applied sciences

MDPI

Article

Distributed Online Multi-Label Learning with Privacy Protection in Internet of Things

Fan Huang, Nan Yang, Huaming Chen, Wei Bao and Dong Yuan *

School of Electrical and Information Engineering, The University of Sydney, Sydney, NSW 2008, Australia
* Correspondence: dong.yuan@sydney.edu.au

Abstract: With the widespread use of end devices, online multi-label learning has become popular as the data generated by users using the Internet of Things devices have become huge and rapidly updated. However, in many scenarios, the user data are often generated in a geographically distributed manner that is often inefficient and difficult to centralize for training machine learning models. At the same time, current mainstream distributed learning algorithms always require a centralized server to aggregate data from distributed nodes, which inevitably causes risks to the privacy of users. To overcome this issue, we propose a distributed approach for multi-label classification, which trains the models in distributed computing nodes without sharing the source data from each node. In our proposed method, each node trains its model with its local online data while it also learns from the neighbour nodes without transferring the training data. As a result, our proposed method achieved the online distributed approach for multi-label classification without losing performance when taking existing centralized algorithms as a reference. Experiments show that our algorithm outperforms the centralized online multi-label classification algorithm in F1 score, being 0.0776 higher in macro F1 score and 0.1471 higher for micro F1 score on average. However, for the Hamming loss, both algorithms beat each other on some datasets, and our proposed algorithm loses 0.005 compared to the centralized approach on average, which can be neglected. Furthermore, the size of the network and the degree of connectivity are not factors that affect the performance of this distributed online multi-label learning algorithm.

Keywords: distributed learning; online learning; multi-label classification

Citation: Huang, F.; Yang, N.; Chen, H.; Bao, W.; Yuan, D. Distributed Online Multi-Label Learning with Privacy Protection in Internet of Things. *Appl. Sci.* **2023**, *13*, 2713. https://doi.org/10.3390/app13042713

Academic Editor: Luis Javier García Villalba

Received: 10 January 2023
Revised: 12 February 2023
Accepted: 15 February 2023
Published: 20 February 2023

1. Introduction

Label classification is a general task in machine learning, and multi-label classification builds on this by allowing each instance to be able to belong to a collection of multiple categories. Many mature multi-label classification algorithms have been developed, with numerous experiments demonstrating their performance. However, these approaches cannot be applied straightforwardly in real-world application scenarios due to various issues. For example, in the era of big data, where the Internet of Things (IoT) devices are widespread, privacy protection and big data communication cost need to be addressed by specialized technologies. Then, online multi-label classification (OMLC) learning algorithms are proposed to tackle these challenges by directly processing and discarding the incoming data. Because of the efficiency and fast processing of data by these algorithms, such algorithms are widely used in real-world applications, such as Twitter, Facebook, Instagram postings and RDF Site Summary (RSS) feeds [1].

Although OMLC algorithms can handle most real-life scenarios very well, in recent years, data generation has changed with the era of big data. In real-world applications, a large amount and variety of data are generated by a wide range of computing devices, such as home appliances, surveillance cameras, monitoring sensors, actuators, displays, vehicles and so on [2]. However, traditional online multi-label learning algorithms [3–5] can only process online data and train models independently at each computing node, which is

incompatible with the data generation approach, and the additional communication costs are inevitable. In addition, unnecessary transfer steps inevitably increase the chance of being attacked. For example, multiple channels, such as WiFi, could easily attack user data, thus affecting its performance [6]. Moreover, avoiding user data leakage is also very important. Many technologies such as Privacy Enhancing Technologies or private synthetic data generators have been proposed and applied to protect users' data [7]. This type of security risk can be completely avoided if it is possible to avoid redundant raw data transfers.

Current mainstream online learning is devoted to solving performance problems caused by, for example, concept drift, missing features, missing labels and data imbalance [8]. Extending online learning to distributed contexts has not received sufficient attention. Moreover, research on multi-label classification problems in the distributed domain lacks consideration of online data use, and there is no practical distributed online multi-label learning method.

We propose a novel distributed approach for the OMLC problem (DOML), which updates the local model by allowing each node in the network to self-coordinate by transmitting only parameters. With this approach, each node can achieve the performance of a centralized multi-label classification algorithm by having data interact with its neighbours only and inferring the global model without exposing local data. In our proposed algorithm, the interaction of the metric models and the self-renewal process of each node are parallel, so the efficiency gains from the distributed algorithm are guaranteed. We compared our proposed algorithm with traditional batched learning algorithms and online learning algorithms using a classical dataset. We confirmed that our proposed algorithm has a performance that does not lose out to traditional centralised algorithms in a distributed scenario where the source data are not transferred.

The remainder of the article is organized as follows. The next section discusses the current state of the multi-label classification problem and introduces the distributed least squares iterative method. This is followed by an explanation of the method for transforming real-world IoT networks into an abstract graph representation. The subsequent section shows the mathematical derivation of the distributed constrained optimization problem and gives an iterative update method. The next section shows the computational flow of the DOML algorithm. The next section presents the comparative experiments and discusses the results, while the conclusions section summarizes this work and then points out future research directions.

2. Related Work

In this section, we first introduce the background, category and representative algorithms of multi-label classification. After that, the second subsection will focus on online learning, which is more suitable for realistic scenarios and relevant to our proposed approach. Finally, in the third subsection, we introduce the distributed least squares iterative approach, which is the basis for our proposed DOML algorithm.

2.1. Multi-Label Classification

The multi-label classification problem aims to associate an unseen object with a predefined set of labels depending on its features. This kind of problem has various real-world applications, including but not limited to text categorization [9–13], bioinformatics [14,15], medical diagnosis [16], image/scene and video categorization [17], genomics, map labelling [18], marketing, multimedia, emotion, music categorization, etc. In 2007, Tsoumakas and Katakis grouped the existing multi-label classification methods into two main categories: (a) problem transformation (PT) methods and (b) algorithm adaptation (AA) methods [19].

As the name indicates, PT methods are generally concerned with converting a multi-label classification problem into many single-label classification problems. One of the most frequently used methods is the Binary Relevance method [20], which was proposed by Boutell et al. in 2004. This algorithm breaks down the multi-label learning problem into a set of independent binary classification problems, where each binary classification problem corresponds to a possible label in the label space, and then combines the results of these binary classification problems. Because it can handle data with many labels in a linear proportion to the number of labels, it is appropriate for various practical purposes. However, the Binary Relevance method, confirmed in [21,22], essentially disregarded the interdependence of labels.

On the other hand, algorithmic adaptation methods directly deal with multi-label data by extending specific learning algorithms. The popular AA methods rely on k-Nearest Neighbour (kNN) [23–28], decision tree (DT) [29,30], support vector machines (SVM) [31,32], neural networks (NN) [33,34] and others. For example, multi-Label k-Nearest Neighbour (ML-KNN) [26] is a multi-label lazy learning method derived from the traditional K-nearest neighbours (KNN) algorithm. The algorithm determines the K-nearest neighbours of each unseen occurrence in the training set. Following that, the label set for unseen examples is chosen using the maximum a posteriori (MAP) principle utilizing statistical information extracted from the label sets of these surrounding instances.

Beyond these two basic methods, there is another ensemble approach that combines several classifiers to achieve better performance. Benefiting from these approaches, the performance of traditional multi-label classification problems has become very reliable. However, solving the multi-label classification problem in real-world scenarios is a new challenge.

2.2. Online Learning

In many real-world scenarios, the cost of storing large amounts of data is often a significant expense. However, these costs could be saved by directly processing online data in real time. In contrast to the traditional batched learning methods, online learning needs to continuously update the training model according to the newly collected data. In the OMLC problem, there are some common approaches based on PT, AA, and Ensembles of Multi-Label Classification (EMLCs) [21,35,36].

