Indian Institute of Technology Bombay

CS 754 Spring' 25, Advanced Image Processing

Final Project Report

*On*

# Group Testing for Multi-label Classification

**Submitted By:**

**1.** Abhay Raj (24M0747)

**2.** Ayush Pratap Singh (24M0767)

## Introduction

Multilabel Classification (MLC) refers to the supervised learning problem where an input instance can belong to multiple classes at once. This setting arises in several real-world applications such as image and video annotation, medical tagging, recommendation systems, and text categorization. In such cases, the label space can be very large, and each instance typically belongs to only a small subset of these labels, making the output vector sparse.

As the number of labels grows to tens or hundreds of thousands, we move into the domain of **Extreme Multilabel Classification (XMC)**. This brings its own set of challenges: designing scalable models that can handle such high dimensional output spaces, ensuring fast and memory-efficient prediction for real-time applications, and exploiting correlations or hierarchies among labels to improve accuracy.

Traditional approaches like One-vs-All (OvA), label embeddings, and compressed sensing-based methods become either computationally expensive or lose important label structure at these scales. More recent techniques aim to overcome these limitations through smarter label space reduction, structure-aware grouping, or deep learning-based modeling.

Here, we focus on Group Testing (GT) based methods, which aim to reduce the number of classifiers by organizing labels into groups. Within Group Testing there are several methods to implement it. We studied 3 of those methods. These include:

- **MLGT:** using randomized groupings with error-correcting properties.
- **NMF-GT:** using data-dependent group construction.
- **He-NMFGT:** exploiting hierarchical structure among labels.

## An Idea of Group Testing

Group testing is a classical problem that originated during World War II, primarily to efficiently identify infected individuals in large populations with a minimal number of tests. Instead of testing each individual separately, multiple samples are pooled together and tested as a group. If the group test is negative, all individuals in that group are declared negative. If positive, at least one individual in the group is positive, and further testing is needed.

Formally, suppose we have ***n*** items, among which at most ***d*** **<<** ***n*** are "defective" (i.e., positive). The goal of group testing is to identify all defective items using the fewest number of tests. A test queries a subset of items and returns a binary outcome indicating whether any item in the subset is defective.

There are two primary models:

- **Adaptive Group Testing:** Tests are designed sequentially, where each test can depend on the results of previous tests.
- **Non-Adaptive Group Testing:** All tests are designed beforehand, and executed in parallel, which is often more practical for large-scale applications.

Mathematically, non-adaptive group testing can be represented using a binary test matrix $M \in \{0,1\}^{\{T \times n\}}$, where $T$ is the number of tests. The $(i, j)$ –th entry is 1 if the $j$-th item is included in the $i$-th test, and 0 otherwise. The test outcomes can then be modeled as:

$$y = M \ \vee \ x$$

where $x \in \{0,1\}^n$ is the unknown indicator vector of defective items, and the operation ∨ denotes the bitwise OR.

## Group Testing for Extreme Classification

Extreme classification involves learning classifiers in settings with an extremely large number of labels (e.g., hundreds of thousands or millions). Directly modeling each label independently becomes computationally expensive both during training and inference.

The idea of applying group testing to extreme classification stems from the analogy between identifying defective items and identifying relevant labels:

- In group testing, we aim to identify a small subset of defective items.
- In extreme classification, for a given input, only a small subset of labels are typically relevant.

By modeling labels as items and using a group-testing-inspired design, one can organize labels into groups (tests) and predict group activations instead of individual labels. This offers several advantages:

- **Efficiency:** Predicting a small number of group activations requires fewer computations than scoring all labels independently.
- **Scalability:** The number of group tests can be much smaller than the total number of labels, enabling the handling of extremely large label spaces.
- **Robustness:** Group structures can help in denoising and improving generalization, as group-level predictions can capture label correlations.

