$. Thus, HAMR is slightly stronger than REP for ECC purposes. We include HAMR in our study to verify whether a simple inclusion of some parity bits for ECC can readily improve the performance for multi-label classification.

4.3. Bose-Chaudhuri-Hocquenghem Code

BCH was invented by Bose and Ray-Chaudhuri (1960), and independently by Hocquenghem (1959). It can be viewed as a sophisticated extension of HAM and allows correcting multiple bit errors. BCH with length $M = 2^p - 1$ has $(M - K)$ parity bits, and it can correct $m_{BCH} = \frac{M-K}{p}$ bits of error (Mackay, 2003), which is in general stronger than REP and HAMR. The caveat is that the decoder of BCH is more complicated than the ones of REP and HAMR.

We include BCH in our study because it is one of the most popular ECC in real-world communication systems. Also, we compare BCH with HAMR to see if a strong ECC can do better for multi-label classification.

4.4. Low-density Parity-check Code

Low-density parity-check code (LDPC; Mackay, 2003) is recently drawing much research attention in communications. LDPC shares an interesting connection between ECC and Bayesian learning (Mackay, 2003). While it is difficult to state the strength of LDPC in terms of a single m_{LDPC}, LDPC has been shown to approach the theoretical limit in some special channels (Gallager, 1963), which makes it a state-of-the-art ECC. We choose to include LDPC in our study to see whether it is worthwhile to go beyond BCH with more sophisticated encoder/decoders.

5. ECC View of RAKEL

RAKEL is a multi-label classification algorithm proposed by Tsoumakas and Vlahavas (2007). Define a k-label-set as a size-k subset of $\mathcal{L}$. Each iteration of RAKEL randomly selects a (different) k-label-set and build a multi-label classifier on the k labels with LP. After running for R iterations, RAKEL obtains a size-R ensemble of LP classifiers. The prediction on each label is done by a majority vote from classifiers associated with the label.

Equivalently, we can draw (with replacement) $M = Rk$ labels first before building the LP classifiers. Then, selecting k-label-sets is equivalent to encoding $\boldsymbol{y}$ by a variant of REP, which will be called RAKEL repetition code (RREP). Similar to REP, each bit $\boldsymbol{y}[i]$ is repeated several times in $\boldsymbol{b}$ since label i is involved in several k-label-sets. After encoding $\boldsymbol{y}$ to $\boldsymbol{b}$, each LP classifier, called k-powerset, acts as a sub-channel that transmits a size-k sub-block of the codeword $\boldsymbol{b}$. The prediction procedure follows the decoder of the usual REP.

The ECC view above decomposes the original RAKEL into two parts: the ECC and the base learner $\mathcal{A}_b$. Next, we empirically study how the two parts affect the performance of multi-label classification.

6. Experiments

We compare RREP, HAMR, BCH and LDPC with the ECC framework on four real-world data sets in different domains: `scene`, `emotions`, `yeast`, and `medical` (Tsoumakas et al., 2010), with the default training/test splitting of the data sets. The statistics of these datasets are shown in Table 1. All the results are reported with the mean and standard

Dataset	K	Training	Testing	Features
scene	6	1211	1196	294
emotions	6	391	202	72
yeast	14	1500	917	103
medical	45	333	645	1449

Table 1: Data Set Characteristics

work	# data sets	codes	channels	base learners
RAKEL (Tsoumakas and Vlahavas, 2007)	3	RREP	k-powerset	linear SVM
ML-BCHRF (Kouzani and Nasireding, 2009)	3	BCH	BR	Random Forest
ML-BCHRF & ML-CRF (Kouzani, 2010)	1	convolution/BCH	BR	Random Forest
this work	4	RREP/HAMR /BCH/LDPC	3-powerset/BR	Random Forest, non-linear and linear SVM