Some popular PT methods for online multi-label learning in stationary streaming data include OSML-ELM [37], dw-ELM [38], RLS-Multi [39], and AMLCM [40]. Similar to batched learning, the advantage of PT methods in online multi-label learning tasks is the ability to apply off-the-shelf single-label classifiers to multi-label scenarios directly. The AA approaches, such as HTPS [41] and iSOUP-Tree [42], are another idea to solve this problem which aims to make itself compatible with the multi-label stream data classification problem. On the other hand, the EMLCs approach works by combining multiple weaker classifiers into one stronger classifier which is more straightforward to scale and parallelise than a single approach.

In 2020, Gong et al. [43] pointed out that existing OMLC studies have lacked analysis of loss functions and have not considered label dependencies. Nevertheless, metric learning can be used to optimize this problem. Their proposed online metric learning (OML) method uses the projection of instances and labels to a lower dimension for comparison and then uses an efficient optimization algorithm to learn a metric using the large marginal principle. Although this method significantly improves performance, it relies on the pre-trained model.

All OMLC methods mentioned in this subsection are listed in Table 1.

Table 1. Online Multi-Label Classification methods.

Methods	Type of Methods	Description
OSML-ELM [37]	PT	OSM-ELM for online learning
dw-ELM [38]	PT	dw-ELM for OMLC
RLS-Multi [39]	PT	For imbalanced online data
AMLCM [40]	PT	AMRule problem for OMLC
HTPS [41]	AA	Multi-label data stream
iSOUP-Tree [42]	AA	Regression for classification
OML [43]	EMLCs	Enhance label dependencies

2.3. Distributed Least-Squares Iterative Methods

In traditional distributed algorithms such as federated learning, a centralized server is necessary to collect and aggregate models from each independent node. Because of this, a single node in such a setting can often only access its own database. However, this is often inefficient in real-world scenarios considering various factors such as bandwidth, time and privacy. A distributed algorithm that does not require a centralized server can solve this problem significantly. In the field of multi-label classification problems, many algorithms involve solving linear least-squares systems to be used to determine differences between samples. Therefore, the study of distributed least-squares iterative methods is a major point of our research.

The distributed least-squares iterative methods can be divided into four methods: Distributed Multi-Splitting method, Distributed Modified Conjugate Gradient Least-Squares method, Distributed Least Mean Squares method, and Distributed Recursive Least-Squares method. Each of these algorithms has advantages and disadvantages in different aspects due to the differences in the calculation methods [44]. The original least-squares problem is defined as $\mathbf{Ax} = \mathbf{b}$, and the matrix $\mathbf{A}$ and vector $\mathbf{b}$ are split and distributed to multiple nodes. The following descriptions all refer to this definition.

Distributed Multi-Splitting Method works by parallelising the stationary iterative method based on a well-known single splitting. Parallelisation uses a space decomposition method that splits the matrix $\mathbf{A}$ into blocks to split the original problem into more minor local problems [45,46]. As the algorithm splits matrix $\mathbf{A}$ and assigns computational tasks to different nodes, each iteration needs to pass through all the nodes in the network.

Distributed Modified Conjugate Gradient Least-Squares method is based on a distributed variant of the Modified Conjugate Gradient Least-Squares method. A common approach to solving the least-squares problem is to minimize it by solving the normal equation. The resulting method, the Conjugate Gradient Least-Squares method, is often used as the basic iterative method for solving least-squares problems. Yang and Brent [47] improved the Conjugate Gradient Least-Squares method and described an improved conjugate gradient least-squares method to reduce inner product global synchronization points and improve parallel performance. The method can also be applied to distributed scenarios and is called the Distributed Modified Conjugate Gradient Least-Squares method.

Distributed Least Mean Squares Method allows each node to make an estimation based on local data and calculate the optimal global solution by exchanging the estimation results only with its neighbours. The advantage of this method is that only the exchange of local data is necessary. However, as a cost, it has a relatively slow convergence rate.

Distributed Recursive Least-Squares Method was developed by Sayed and Lopes [48]. This method achieves an exact recursive solution by appealing to collaborative techniques. It requires circular paths in the network to be computed node-by-node. This method has the advantage of a fixed number of iterations, but its drawback is the need to exchange large dense matrices between nodes.

3. Problem Definition

This section describes the scenario in which our proposed DOML algorithm is used and abstracts the real-world IoT environment into a mathematical representation.

In a real-world network environment, all computer devices can communicate with other devices on the same network. Geographically nearer devices can often transfer information directly, while more remote devices must pass through multiple devices to share data. Furthermore, in general, information transfer between devices is bidirectional. Thus, it could be supposed that there is a network composed of m computing nodes in which two nodes that can communicate directly with each other are called neighbours. The value of variable m should be a positive integer. An undirected graph $\mathbb{G}$ can describe the connection of the entire network, where the vertices represent the computing nodes and the edges indicate the neighbours. The proposed network structure is shown in Figure 1. In a real network, multiple terminal devices can continuously obtain and exchange data with neighbouring devices in real-time. Each terminal device is considered a separate node in our supposed network and obtains local data X_i and Y_i. Neighbour nodes can exchange data with each other, thus forming our supposed undirected graph network.

Each node will continuously obtain streaming data of instances with their corresponding labels, and refer to each instance-label pair as an example. Let $x_t \in \mathbb{R}^d$ denote an instance collected at time t and its corresponding labels by $y_t \in \{1,0\}^q$, where d and q denote the number of features and labels of the data, respectively. The instances and their corresponding labels accumulated over a while on a node i are denoted as $X_i \in \mathbb{R}^{n_i \times d}$ and $Y_i \in \{0,1\}^{n_i \times q}$, respectively. The index number of i is counting up from 1. Note that each row of the X_i is the transpose of an instance x_t, and the same row of its corresponding Y_i is also the transpose of the corresponding labels y_t for the single instance x_t. n_i denotes the number of examples accumulated on the node i. The overall instance matrix $X = \mathrm{col}\{X_1, X_2, \ldots, X_m\}$, representing the block matrix in the shape of $[X_1', X_2', \ldots, X_m']'$, similarly its corresponding label matrix, is $Y = \mathrm{col}\{Y_1, Y_2, \ldots, Y_m\}$. X and Y composed the overall dataset across the entire network where the instance and the label belong to the same example and should appear at the same position of the matrix.

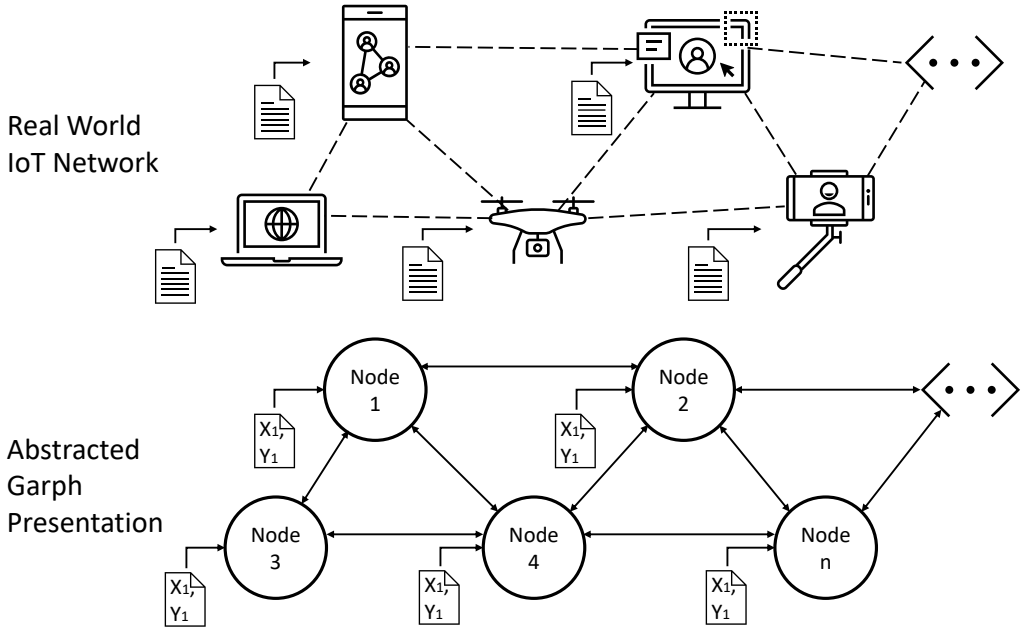

Figure 1. The real-world network scenario with its corresponding abstract graph presentation.