## Disjunctness Properties for Group Testing-based Classification

In classical group testing, the design of the pooling matrix plays a crucial role in enabling efficient recovery of defective (positive) items from the outcomes of group tests. A fundamental concept in this context is that of disjunctness.

***Disjunctness:*** *An $m \times d$ binary matrix $A$ is called $k$-disjunct if the support of any of its columns is not contained in the union of the supports of any other $k$ columns.*

In the language of group testing, this condition ensures that no non-defective item appears only in positive tests. This property not only guarantees that the defective set (active labels) can be uniquely identified but also provides a simple decoding rule: *any item that appears in a negative test is non-defective, whereas an item that appears exclusively in positive tests is defective.*

## Error Correction via $(k, e)$ −Disjunctness

*An $m \times d$ binary matrix $A$ is called $(k, e)$ −disjunct} (i.e., $k$ −disjunct and $e$ −error detecting), where $e \geq 1$, if for every set $S \subseteq [d]$ with $|S| \leq k$ and for every $i \notin S$, the support of column $i$ differs from the union of supports of the columns in $S$ in more than $e$ positions. Formally,*

$$\left| \operatorname{supp}(A(i)) \setminus \bigcup_{j \in S} \operatorname{supp}(A(j)) \right| > e,$$

*where* $(A(i)) = \{ r \in [m] | A_{r,i} = 1 \}$ *denotes the support of the* $\mathrm{i}$ *−th column of matrix A.*

This property ensures that the matrix $A$ can detect up to $e$ errors in the measurement outcomes (test results) and can correct up to e/2 errors. Such robustness is crucial in real-world applications where group test results may be noisy or unreliable, such as in noisy multi-label classification or biological testing.

## Approaches

### Group Testing via Error Correcting Codes (MLGT)

**Key Idea:** The MLGT (MultiLabel Group Testing) approach draws inspiration from classical group testing to reduce the number of classifiers needed. Instead of learning d classifiers—one for each label—a smaller number m of group classifiers is trained using a binary matrix $A \in \{0,1\}^{\{m \times d\}}$ that defines how labels are grouped. The compressed label vector is computed as $z = A \vee y$, where $y$ is the true label vector and $\vee$ denotes the elementwise OR.

The main innovation lies in how the matrix $A$ is constructed. MLGT uses random code-based constructions such as **concatenated Reed-Solomon codes and expander graphs** to ensure that A satisfies the $(k, e)$ −disjunct property. This means that up to k active labels can be identified even when up to $e/2$ test outcomes are incorrect. The decoding procedure is simple and linear in time, using bitwise comparisons with the support of the group classifier outputs.

**Strengths:**

- Reduces classifier count from $d$ to $m = O(k \log d)$.
- Simple decoding using combinatorial support intersections.
- No need to solve complex optimization or regression problems.

**Limitations:**

- Random grouping may place unrelated labels together.
- Classifier performance can degrade if label structure is ignored.

## Data-Dependent Grouping and Hierarchical Partitioning (NMF-GT and He-NMFGT)

**Key Idea:** To address the limitations of random grouping, Ubaru et al. extended MLGT by introducing data-aware grouping. The NMF-GT approach uses low-rank symmetric Non-negative Matrix Factorization (NMF) on the label co-occurrence matrix $C \in \{R\}^{\{d \times d\}}$ to learn a soft clustering of labels. The final group testing matrix formed ensures that labels that frequently co-occur are grouped together.

**He-NMFGT** adds a hierarchical flavour to the grouping by recursively partitioning the label set using a greedy graph partitioning strategy based on label co-occurrence statistics. Within each partition, NMF is applied independently to construct group matrices. This leads to better scalability and accuracy in datasets where labels follow a natural hierarchy.

Decoding in both methods is based on SAFFRON (Sparse-grAph codes Framework For gROup testiNg), a bipartite graph-based decoding algorithm that improves recovery performance over traditional GT decoding.