Table 2: Focus of Existing Works under the ECC Framework

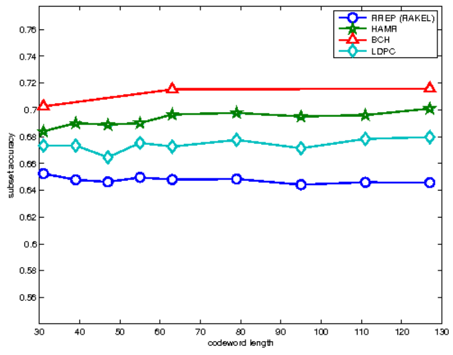
error on the test set over 50 runs. We set RREP with $k = 3$. Then, for each ECC, we first consider 3-powerset with either Random Forest, non-linear support vector machine (SVM), or linear SVM as the multi-class classifier inside 3-powerset. Note that we randomly permute the bits of $\boldsymbol{b}$ and apply an inverse permutation on $\tilde{\boldsymbol{b}}$ for those ECC other than RREP to ensure that each 3-powerset works on diverse sub-blocks. In addition to the 3-powerset base learners, we also consider BR base learners in Subsection 6.3.

We take the default Random Forest from Weka (Hall et al., 2009) with 60 trees. For the non-linear SVM, we use LIBSVM (Chang and Lin, 2001) with the Gaussian kernel and choose (C, g) by cross-validation from $\{2^{-5}, 2^{-3}, \cdots, 2^7\} \times \{2^{-9}, 2^{-7}, \cdots, 2^1\}$. In addition, we use LIBLINEAR (Fan et al., 2008) for the linear SVM and choose the parameter C by cross-validation from $\{2^{-5}, 2^{-3}, \cdots, 2^7\}$.

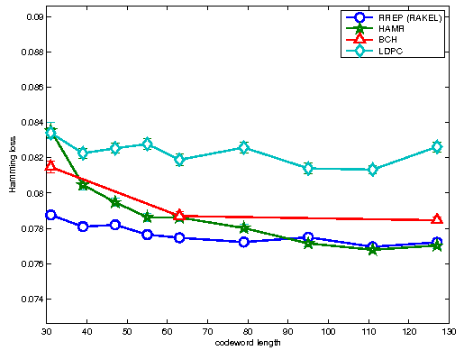
Note that the experiments taken in this paper are generally broader than existing works that are related to multi-label classification with ECC in terms of the data sets, the codes, the "channels", and the base learners, as shown in Table 2. The goal of the experiments is not only to justify that the framework is promising, but also to rigorously identify the best codes, channels and base learners for solving general multi-label classification tasks via ECC.

6.1. Comparison with RAKEL

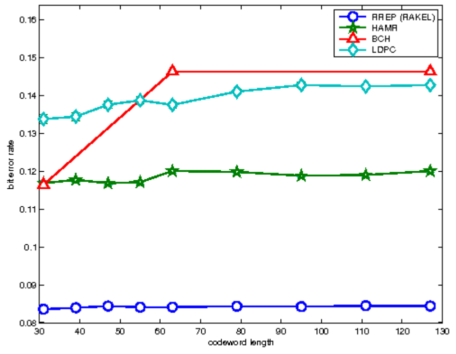
The performance of the ECC framework on the `scene` data set is shown on Figure 1. Here the base learner is 3-powerset with Random Forest. Following the description in Section 5,



(*a*) subset accuracy



(*b*) Hamming loss



(*c*) bit error rate

Figure 1: `scene`: ECC using 3-powerset with Random Forest

RREP with 3-powerset is exactly the same as RAKEL with $k = 3$. The standard error over 50 runs is very small, so the differences shown in the figures are significant. The codeword length M varies from 31 to 127. Note that BCH only allows $M = 2^p - 1$ and thus we conduct experiments of BCH on those codeword lengths. We do not include shorter codewords because their performance is not stable.

We first look at the subset accuracy in Figure 1(*a*). The horizontal axis indicates the codeword length M and the vertical axis is the subset accuracy on the test set. We see that accuracy is slightly increasing with M, except for RAKEL. The differences between $M = 63$ and $M = 127$ are generally small, which implies that a sufficiently large M is good enough for reaching good accuracy.