4. Distributed OMLC Algorithm

This section first introduces an overview of an approach to construct distributed constrained optimization problems by using a graph-theoretic approach for the DOML algorithm. Subsequently, the second subsection describes the mathematical derivation of the DOML algorithm and gives the specific method for iterative updates.

4.1. Abstracted Problem Formulation

The task of multi-label classification is to find a function $h : \mathcal{X} \to \mathcal{Y}$ mapping from the instance space to the label space in the most efficient way, where $\mathcal{X} \in \mathbb{R}^d$ stands for the d-dimensional instance space and $\mathcal{Y} \in \{1, 0\}^q$ stands for the label space with q labels. Considering that the data are stored in a centralized way, then the initial problem can be reduced to a global least-squares problem. The global problem for the whole network is described as projecting each instance to the label space as close as possible through a projection matrix $P \in \mathbb{R}^{d \times q}$,

$$P = \arg\min_{P \in \mathbb{R}^{d \times q}} \frac{1}{2} \left\| P'X' - Y' \right\|_F^2 \tag{1}$$

In a distributed scenario, the whole network is split by m nodes and each node only has access to its local data. As a result, all instances and labels are reorganized into m local instance sets, $X_1, X_2, \ldots, X_m$ and m local label sets, $Y_1, Y_2, \ldots, Y_m$. Under this scenario, the global problem is reshaped as

$$P = \arg\min_{P \in \mathbb{R}^{d \times q}} \sum_{i=1}^{m} \frac{1}{2} \| X_i P - Y_i \|_F^2 \tag{2}$$

We provide each node with its own projection matrix P_i. When all nodes have the same P_i, this problem is equivalent to Equation (2):

$$\begin{aligned} P^* = &\arg\min_{P_i \in \mathbb{R}^{d \times q}} \sum_{i=1}^{m} \frac{1}{2} \| X_i P_i - Y_i \|_F^2 \\ &s.t.\ P_1 = P_2 = \ldots = P_m \end{aligned} \tag{3}$$

At this time, an optimal projection matrix P^*, which fits the overall dataset, is obtained. The problem of interest in this article is to find an iterative update of P_i for each node and provide an algorithm that fits the distributed scenario.

4.2. Distributed Discrete-Time Update

To make it easier to handle the problem state in the last section, according to graph theory [49], the network connection is described in a matrix form to better express the constraints in (3). We define W to be the adjacency matrix for the graph $\mathbb{G}$ and the ij-th entry of W is w_{ij}, representing the weighting parameter for the edge between the i-th and j-th node, which should be a positive real number or zero. The range of index numbers i and j starts from 1 and ends with the number of nodes in the network. Zero of w_{ij} means no connection between the node i and j. Let D denote the degree matrix with each i-th entry equal to the sum of the i-th row of W, where $d_i = \sum_{j=1}^{m} w_{ij}$.

Then, the Laplacian matrix L of the graph $\mathbb{G}$ is given by $L = D - W$. Note that L is symmetric and positive semi-definite. To describe the constraints of Equation (3) in terms of the Laplacian matrix, $\bar{L} = L \otimes I_d$ is defined, where $\otimes$ denotes the Kronecker product. To adapt $\bar{L}$, it could be defined that $\bar{P} = \text{col}\{P_1, P_2, \ldots, P_m\}$, which summarized the projection matrices for all nodes, and $\bar{X} = \text{diag}\{X_1, X_2, \ldots, X_m\}$, which represents a block diagonal matrix with the ith diagonal block equal to X_i.

Then, the original problem can be transferred to an equivalent global form as

$$\begin{aligned} P^* = &\arg\min_{P_i \in \mathbb{R}^{d \times q}} \frac{1}{2} \| \bar{X}\bar{P} - Y \|_F^2 \\ &s.t.\ \bar{L}\bar{P} = \mathbf{0} \end{aligned} \tag{4}$$

We use the Lagrange multiplier method to solve the problem state in Equation (4). The Lagrangian function is defined as

$$G(\bar{P}, B) = \frac{1}{2}\|\bar{X}\bar{P} - Y\|_F^2 I_q + B'\bar{L}\bar{P} \tag{5}$$

where $B \in \mathbb{R}^{md \times q}$, which is the Lagrange multiplier.

Setting the partial derivatives of $G(\bar{P}, B)$ with respect to the elements of $\bar{P}$ and B to zero, respectively, gives

$$\begin{aligned} \frac{\partial G(\bar{P}, B)}{\partial \bar{P}} &= \bar{X}'(\bar{X}\bar{P} - Y) + \bar{L}'B = 0 \\ \frac{\partial G(\bar{P}, B)}{\partial B} &= \bar{L}\bar{P} = 0 \end{aligned} \tag{6}$$

Since L is symmetric, it could be deduced that $L = L'$. In order to break up the whole system into parts, Equation (6) is disassembled into blocks as nodes. Introduce an additional matrix β_i for each node to split B as $B = \text{col}\{\beta_1, \beta_2, \ldots, \beta_m\}$; then, the following equivalent equations could be obtained:

$$\begin{aligned} (X_i'X_iP_i - X_i'Y) + \sum_{j \in \mathcal{N}_i} w_{ij}(\beta_i - \beta_j) = 0 \\ \sum_{j \in \mathcal{N}_i} w_{ij}(P_i - P_j) = 0 \end{aligned} \tag{7}$$

where $\mathcal{N}_i$ stands for the neighbours of node i.

In order to solve discrete-time updates between multiple nodes in the system, a common way is to find the change of the state matrix, $\dot{P}_i$ and $\dot{\beta}_i$. Then, replace them with $\frac{P_i(t+1)-P_i(t)}{\Delta t}$ and $\frac{\beta_i(t+1)-\beta_i(t)}{\Delta t}$, respectively. t stands for the round of update. Motivated by the discrete update ideas of Wang et al. [50], we define that

$$\begin{aligned} \dot{P}_i(t) = &-(X_i'X_iP_i(t) - X_i'Y) \\ &- \sum_{j \in \mathcal{N}_i} w_{ij}(\beta_i(t) - \beta_j(t)) \\ \dot{\beta}_i(t) = &\sum_{j \in \mathcal{N}_i} w_{ij}(P_i(t) - P_j(t)) \end{aligned} \tag{8}$$

with the mix use of $\dot{P}_i(t+1)$, $\dot{\beta}_i(t+1)$ and $\dot{P}_j(t)$, $\dot{\beta}_j(t)$, for $j \in \mathcal{N}_i$ on the right-hand side of Equation (8). In addition, we replace $\dot{P}_i$ and $\dot{\beta}_i$ with $\frac{P_i(t+1)-P_i(t)}{\Delta t}$ and $\frac{\beta_i(t+1)-\beta_i(t)}{\Delta t}$, respectively, where $\Delta t = \frac{1}{d_i}$. For easy of derivation, k_i will be used in subsequent content to replace $\frac{1}{d_i}$. We give the result of the discrete-time update as

$$\begin{aligned} P_i(t+1) = P_i(t) &- k_i[X_i'X_iP_i(t+1) - X_i'Y_i] \\ &- k_i \sum_{j \in \mathcal{N}_i} w_{ij}(\beta_i(t+1) - \beta_j(t)) \\ \beta_i(t+1) = \beta_i(t) &+ k_i \sum_{j \in \mathcal{N}_i} w_{ij}(P_i(t+1) - P_j(t)) \end{aligned} \tag{9}$$

Rewrite Equation (9) in the state form, moving all terms with $(t+1)$ moment to the left-hand side and all terms with (t) moment to the right-hand side,

$$\begin{bmatrix} P_i(t+1) \\ \beta_i(t+1) \end{bmatrix} = E_i^{-1}F_i \tag{10}$$

where

$$E_i = \begin{bmatrix} I_d + k_i X_i' X_i & I_d \\ -I_d & I_d \end{bmatrix} \tag{11}$$

$$F_i = \begin{bmatrix} P_i(t) + k_i \sum_{j \in \mathcal{N}_i} w_{ij} \beta_j(t) + k_i X_i' Y_i \\ \beta_i(t) - k_i \sum_{j \in \mathcal{N}_i} w_{ij} P_j(t) \end{bmatrix} \tag{12}$$

Lemma 1. *The term E_i in Equations (10) and (11) is always invertible.*

Remark 1. *In Equation (10), at each node i, the X_i and Y_i terms originate only from node i itself. Therefore, this update method satisfies the distributed requirements. As this method performs the update in discrete-time, it is therefore capable of being used in an online manner.*

We provide a fully distributed OMLC algorithm based on the mathematical model discussed before that avoids the transfer of private data between individual nodes.