**Strengths:**

- Groups labels based on actual co-occurrence patterns.
- Improves classifier accuracy and reduces false positives.
- Hierarchical version reduces runtime complexity to $O(k \, log \, d)$ where $k$ is sparsity of labels.

**Limitations:**

- Requires computing and storing co-occurrence matrices.
- Symmetric NMF involves factorizing a $d \times d$ label co-occurrence matrix, which becomes computationally expensive for large $d$, especially in unpartitioned settings.

# Experiments Performed

## Datasets

For our experiments, we have considered some of the popular publicly available multilabel datasets put together in **The Extreme Classification Repository** by Manik Varma and his team. The papers we implemented also used the same datasets for their experiments.
http://manikvarma.org/downloads/XC/XMLRepository.html

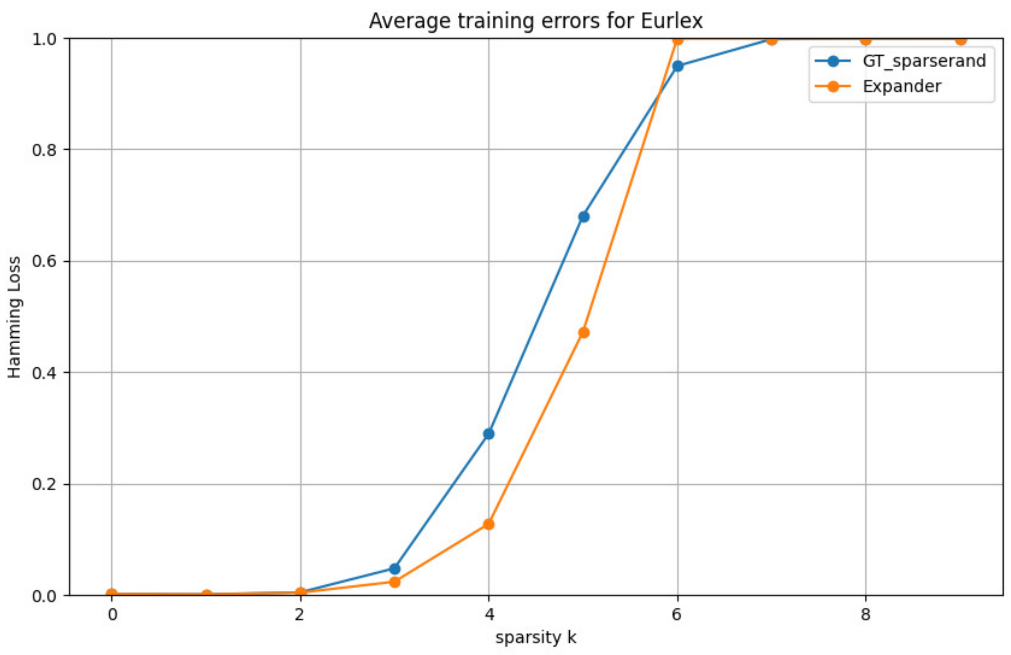
## Experiments

Our experiments can be categorised into following approaches:

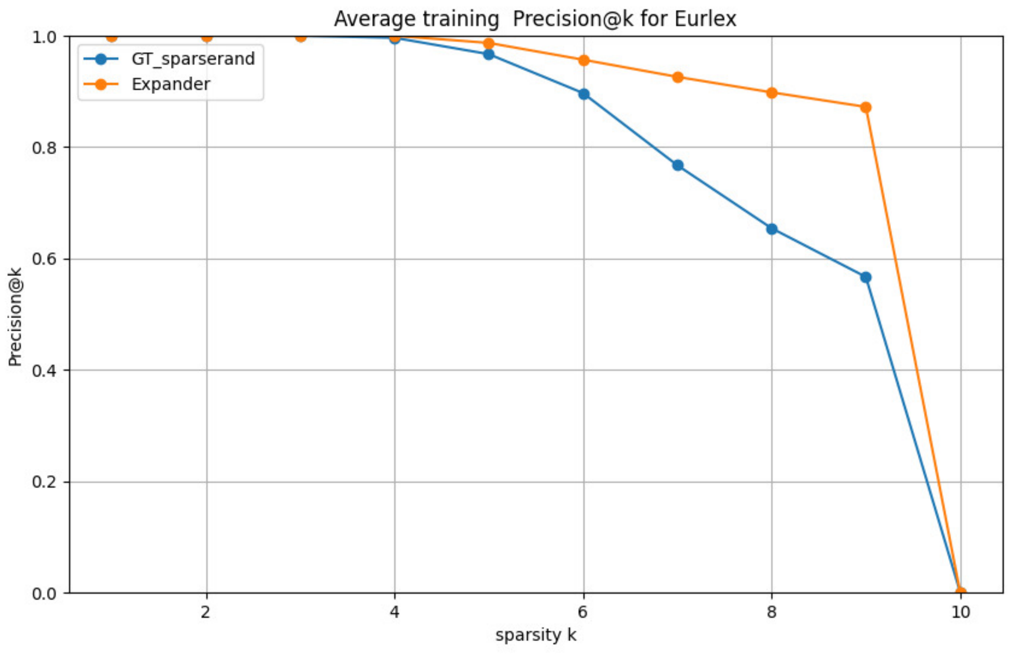
- MLGT
- NMF-GT
- He-NMFGT

### MLGT Experiment 1: Average training and test errors and Precison@k versus sparsity k for EurLex(d=3993, k=5.3) for 2 constructions of GT matrix i.e., Sparse Random and Expander Graphs

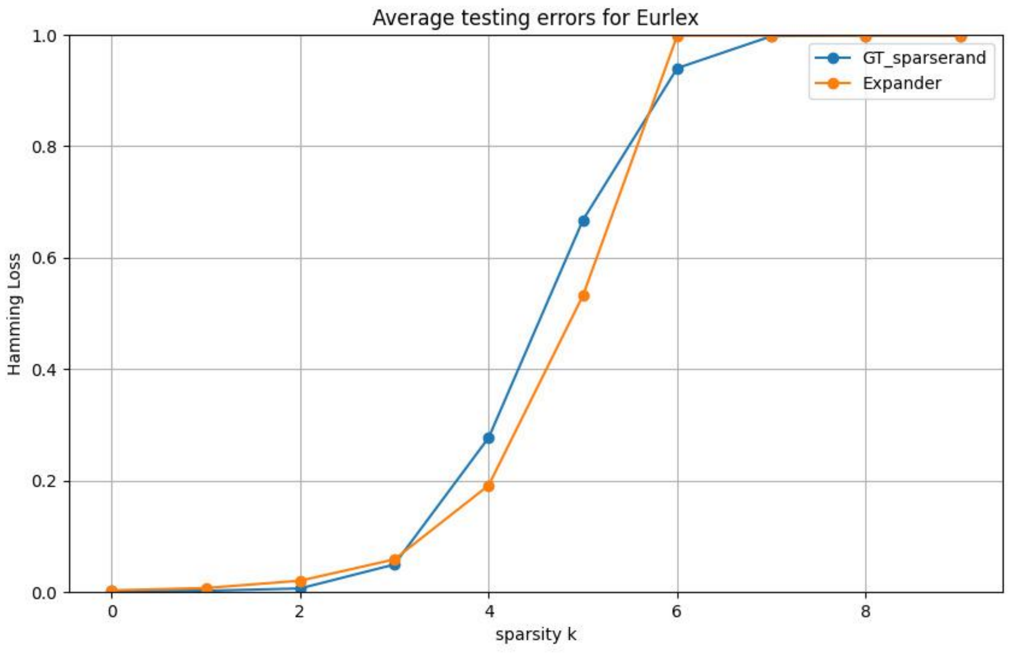
We got the following results:



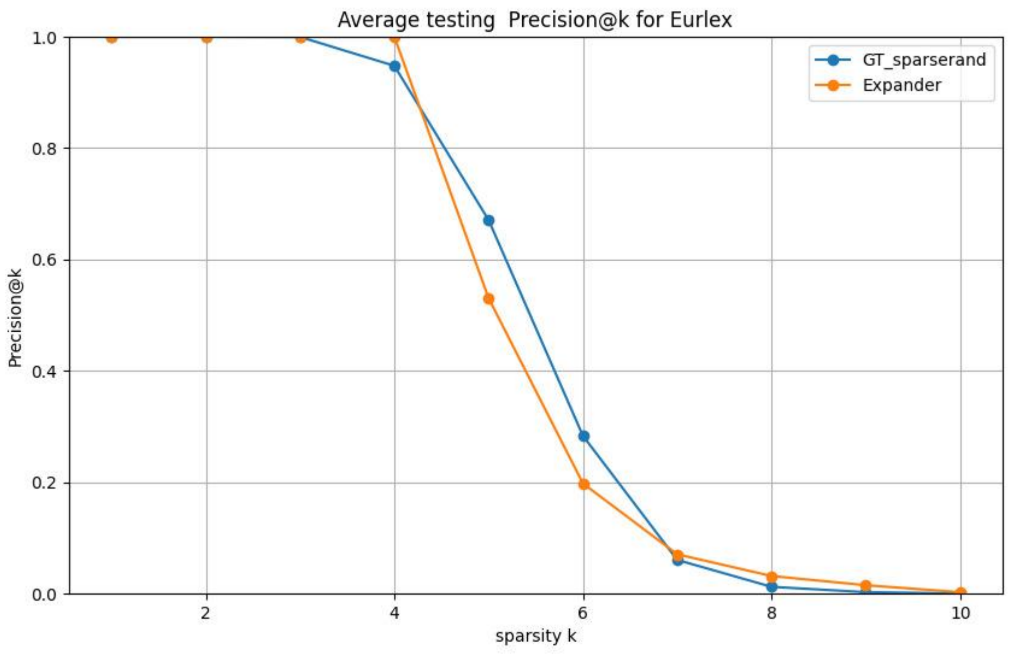
Training Error (Hamming Loss) vs Sparsity(k) for Eurlex dataset



Precision@k vs Sparsity(k) for Eurlex training dataset



Testing Error (Hamming Loss) vs Sparsity(k) for Eurlex dataset



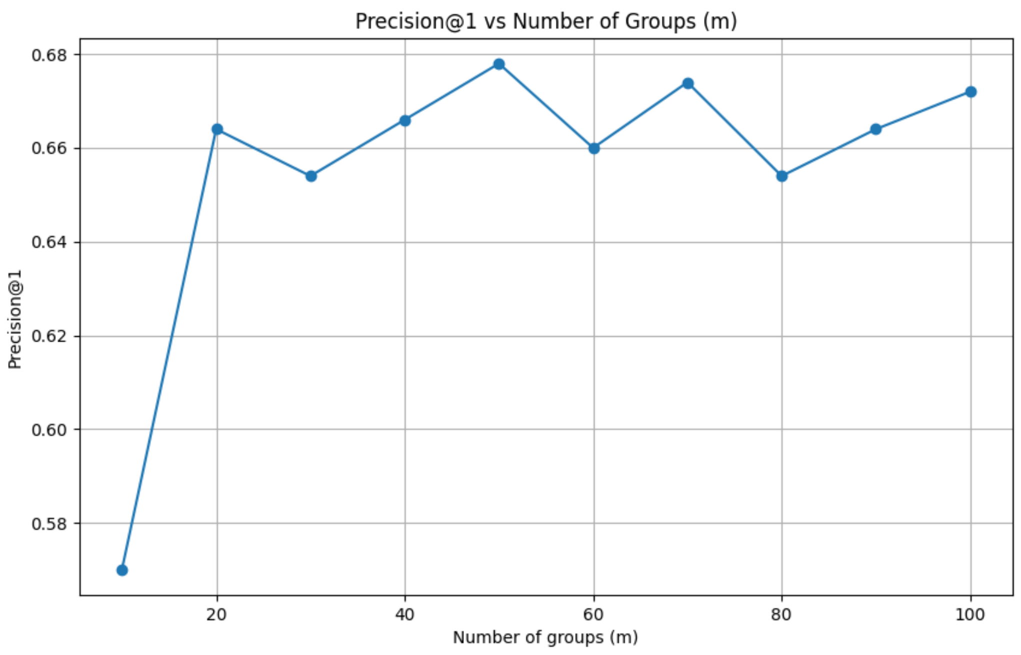
Precision@k vs Sparsity(k) for Eurlex testing dataset