HAMR achieves consistently higher accuracy than RREP, which verifies that using some parity bits instead of repetition improves the strength of ECC, which in turn improves accuracy. Along the same direction, BCH performs even better than both HAMR and RREP. The superior performance of BCH justifies that ECC is useful for multi-label classification. On the other hand, another sophisticated code, LDPC, gets lower accuracy than BCH and HAMR, which suggest that LDPC may not be a good choice for the ECC framework.

base learner	ECC	scene $M = 63$	yeast $M = 127$	emotions $M = 63$	medical $M = 511$
Random Forest	RREP (RAKEL)	$.648 \pm .001$	$.203 \pm .001$	$.350 \pm .001$	$.334 \pm .001$
	HAMR	$.696 \pm .001$	$.212 \pm .001$	$\mathbf{.356 \pm .002}$	$.343 \pm .001$
	BCH	$\mathbf{.715 \pm .001}$	$.220 \pm .001$	$\mathbf{.372 \pm .002}$	$.547 \pm .001$
	LDPC	$.673 \pm .002$	$.190 \pm .001$	$.340 \pm .003$	$.475 \pm .001$
Gaussian SVM	RREP (RAKEL)	$.690 \pm .000$	$.227 \pm .001$	$.213 \pm .001$	$.623 \pm .001$
	HAMR	$.710 \pm .001$	$\mathbf{.231 \pm .000}$	$.211 \pm .003$	$.627 \pm .001$
	BCH	$\mathbf{.720 \pm .000}$	$\mathbf{.247 \pm .001}$	$.211 \pm .002$	$\mathbf{.655 \pm .001}$
	LDPC	$.693 \pm .001$	$.229 \pm .001$	$.181 \pm .004$	$.614 \pm .001$
Linear SVM	RREP (RAKEL)	$.612 \pm .001$	$.122 \pm .001$	$.255 \pm .002$	$.609 \pm .001$
	HAMR	$.642 \pm .001$	$.137 \pm .001$	$.267 \pm .003$	$.615 \pm .001$
	BCH	$.658 \pm .001$	$.167 \pm .001$	$.285 \pm .003$	$\mathbf{.653 \pm .001}$
	LDPC	$.618 \pm .002$	$.107 \pm .001$	$.248 \pm .005$	$.617 \pm .001$

Table 3: subset accuracy of 3-powerset base learners

Figure 1(b) shows Δ_{HL} versus M for each ECC. Simpler codes such as RREP and HAMR perform better than others. Thus, while a strong code like BCH may guard accuracy better, it can pay more in terms of Δ_{HL}.

As stated in Sections 2 and 3, the base learners serve as the channels in the ECC framework and the performance of base learners may be affected by the codes. Therefore, using a strong ECC does not always improve multi-label classification performance. Next, we verify the trade-off by measuring the bit error rate Δ_{BER} of $\tilde{h}$, , which is defined as the Hamming loss between the predicted codeword $\tilde{h}(\boldsymbol{x})$ and the actual codeword $\boldsymbol{b}$. Higher bit error rate implies that the transformed task is harder.

Figure 1(c) shows the Δ_{BER} versus M for each ECC. RREP has almost constant bit error rate. HAMR also has nearly constant bit error rate, but at a higher value. The bit error rate of BCH is similar to that of HAMR when the codeword is short. But the bit error rate increases with M. The different bit error rates justify the trade-off between the strength of ECC and the hardness of the base learning tasks. With more parity bits, one can correct more bit errors, but may have harder tasks to learn; when using fewer parity bits or even no parity bits, one cannot correct many errors, but will enjoy simpler learning tasks.

Similar results show up in other three data sets with both Random Forest and SVM, as shown in Tables 3 and 4. Based on this experiment, we suggest that using HAMR for multi-label classification will improve the accuracy while maintaining comparable Δ_{HL} with RAKEL. If we use BCH instead, we will get even higher accuracy, but may pay for Δ_{HL}.