5. Our Proposed Algorithm

Once the network connection between nodes is set up, each node needs an iterative update, as shown in Figure 2 and Algorithm 1. The steps in Algorithm 1 will be explained in detail in the next paragraph. Followed by the update, each node would periodically broadcast the local P_i and B_i to all neighbour nodes and continuously listen for neighbouring nodes' broadcasted P_i and B_i. There are no strict requirements for broadcast frequency.

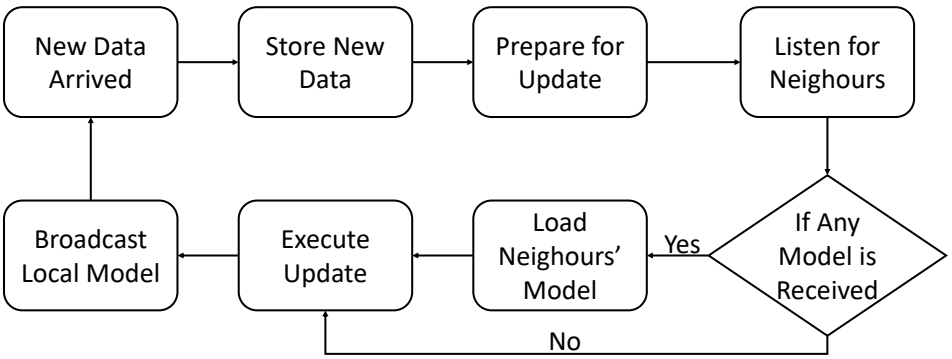

Figure 2. The workflow of the DOML algorithm.

In Algorithm 1, the algorithm requires pre-defined balanced parameters and weighted parameters with positive real numbers, one by default. The maximum instance capacity s depends on the memory of the node device and will define the maximum number of instances that could be cached on this edge device. In line 2, X_i and Y_i denote the instances accumulated by the current node. Their initial values could be empty and represent that the node has not yet cached any instances. In line 4, the current node receives the P_j and β_j transmitted from the neighbouring nodes. Lines 5 through 6 indicate that the end device of the current node fetched a set of pairwise instances (x_t, y_t) at time t and added them to the cache. After this, each node will calculate the corresponding E_i and F_i based on Equations (11) and (12), with i denoting the number of the current node. If the memory requirements of X_i and Y_i exceed the capacity of node i, X_i^s and Y_i^s will be employed to replace X_i and Y_i by discarding some of the old data. After that, referring to line 16, the algorithm updates and finally broadcasts the updated results to all neighbouring nodes according to Equation (10), ending the current iteration.

Algorithm 1 Single node update of the distributed OMLC algorithm

Input: weighted parameters $\{w_{ij} \in \mathbb{R} | w_{ij} > 0\}_{(i,j)\in G}$, maximum instance capacity s;
Initialize: $X_i = Y_i = \varnothing$;
for $t = 1, 2, \ldots,$ **do**
  Receive $\begin{bmatrix} P_j(t) \\ \beta_j(t) \end{bmatrix}$ pairs for all neighbours $j \in \mathcal{N}_i$
  Receive pairwise instances: (x_t, y_t)
  Append x_t to X_i and y_t to Y_i
  if $n_i < s$ **then**
    Calculate $E_i(X_i)$ by Equation (11)
    Calculate $F_i(X_i, Y_i, P_j(t), \beta_j(t))$ by Equation (12)
  else
    $X_i^s \leftarrow X_i$ {select s newest instance from X_i to X_i^s}
    $Y_i^s \leftarrow Y_i$ {select s newest instance from Y_i to Y_i^s}
    Calculate $E_i(X_i^s)$ by Equation (11)
    Calculate $F_i(X_i^s, Y_i^s, P_j(t), \beta_j(t))$ by Equation (12)
  end if
  Update $\begin{bmatrix} P_i(t+1) \\ \beta_i(t+1) \end{bmatrix}$ pair by Equation (10)
  Broad cast $\begin{bmatrix} P_i(t+1) \\ \beta_i(t+1) \end{bmatrix}$ to all neighbours $j \in \mathcal{N}_i$
end for

6. Experiment Setup

Since our DOML algorithm is developed to deal with a new scenario, our experimental setup will evaluate our proposed DOML algorithm through the performance compared with traditional approaches and large-scale network impact. All experiments are conducted on a desktop with Intel Core i5-7500 @ 3.40 GHz and 16 GB RAM, running on Python 3.8 with a Windows 10 platform.

6.1. Evaluation Metrics

This section measures the performance of our proposed DOML algorithm in three dimensions: accuracy, precision and recall. Therefore, in the experiment, the Hamming loss and F1 score are measured because they are the most commonly used evaluation criteria for assessing multi-label classification problems, for different cases to evaluate our proposed DOML algorithm.

6.1.1. Hamming Loss

Hamming loss is a commonly used measure of accuracy in multi-label classification problems. It is also the most intuitive measure of a classifier's performance. A low Hamming loss means that the trained classifier has higher accuracy.

6.1.2. F1 Scores

In multi-label classification problems, the F1 score is a special kind of F-score, which is often used to evaluate the combined level of precision and recall of a classifier. A good classifier corresponds to a high F1 score, which means that the classifier can identify more positive examples and the identified positive examples have higher confidence.

6.1.3. Datasets and Baseline Methods

We use seven benchmark datasets: CAL500 [51], Corel5k [52], Emotions [53], Enron, Medical [54], scene [20] and yeast [55] as Table 2 shown to perform the experiment. These benchmark datasets are collected from several different domains in the real world. In addition, the dimensions of the datasets are also varied to test the adaptability of the algorithm to different situations.

We compare the performance of our algorithm with the Online metric learning for Multi-Label classification (OML) [43] method, which has been verified to outperform other state-of-art online multi-label prediction methods. In addition, take this as a baseline method of the OMLC algorithm. In our experiment, we use the same parameters as the author: M is set to 100,000, m is set to 0.00001, and k is set to 10.

Moreover, the ML-KNN [26] is a lazy learning algorithm that has been verified to outperform some well-established multi-label learning algorithms. Hence, ML-KNN is taken as the baseline of a batched learning algorithm, where both OML and our algorithm are based on it but use a different distance measurement method in place of the original Euclidean distance.

The performance of the algorithms is measured by Hamming loss, Macro-F1 and Micro-F1, which can show the comprehensive performance of the algorithms in solving a multi-label classification problem.

Table 2. Statistics of multi-label benchmark datasets.

Datasets	Number of Instances	Number of Features	Number of Labels	Domain
CAL500	502	68	174	music
Corel5k	5000	499	374	images
Emotions	593	72	6	music
Enron	1702	1001	53	text
Medical	978	1449	45	text
scene	2407	294	6	image
yeast	2417	103	14	biology

7. Results

Comparative experiments have been conducted on the overall performance of the proposed algorithm and the adaptability of the method for large-scale networks. In this subsection, we present a specific analysis of each of these two experiments.