## Observations:

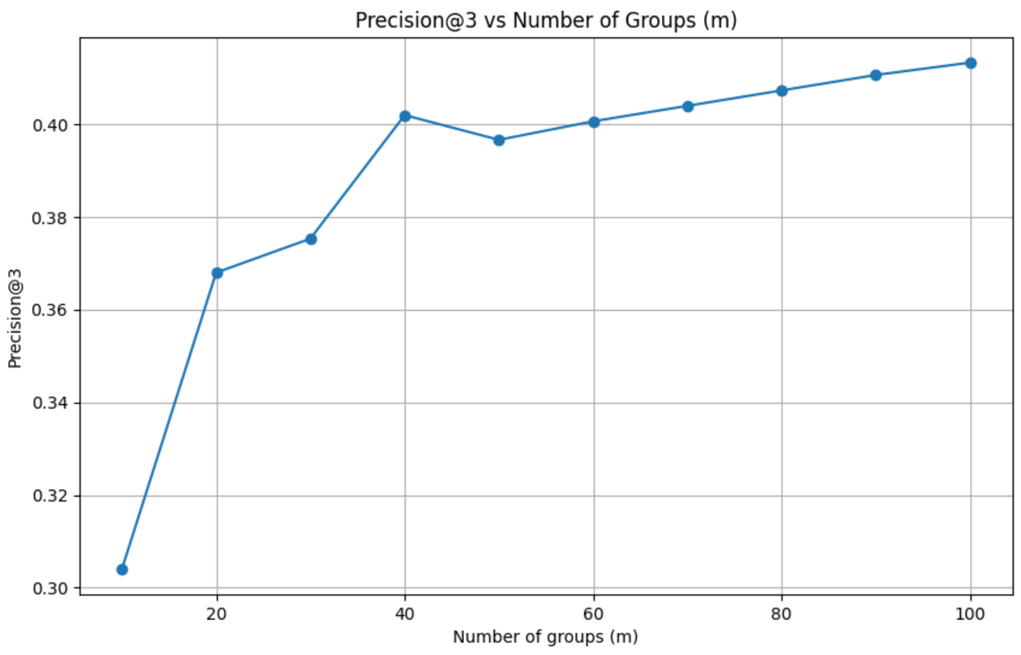
As can be seen from the above plots that **Hamming Loss increases as sparsity (k) increases for both training and testing datasets**. This is expected as if each sample has more active labels (higher sparsity), the model has more chances to make mistakes (both false positives and false negatives). That is, more active labels may lead to more potential mismatches.

Secondly, we see that **Precision@k values are decreasing with increasing sparsity(k)**. This is also obvious due to the fact that as sparsity increases, there are more relevant labels per instance, but the model has to spread its prediction capacity over more possible true labels. It becomes harder to include the most relevant ones in the top-*k* predictions.