6.2. Bit Error Analysis

To further analyze the difference between different ECC designs, we zoom in to $M = 63$ of Figure 1. The instances are divided into groups according to the number of bit errors at that instance. The relative frequency of each group, i.e., the ratio of the group size to the total number of instances, is plotted in Figure 2(a). The average accuracy and Δ_{HL}

base learner	ECC	scene $M = 63$	yeast $M = 127$	emotions $M = 63$	medical $M = 511$
Random Forest	RREP (RAKEL)	**$.077 \pm .000$**	**$.191 \pm .000$**	**$.186 \pm .001$**	$.019 \pm .000$
	HAMR	$.079 \pm .000$	$.194 \pm .000$	**$.191 \pm .001$**	$.019 \pm .000$
	BCH	$.079 \pm .000$	$.196 \pm .000$	**$.190 \pm .001$**	$.015 \pm .000$
	LDPC	$.082 \pm .000$	$.201 \pm .000$	$.192 \pm .001$	$.018 \pm .000$
Gaussian SVM	RREP (RAKEL)	**$.077 \pm .000$**	**$.190 \pm .000$**	$.270 \pm .001$	**$.011 \pm .000$**
	HAMR	$.078 \pm .000$	$.193 \pm .000$	$.279 \pm .001$	**$.011 \pm .000$**
	BCH	$.078 \pm .000$	$.195 \pm .000$	$.289 \pm .001$	**$.011 \pm .000$**
	LDPC	$.080 \pm .000$	$.196 \pm .000$	$.287 \pm .001$	$.013 \pm .000$
Linear SVM	RREP (RAKEL)	$.099 \pm .000$	$.255 \pm .001$	$.238 \pm .001$	$.012 \pm .000$
	HAMR	$.099 \pm .000$	$.247 \pm .001$	$.244 \pm .001$	$.012 \pm .000$
	BCH	$.099 \pm .000$	$.255 \pm .001$	$.243 \pm .002$	$.012 \pm .000$
	LDPC	$.101 \pm .000$	$.301 \pm .001$	$.247 \pm .002$	$.013 \pm .000$

Table 4: Hamming loss of 3-powerset base learners

of each group are also plotted in Figure 2(b) and 2(c). The curve of each ECC forms two peak regions in Figure 2(a). Besides the peak at 0, which means no bit error happens on the instances, the other peak varies from one code to another. The positions of the peaks suggest the hardness of the transformed learning task, similar to our findings in Figure 1(c).

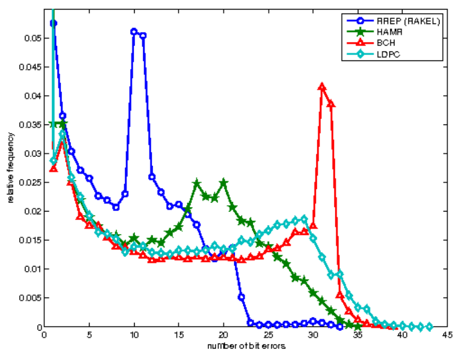
We can clearly see the difference on the strength of different ECC from Figure 2(b). BCH can tolerate up to 15-bit errors, but its accuracy sharply drops to about 0.1 for 16-bit errors. HAMR can correct 6-bit errors perfectly, and its accuracy decreases slowly when more errors occur. Both RREP and LDPC can perfectly correct only 5-bit errors, but LDPC is able to sustain a high accuracy even when there are 16-bit errors. It would be interesting to study the reason behind this long tail from a Bayesian network perspective.

We can also look at the relation between the number of bit errors and Δ_{HL}, as shown in Figure 2(c). The BCH curve grows sharply when the number of bit errors is larger than 15, which links to the inferior performance of BCH over RREP in terms of Δ_{HL}. The LDPC curve grows much slower, but its right-sided peak in Figure 2(a) still leads to higher overall Δ_{HL}. On the other hand, RREP and HAMR enjoy a better balance between the peak position in Figure 2(a) and the growth in Figure 2(c) and thus lower overall Δ_{HL}.