7.1. Performance Comparison with Centralized Methods

This experiment compares the performance of different algorithms when processing the same number of datasets, of which DOML uses a fully connected network with three nodes to achieve the best results. In order to evaluate the overall performance of DOML with the current well-performance online multi-label algorithms and batched multi-label algorithms, the entire dataset will be equally distributed among the three nodes in DOML. In this way, each piece of data in the dataset will be called only once by each algorithm. In DOML, all parameters w_{ij} are set to one, respectively, while s is set to infinity. The algorithm only runs an iteration when a new instance is received.

7.1.1. Hamming Loss Evaluation

Table 3 and Figure 3 compare the Hamming loss of the KNN algorithm, the OML algorithm and the DOML algorithm for different datasets.

Table 3. Hamming loss of different algorithms with full benchmark datasets.

	KNN	OML	DOML
CAL500	0.1325	**0.1310**	0.1359 ± 0.0007
Corel5k	**0.0093**	0.0094	0.0114 ± 0.0003
Emotions	0.2937	0.2979	**0.2851 ± 0.0161**
Enron	0.0522	0.0610	**0.0479 ± 0.0006**
Medical	0.0251	0.0292	**0.0242 ± 0.0050**
scene	**0.0989**	0.1820	0.2294 ± 0.0375
yeast	**0.1210**	0.1254	0.1362 ± 0.0008

The numbers in bold highlight the best performing of the three algorithms.

We can see that the three algorithms have comparable performance in most datasets. In this experiment, we simulated DOML with three independent nodes, i.e., each node has

only one-third of the source data. Therefore, each node in the DOML algorithm lacks source data compared to the KNN and OML algorithms. The experimental results demonstrate that our proposed DOML algorithm can compensate for the disadvantage of the lack of source data at a single node through node communication.

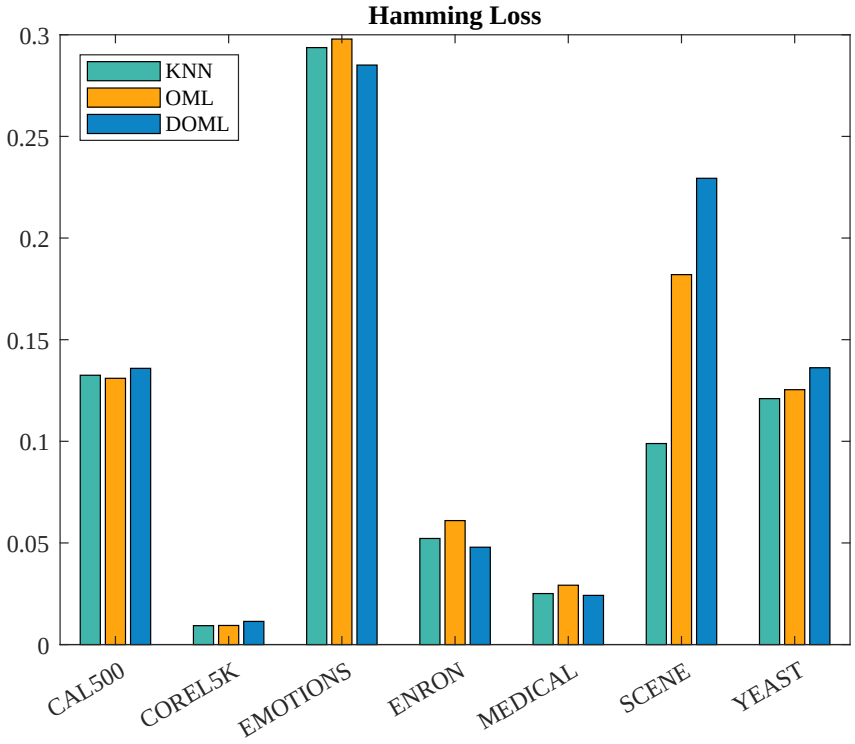

Figure 3. Hamming loss of different algorithms with full benchmark datasets.

7.1.2. F1 Score Evaluation

Table 4 and Figure 4 compare the differences in F1 scores between traditional KNN, OML and DOML methods for different datasets.

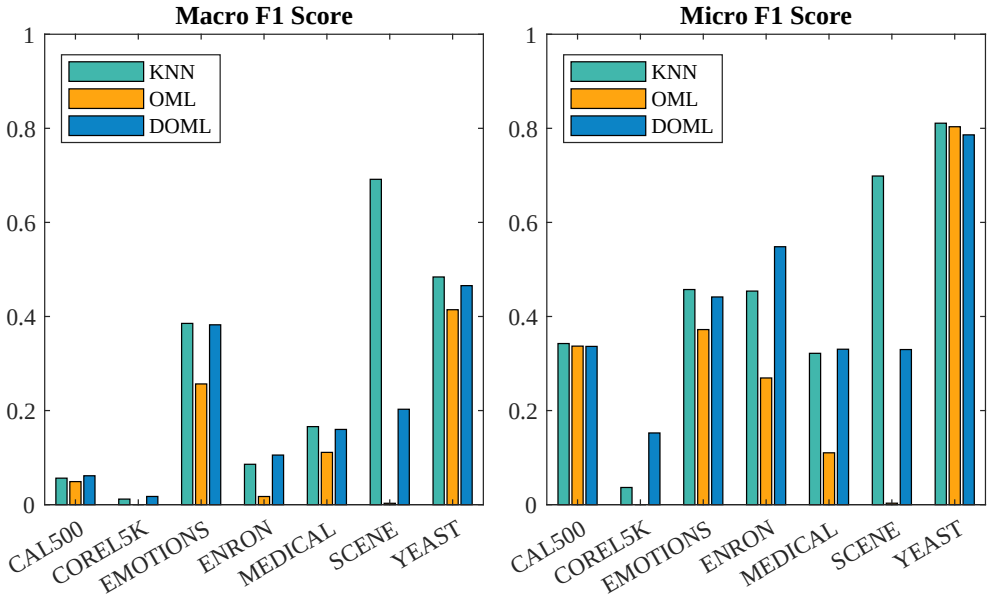

Figure 4. F1 score of different algorithms with full benchmark datasets.

Table 4. F1 Score of different algorithms with full benchmark datasets.

	Macro F1 Score			Micro F1 Score		
	KNN	**OML**	**DOML**	**KNN**	**OML**	**DOML**
CAL500	0.0563	0.0490	**0.0614 ± 0.0035**	**0.3425**	0.3370	0.3364 ± 0.0157
Corel5k	0.0118	0.0000	**0.0174 ± 0.0066**	0.0365	0.0000	**0.1524 ± 0.0246**
Emotions	**0.3853**	0.2566	0.3823 ± 0.0646	**0.4573**	0.3722	0.4415 ± 0.0661
Enron	0.0858	0.0173	**0.1054 ± 0.0082**	0.4540	0.2693	**0.5483 ± 0.0161**
Medical	**0.1659**	0.1111	0.1599 ± 0.1037	0.3217	0.1102	**0.3304 ± 0.2344**
scene	**0.6916**	0.0029	0.2028 ± 0.0857	**0.6986**	0.0031	0.3297 ± 0.0973
yeast	**0.4840**	0.4144	0.4656 ± 0.0177	**0.8109**	0.8034	0.7861 ± 0.0050

The numbers in bold highlight the best performing of the three algorithms.

KNN algorithms generally have higher F1 scores in all datasets, both Macro F1 scores and Micro F1 scores. This is because the KNN method is a batch lazy learning algorithm. Compared to online learning, the training of KNN methods always considers all the data in the dataset, so KNN methods should have better performance than online algorithms. Compared with the KNN method, the traditional OML method shows a significant disadvantage in terms of F1 scores. However, our proposed DOML method does not lose performance in F1 scores in most datasets but also has better training results in COREL5K and ENRON datasets. Our proposed algorithm improves the recall and precision of the model through the mechanism of distributed communication compared to the traditional OML.