## NMF-GT Experiment 2: : P@1and P@3 for test data instances for Bibtex dataset (d = 159, k = 2.40) as a function of number of groups m



Precision@1 vs Number of groups(m)

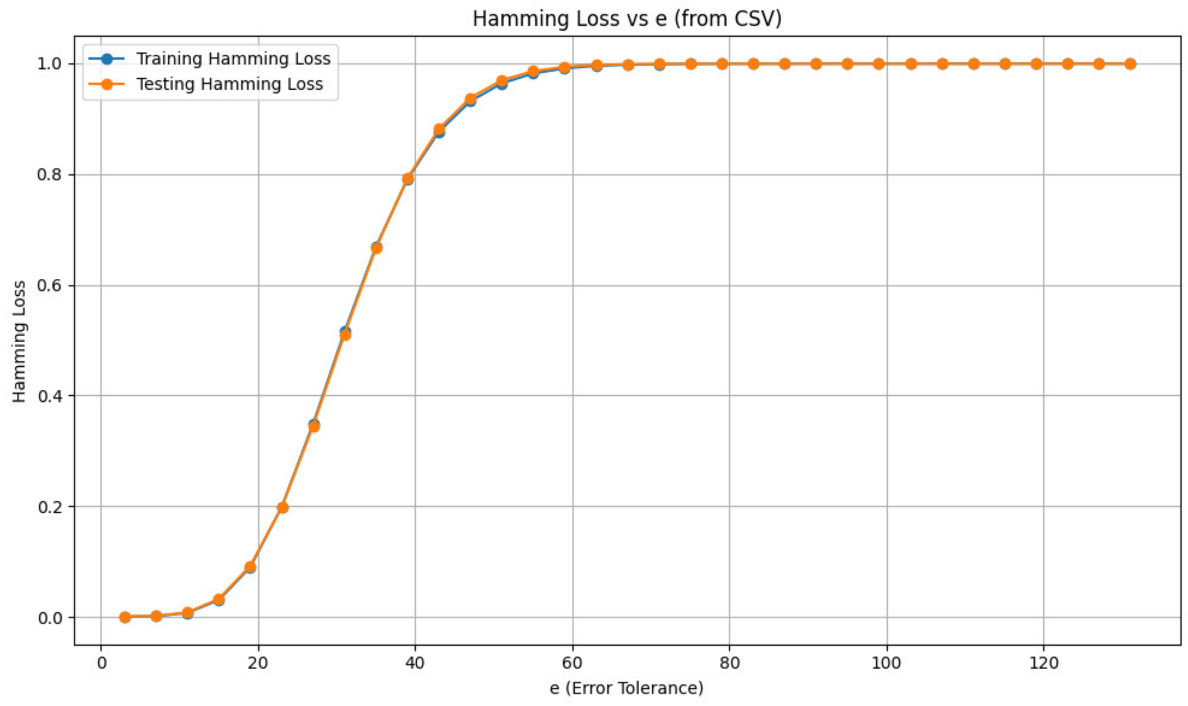


Precision@3 vs Number of groups(m)

## Observations

We can see from the above plots that **as number of groups (m) increases, both Precision@1 and Precision@3 improves**. This result aligns with the theory. With fewer groups, multiple labels may map to the same group, causing label confusion. With more groups, such collisions become rarer. This improvement saturates after a point due to overfitting.

## MLGT Experiment 3: Hamming loss vs e (Error Tolerance) for Sparse Random GT Matrix Construction



## Observations

We ran this code for Eurlex Dataset (d = 3993, k = 5.32) with sparse random GT matrix. Interestingly, we found that the hamming loss was way more than expected. For sparse random construction of GT matrix, it was mentioned in the paper that $O(klogd)$ will work as optimum value for e. But for that value of e, we were getting very large hamming losses.

## References

1. Ubaru, Shashanka, and Arya Mazumdar. "Multilabel classification with group testing and codes." In *International Conference on Machine Learning*, pp. 3492-3501. PMLR, 2017.

2. Ubaru, Shashanka, Sanjeeb Dash, Arya Mazumdar, and Oktay Gunluk. "Multilabel classification by hierarchical partitioning and data-dependent grouping." *Advances in Neural Information Processing Systems* 33 (2020): 22542-22553.