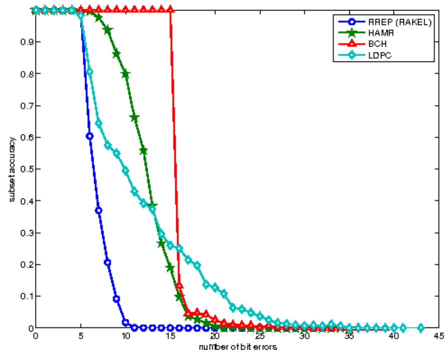
6.3. Comparison with Binary Relevance

In addition to the 3-powerset base learners, we also consider BR base learners, which simply build a classifier for each bit in the codeword space. Note that if we couple the ECC framework with RREP and BR, the resulting algorithm is almost the same as the original BR. For example, using RREP and BR with SVM is equivalent to using BR with bootstrap aggregated SVM.

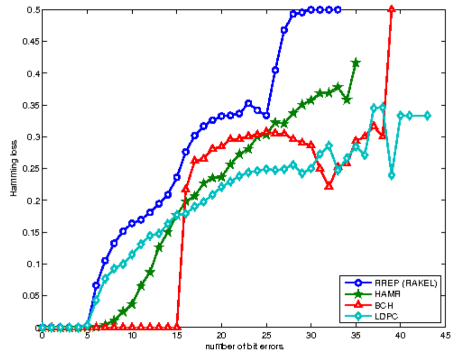
We first compare the performance between the ECC designs using the BR base learner with Random Forest. The result on `scene` is shown in Figure 3. Figure 3(a) shows that the accuracy of BCH and HAMR is superior to other ECC, with BCH being a better choice. RREP (BR), on the other hand, leads to the worst accuracy. The result again justifies the



(a) relative frequency v.s. number of bit errors



(b) subset accuracy v.s. number of bit errors



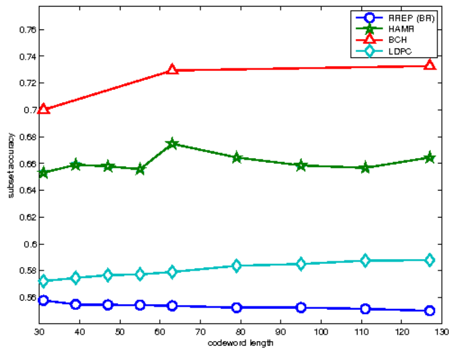
(c) Hamming loss v.s. number of bit errors

Figure 2: `scene`: ECC using 3-powerset with Random Forest and $M = 63$

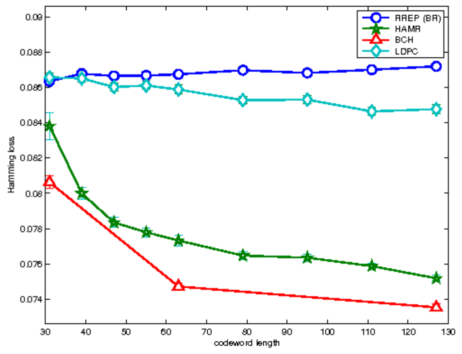
usefulness of coupling BR with ECC instead of only the original $\boldsymbol{y}$. Note that LDPC also performs better than BR, but is not as good as HAMR and BCH. Thus, over-sophisticated ECC like LDPC may not be necessary for multi-label classification.

In Figure 3(b), we present the results on Δ_{HL}. In contrast to the case when using the 3-powerset base learner, HAMR, BCH and LDPC can all achieves better Δ_{HL} than RREP (BR). That is, coupling stronger ECC with the BR base learner can improve both accuracy and Δ_{HL}. In Figure 3(c), we present the bit error rate of the ECC designs. Similar to the results of 3-powerset, we see the trade-off between the strength of ECC and the hardness of the learning task.