7.2. Performance Analysis in a Large-Scale Distributed Environment

This experiment will evaluate the performance of DOML with different numbers of nodes and different network connections. This experiment is set up with 20 and 100 nodes and evaluates different networks with 25% connectivity, 50% connectivity and full connectivity, respectively.

A connectivity of n% represents that n% of the edges are used compared to a fully connected network with the same number of nodes. We also evaluated the performance shown by the OML algorithm when having the same dataset with a single node of DOML as a reference. Other parameters remain the same as in the previous experiment. At the same time, we make the traditional OML algorithm use the data with the same amount as a single node in the DOML algorithm as a reference to eliminate the effect of different network sizes causing the variation of data volume in a single node.

7.2.1. Hamming Loss Evaluation

Tables 5 and 6 show the Hamming loss of the DOML algorithm with different network connectivity for a network of 20 nodes and 100 nodes, respectively. This result is also summarized in Figure 5 for comparison. At the same time, we evaluate the performance of the traditional OML algorithm with the same number of datasets as a single node in the DOML algorithm as a reference.

Table 5. Hamming loss of DOML in a 20-node network with different connectivity and OML with the same amount of instances as each node.

	DOML			OML
	25%	**50%**	**100%**	**-**
CAL500	0.1511 ± 0.0098	**0.1486 ± 0.0112**	0.1489 ± 0.0112	0.1499
Corel5k	0.0130 ± 0.0015	0.0126 ± 0.0015	0.0125 ± 0.0015	**0.0110**
Emotions	0.3556 ± 0.0545	**0.3531 ± 0.0479**	0.3886 ± 0.0792	0.3564
Enron	0.0556 ± 0.0034	0.0559 ± 0.0035	**0.0539 ± 0.0032**	0.0639
Medical	0.0306 ± 0.0014	0.0311 ± 0.0017	0.0314 ± 0.0019	**0.0300**
scene	0.2663 ± 0.0259	0.2651 ± 0.0272	**0.2647 ± 0.0275**	0.2919
yeast	0.1687 ± 0.0344	0.1609 ± 0.0265	0.1605 ± 0.0287	**0.1332**

The numbers in bold highlight the best performing of the three algorithms.

Table 6. Hamming loss of DOML in a 100-node network with different connectivity and OML with same amount of instances as each node.

	DOML			OML
	25%	**50%**	**100%**	**-**
CAL500	0.1954 ± 0.0450	0.1954 ± 0.0450	0.1954 ± 0.0450	**0.1499**
Corel5k	0.0145 ± 0.0051	0.0147 ± 0.0051	0.0145 ± 0.0049	**0.0124**
Emotions	**0.4422 ± 0.1048**	**0.4422 ± 0.1048**	**0.4422 ± 0.1048**	0.4695
Enron	0.1027 ± 0.0380	0.0955 ± 0.0311	0.0881 ± 0.0239	**0.0665**
Medical	0.1133 ± 0.0833	0.1133 ± 0.0833	0.1133 ± 0.0833	**0.0300**
scene	**0.2656 ± 0.0846**	0.2667 ± 0.0857	0.2743 ± 0.0933	0.2919
yeast	0.2094 ± 0.0751	0.2009 ± 0.0673	0.2099 ± 0.0756	**0.1310**

The numbers in bold highlight the best performing of the three algorithms.

First, we can observe that the DOML algorithm obtains nearly the same Hamming loss results for various datasets with different network connectivity. Therefore, we can conclude that the Hamming loss of the DOML algorithm is not affected by the connectivity of the training network. However, as the size of the network increases, the number of training nodes increases, and the Hamming loss of the DOML algorithm increases. This means that the DOML algorithm loses accuracy when more nodes are involved in the training network. As the network size increases, the number of data assigned to each node is reduced, thus making each node have more difficulty with obtaining accurate results.

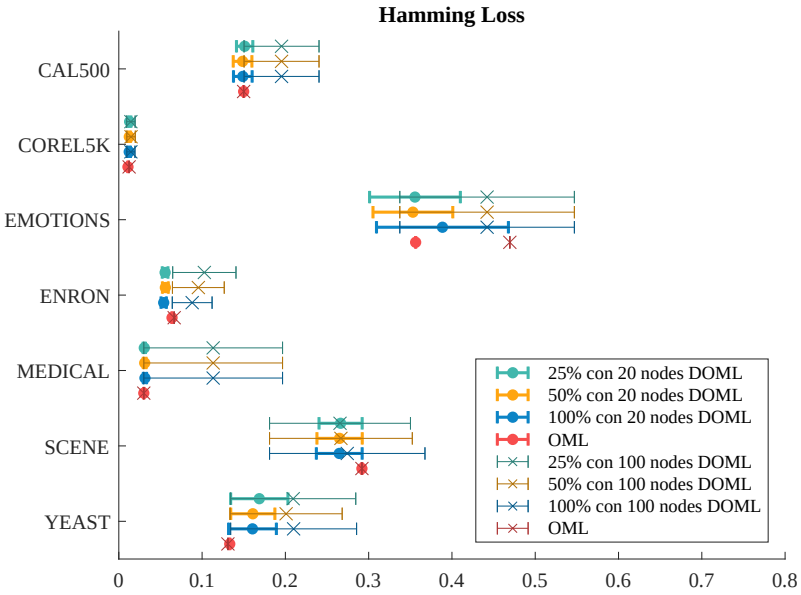

Figure 5. Hamming loss of DOML in a 100-node network with different connectivity and OML with same amount of instances as each node.

7.2.2. F1 Score Evaluation

Tables 7 and 8 present the DOML algorithm's F1 scores for a 20-node network and 100-node network, respectively, with varying degrees of network connectivity. Figures 6 and 7 compare the performance of the macro f1 score and the micro f1 score under different network conditions, respectively. In the experiment of F1 scores, we can observe a large variance in the DOML results. It is especially more obvious in larger networks. This is due to the differentiation caused by a large number of nodes. We use the middle value of the maximum and minimum values as a reference to consider the F1 score performance of all nodes in the DOML algorithm.

Table 7. F1 Score of DOML in a 20-node network with different connectivity and OML with same amount of instances as each node.

	Macro F1 Score				Micro F1 Score			
	DOML			OML	DOML			OML
	25%	50%	100%	-	25%	50%	100%	-
CAL500	**0.0773**	0.0725	0.0720	0.0569	0.3331	**0.3423**	0.3358	0.3014
Corel5k	0.0054	**0.0058**	0.0040	0.0009	**0.0829**	0.0761	0.0681	0.0430
Emotions	0.2061	**0.2207**	0.1864	0.0838	**0.2638**	0.2622	0.2273	0.1692
Enron	0.0577	0.0587	**0.0657**	0.0124	0.4141	0.4356	**0.4570**	0.2233
Medical	**0.0281**	0.0259	0.0254	0.0000	**0.0974**	0.0903	0.0892	0.0000
scene	0.1133	0.1090	**0.1442**	0.0478	0.2341	0.2377	**0.2509**	0.1603
yeast	0.2624	0.2668	0.2820	**0.2845**	0.6889	0.7086	0.7120	**0.7841**

The numbers in bold highlight the best performing of the three algorithms.

Table 8. F1 Score of DOML in a 100-node network with different connectivity and OML with the same amount of instances as each node.

	Macro F1 Score				Micro F1 Score			
	DOML			OML	DOML			OML
	25%	50%	100%	-	25%	50%	100%	-
CAL500	**0.0672**	**0.0672**	**0.0672**	0.0369	**0.3173**	**0.3173**	**0.3173**	0.2746
Corel5k	0.0016	**0.0020**	0.0017	0.0009	0.0798	**0.0862**	0.0791	0.0382
Emotions	**0.1708**	**0.1708**	**0.1708**	0.1497	**0.3167**	**0.3167**	**0.3167**	0.2914
Enron	0.0232	0.0214	0.0211	**0.0362**	0.2312	0.2217	0.2055	**0.4457**
Medical	**0.0122**	**0.0122**	**0.0122**	0.0000	**0.0670**	**0.0670**	**0.0670**	0.0000
scene	0.0577	0.0582	**0.0585**	0.0478	0.1275	0.1295	0.1299	**0.1603**
yeast	0.2357	0.2483	0.2154	**0.2862**	0.5381	0.5718	0.4908	**0.7894**

The numbers in bold highlight the best performing of the three algorithms.

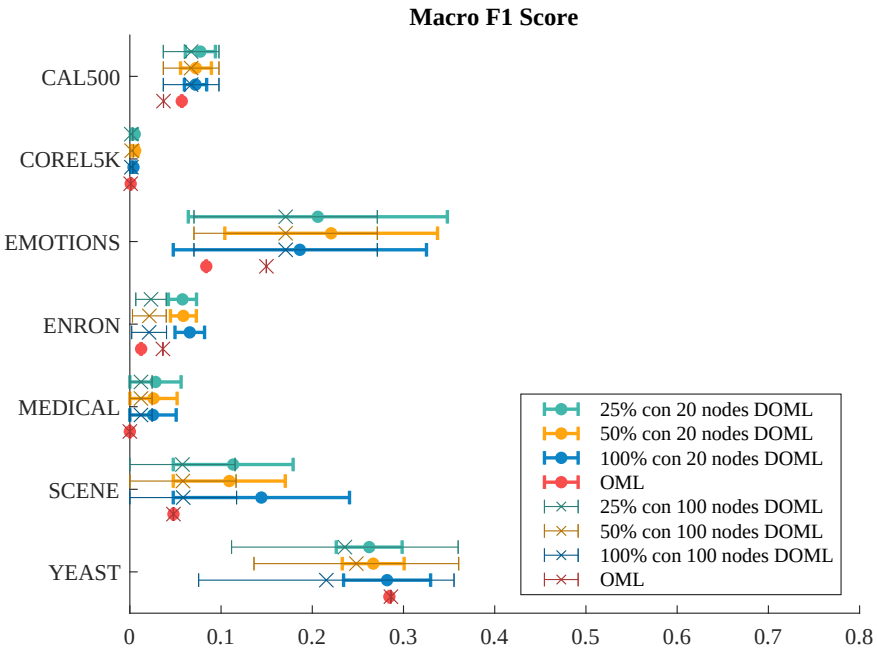

Figure 6. Macro F1 Score of DOML in a 100-node network with different connectivity and OML with same amount of instances as each node.

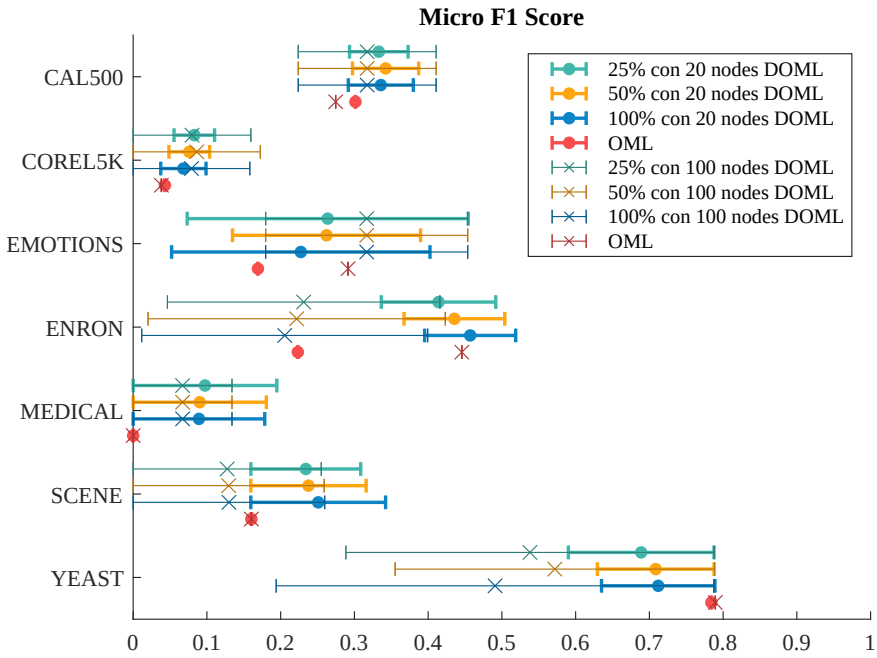

Figure 7. Micro F1 Score of DOML in a 100-node network with different connectivity and OML with same amount of instances as each node.

As shown in Tables 7 and 8, the DOML algorithm has a better F1 score than the OML algorithm in most cases when the same amount of data is called by a single node. This is because the DOML algorithm obtains the common features in the whole network by self-coordination. In addition, as shown in Figures 6 and 7, the same dataset with the same network size but different connectivity has almost the same distribution of F1 scores. This indicates that networks with different degrees of connectivity do not significantly affect the results of the DOML algorithm.

Hence, we can conclude that our proposed DOML algorithm can effectively derive a model that better fits the global characteristics through autonomous node coordination.

8. Conclusions

Distributed multi-label algorithms without centralized servers have great potential for application in real-world situations. However, there is still a research gap in implementing these problems. This article proposes a novel distributed multi-label classification learning method without a central server based on the distributed least-squares method. This approach succeeds in building a self-coordinated network structure, thus removing the centralized servers. This allows the potential for privacy data leakage or backdating to be eradicated on the transmission side. Experimental results show that our proposed DOML algorithm is not inferior to existing centralized algorithms in terms of Hamming loss and F1 score, and outperforms traditional centralized algorithms in some datasets. Meanwhile, we find that the DOML algorithm is a constant trade-off between the pursuit of local optimality and convergence with neighbouring nodes' data, which effectively suppresses the overfitting situation. At the current stage, we have implemented distributed online multi-label learning methods under a static network. However, in real situations, the network is often unstable. Future work will include the improvement and experimentation of our proposed algorithm for dynamic networks.

Author Contributions: Conceptualization, F.H.; Methodology, F.H.; Software, F.H.; Formal analysis, F.H.; Investigation, F.H. and N.Y.; Resources, D.Y.; Data curation, F.H.; Writing—original draft, F.H.; Writing—review & editing, F.H. and D.Y.; Visualization, F.H.; Supervision, H.C., W.B. and D.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: In this research, seven public datasets have been used for performance evaluations; they are: CAL500 [51], Corel5k [52], Emotions [53], Enron, Medical [54], scene [20] and yeast [55].