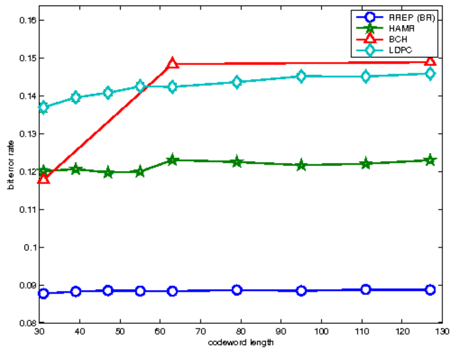
Experiments with both Random Forest and SVM as well as other data sets support similar findings, as shown in Tables 5 and 6. Thus, extending BR by learning some more parity bits and decoding them suitably by ECC is a superior algorithm over the original BR.



(a) subset accuracy



(b) Hamming loss



(c) bit error rate

Figure 3: `scene`: ECC using BR with Random Forest

Comparing Tables 3 and 5, we see that using 3-powerset achieves higher accuracy than using BR in most of the cases. But in terms of Δ_{HL}, as shown in Tables 4 and 6, there is no clear winner between 3-powerset and BR.

7. Conclusion

We presented a framework for applying error-correcting codes (ECC) on multi-label classification. We then studied the use of four classic ECC designs, namely RREP, HAMR, BCH and LDPC. We showed that RREP can be used to give a new perspective of the RAKEL algorithm as a special instance of the framework with k-powerset as the base learners.

We conducted experiments with the four ECC designs on various real-world data sets. The experiments further clarified the trade-off between the strength of ECC and the hardness of the base learning tasks. Experimental results demonstrated that several ECC designs can lead to a better use of the trade-off. For instance, HAMR is superior over RREP for k-powerset base learners, because it leads to a new algorithm that is better than the original RAKEL in terms of subset accuracy while maintaining a comparable level of Hamming loss; BCH is another superior design, which could significantly improve RAKEL in terms of

base learner	ECC	scene $M = 63$	yeast $M = 127$	emotions $M = 63$	medical $M = 511$
Random Forest	RREP (BR)	$.554 \pm .001$	$.173 \pm .001$	$.295 \pm .001$	$.329 \pm .001$
	HAMR	$.675 \pm .002$	$.210 \pm .001$	**.332 ± .002**	$.346 \pm .001$
	BCH	**.729 ± .001**	**.220 ± .001**	**.361 ± .002**	$.560 \pm .001$
	LDPC	$.579 \pm .001$	$.167 \pm .001$	$.295 \pm .002$	$.438 \pm .001$
Gaussian SVM	RREP (BR)	$.639 \pm .000$	$.201 \pm .000$	$.152 \pm .001$	$.617 \pm .001$
	HAMR	$.695 \pm .001$	$.218 \pm .001$	$.205 \pm .003$	$.626 \pm .001$
	BCH	**.719 ± .000**	**.242 ± .001**	$.201 \pm .002$	**.649 ± .001**
	LDPC	$.651 \pm .001$	$.201 \pm .001$	$.167 \pm .001$	$.584 \pm .001$
Linear SVM	RREP (BR)	$.479 \pm .000$	$.042 \pm .001$	$.171 \pm .003$	$.594 \pm .001$
	HAMR	$.574 \pm .001$	$.068 \pm .001$	$.199 \pm .004$	$.610 \pm .001$
	BCH	$.649 \pm .001$	$.101 \pm .001$	$.198 \pm .006$	**.645 ± .001**
	LDPC	$.493 \pm .001$	$.068 \pm .000$	$.153 \pm .006$	$.574 \pm .001$

Table 5: subset accuracy of BR base learners

base learner	ECC	scene $M = 63$	yeast $M = 127$	emotions $M = 63$	medical $M = 511$
Random Forest	RREP (BR)	$.087 \pm .000$	$.192 \pm .000$	**.190 ± .000**	$.019 \pm .000$
	HAMR	**.077 ± .000**	$.191 \pm .000$	$.192 \pm .001$	$.019 \pm .000$
	BCH	**.075 ± .000**	$.193 \pm .000$	**.189 ± .001**	$.015 \pm .000$
	LDPC	$.086 \pm .000$	$.197 \pm .000$	$.196 \pm .001$	$.019 \pm .000$
Gaussian SVM	RREP (BR)	$.078 \pm .000$	**.188 ± .000**	$.253 \pm .000$	**.011 ± .000**
	HAMR	$.078 \pm .000$	**.190 ± .000**	$.258 \pm .001$	**.011 ± .000**
	BCH	$.081 \pm .000$	**.190 ± .000**	$.267 \pm .001$	**.011 ± .000**
	LDPC	$.080 \pm .000$	$.192 \pm .000$	$.256 \pm .000$	$.014 \pm .000$
Linear SVM	RREP (BR)	$.109 \pm .000$	$.428 \pm .000$	$.245 \pm .001$	$.012 \pm .000$
	HAMR	$.105 \pm .000$	$.433 \pm .001$	$.251 \pm .002$	$.012 \pm .000$
	BCH	$.101 \pm .000$	$.418 \pm .000$	$.261 \pm .004$	**.011 ± .000**
	LDPC	$.111 \pm .000$	$.420 \pm .000$	$.265 \pm .004$	$.015 \pm .000$

Table 6: Hamming loss of BR base learners

subset accuracy. When compared with the traditional BR algorithm, we showed that using a stronger ECC like HAMR or BCH can lead to better performance in terms of both subset accuracy and Hamming loss.

The results justify the validity and usefulness of the framework when coupled with some classic ECC. An interesting future direction is to consider adaptive ECC like the ones studied for multi-class classification (Schapire, 1997; Li, 2006).

Acknowledgments

We thank the anonymous reviewers for valuable suggestions. This work is supported by National Science Council of Taiwan via the grant NSC 99-2628-E-002-017.

References

R. C. Bose and D. K. Ray-Chaudhuri. On a class of error correcting binary group codes. *Information and Control*, 3(1):68–79, 1960.

C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier. On label dependence in multi-label classification. In *Proc. of MLD'10*, pages 5–12, 2010.

T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

R. G. Gallager. *Low Density Parity Check Codes, Monograph.* M.I.T. Press, 1963.

M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1):10–18, 2009.

R. W. Hamming. Error detecting and error correcting codes. *Bell System Technical Journal*, 26(2):147–160, 1950.

A. Hocquenghem. Codes correcteurs d'erreurs. *Chiffres*, 2:147–158, 1959.

D. Hsu, S. Kakade, J. Langford, and T. Zhang. Multi-label prediction via compressed sensing. In *Proc. of NIPS'09*, pages 772–780, 2009.

A. Z. Kouzani. Multilabel classification using error correction codes. In *Proc. of ISICA'10*, pages 444–454, 2010.

A. Z. Kouzani and G. Nasireding. Multilabel classification by BCH code and random forests. *International Journal of Recent Trends in Engineering*, 2(1):113–116, 2009.

L. Li. Multiclass boosting with repartitioning. In *Proc. of ICML'06*, pages 569–576, 2006.

D. J. C. Mackay. *Information Theory, Inference and Learning Algorithms.* Cambridge University Press, 1st edition, 2003.

R. E. Schapire. Using output codes to boost multiclass learning problems. In *Proc. of ICML'97*, pages 313–321, 1997.

C. E. Shannon. A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423, 623–656, 1948.

F. Tai and H.-T. Lin. Multi-label classification with principle label space transformation. In *Proc. of MLD'10*, pages 45–52, 2010.

G. Tsoumakas and I. Vlahavas. Random k-labelsets: An ensemble method for multilabel classification. In *Proc. of ECML'07*, pages 406–417, 2007.

G. Tsoumakas, J. Vilcek, and E. S. Xioufis. MULAN: A Java library for multi-label learning, 2010.