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhang, X.; Graepel, T.; Herbrich, R. Bayesian online learning for multi-label and multi-variate performance measures. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, JMLR Workshop and Conference Proceedings, Sardinia, Italy, 13–15 May 2010; pp. 956–963.
2. Zanella, A.; Bui, N.; Castellani, A.; Vangelista, L.; Zorzi, M. Internet of things for smart cities. *IEEE Internet Things J.* **2014**, *1*, 22–32. [CrossRef]
3. Spyromitros-Xioufis, E.; Spiliopoulou, M.; Tsoumakas, G.; Vlahavas, I. Dealing with concept drift and class imbalance in multi-label stream classification. In Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, Barcelona, Spain, 16–22 July 2011.
4. Büyükçakir, A.; Bonab, H.; Can, F. A novel online stacked ensemble for multi-label stream classification. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, Torino, Italy, 22–26 October 2018; pp. 1063–1072.
5. Li, P.; Wang, H.; Böhm, C.; Shao, J. Online semi-supervised multi-label classification with label compression and local smooth regression. In Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence, Yokohama, Japan, 7–15 January 2021; pp. 1359–1365.
6. Granato, G.; Martino, A.; Baiocchi, A.; Rizzi, A. Graph-Based Multi-Label Classification for WiFi Network Traffic Analysis. *Appl. Sci.* **2022**, *12*, 11303. [CrossRef]
7. Appenzeller, A.; Leitner, M.; Philipp, P.; Krempel, E.; Beyerer, J. Privacy and Utility of Private Synthetic Data for Medical Data Analyses. *Appl. Sci.* **2022**, *12*, 12320. [CrossRef]
8. Zheng, X.; Li, P.; Chu, Z.; Hu, X. A survey on multi-label data stream classification. *IEEE Access* **2019**, *8*, 1249–1275. [CrossRef]
9. Joachims, T. Text categorization with support vector machines: Learning with many relevant features. In *Machine Learning: ECML-98*; Nédellec, C., Rouveirol, C., Eds.; Springer: Berlin, Germany, 1998; pp. 137–142.
10. Gonçalves, T.; Quaresma, P. A preliminary approach to the multilabel classification problem of Portuguese juridical documents. In *Progress in Artificial Intelligence*; Pires, F.M., Abreu, S., Eds.; Springer: Berlin, Germany, 2003; pp. 435–444.
11. Luo, X.; Zincir-Heywood, A.N. Evaluation of two systems on multi-class multi-label document classification. In *Foundations of Intelligent Systems*; ISMIS 2005; Lecture Notes in Computer Science; Hacid, M.S., Murray, N.V., Raś, Z.W., Tsumoto, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3488, pp. 161–169. [CrossRef]
12. Yu, K.; Yu, S.; Tresp, V. Multi-label informed latent semantic indexing. In Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Salvador, Brazil, 15–19 August 2005; pp. 258–265.
13. Tsoumakas, G.; Katakis, I.; Vlahavas, I. Mining multi-label data. In *Data Mining and Knowledge Discovery handbook*; Springer: Berlin, Germany, 2009; pp. 667–685.
14. Elisseeff, A.; Weston, J. A kernel method for multi-labelled classification. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2001; Volume 14.
15. Zhang, M.L.; Zhou, Z.H. A k-nearest neighbour based algorithm for multi-label classification. In Proceedings of the IEEE International Conference on Granular Computing, Beijing, China, 25–27 July 2005; Volume 2, pp. 718–721.
16. Karalic, A.; Pirnat, V. Significance level based multiple tree classification. *Informatica* **1991**, *15*, 12.
17. Boutell, M.; Shen, X.; Luo, J.; Brown, C. Multi-label Semantic Scene Classfication. 2003. Available online: https://urresearch.rochester.edu/fileDownloadForInstitutionalItem.action?itemId=186&itemFileId=269 (accessed on 12 February 2023).
18. Zhu, B.; Poon, C.K. Efficient approximation algorithms for multi-label map labeling. In *Algorithms and Computation*; ISAAC 1999; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 1999; Volume 1741, pp. 143–152. [CrossRef]
19. Tsoumakas, G.; Katakis, I. Multi-label classification: An overview. *Int. J. Data Warehous. Min. (IJDWM)* **2007**, *3*, 1–13. [CrossRef]
20. Boutell, M.R.; Luo, J.; Shen, X.; Brown, C.M. Learning multi-label scene classification. *Pattern Recognit.* **2004**, *37*, 1757–1771. [CrossRef]
21. Zhang, M.L.; Zhou, Z.H. A review on multi-label learning algorithms. *IEEE Trans. Knowl. Data Eng.* **2013**, *26*, 1819–1837. [CrossRef]
22. Read, J.; Pfahringer, B.; Holmes, G.; Frank, E. Classifier chains for multi-label classification. *Mach. Learn.* **2011**, *85*, 333–359. [CrossRef]
23. Brinker, K.; Hüllermeier, E. Case-Based Multilabel Ranking. In Proceedings of the IJCAI, Hyderabad, India, 6–12 January 2007; pp. 702–707.
24. Lin, X.; Chen, X.w. Mr. KNN: Soft relevance for multi-label classification. In Proceedings of the 19th ACM International Conference on Information and Knowledge Management, Toronto, ON, Canada, 26–30 October 2010; pp. 349–358.

25. Veloso, A.; Meira, W.; Gonçalves, M.; Zaki, M. Multi-label lazy associative classification. In *Knowledge Discovery in Databases: PKDD 2007*; Lecture Notes in Computer Science; Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2007; Volume 4702, pp. 605–612. [CrossRef]
26. Zhang, M.L.; Zhou, Z.H. ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognit.* **2007**, *40*, 2038–2048. [CrossRef]
27. Huang, J.; Li, G.; Wang, S.; Huang, Q. Categorizing social multimedia by neighbourhood decision using local pairwise label correlation. In Proceedings of the 2014 IEEE International Conference on Data Mining Workshop, Shenzhen, China, 14 December 2014; pp. 913–920.
28. Liu, H.; Wu, X.; Zhang, S. Neighbour selection for multilabel classification. *Neurocomputing* **2016**, *182*, 187–196. [CrossRef]
29. Clare, A.; King, R.D. Knowledge discovery in multi-label phenotype data. In *Principles of Data Mining and Knowledge Discovery*; PKDD 2001; Lecture Notes in Computer Science; De Raedt, L., Siebes, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2001; Volume 2168, pp. 42–53.
30. Blockeel, H.; De Raedt, L.; Ramon, J. Top-down induction of clustering trees. *arXiv* **2000**, arXiv:cs/0011032.
31. Petrovskiy, M. Paired comparisons method for solving multi-label learning problem. In Proceedings of the 2006 Sixth International Conference on Hybrid Intelligent Systems (HIS'06), Auckland, New Zealand, 13–15 December 2006; p. 42.
32. Li, J.; Xu, J. A fast multi-label classification algorithm based on double label support vector machine. In Proceedings of the IEEE International Conference on Computational Intelligence and Security, Beijing, China, 11–14 December 2009; Volume 2, pp. 30–35.
33. Crammer, K.; Singer, Y. A family of additive online algorithms for category ranking. *J. Mach. Learn. Res.* **2003**, *3*, 1025–1058.
34. Zhang, M.L.; Zhou, Z.H. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE Trans. Knowl. Data Eng.* **2006**, *18*, 1338–1351. [CrossRef]
35. Gibaja, E.; Ventura, S. Multi-label learning: A review of the state of the art and ongoing research. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2014**, *4*, 411–444. [CrossRef]
36. Moyano, J.M.; Gibaja, E.L.; Cios, K.J.; Ventura, S. Review of ensembles of multi-label classifiers: Models, experimental study and prospects. *Inf. Fusion* **2018**, *44*, 33–45. [CrossRef]
37. Venkatesan, R.; Er, M.J.; Wu, S.; Pratama, M. A novel online real-time classifier for multi-label data streams. In Proceedings of the IEEE 2016 International Joint Conference on Neural Networks (IJCNN), Vancouver, BC, Canada, 24–29 July 2016; pp. 1833–1840.
38. Zhang, Y.; Liu, W.; Ren, X.; Ren, Y. Dual weighted extreme learning machine for imbalanced data stream classification. *J. Intell. Fuzzy Syst.* **2017**, *33*, 1143–1154. [CrossRef]
39. Arabmakki, E.; Kantardzic, M.; Sethi, T.S. A partial labeling framework for multi-class imbalanced streaming data. In Proceedings of the IEEE 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, USA, 14–19 May 2017; pp. 1018–1025.
40. ALattas, A.M. Adaptive model over a multi-label streaming data. In Proceedings of the 2018 IEEE 21st Saudi Computer Society National Computer Conference (NCC), Riyadh, Saudi Arabia, 25–26 April 2018; pp. 1–5.
41. Read, J.; Bifet, A.; Holmes, G.; Pfahringer, B. Scalable and efficient multi-label classification for evolving data streams. *Mach. Learn.* **2012**, *88*, 243–272. [CrossRef]
42. Osojnik, A.; Panov, P.; Džeroski, S. Multi-label classification via multi-target regression on data streams. *Mach. Learn.* **2017**, *106*, 745–770. [CrossRef]
43. Gong, X.; Yuan, D.; Bao, W. Online metric learning for multi-label classification. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 4012–4019.
44. Shi, L.; Zhao, L.; Song, W.Z.; Kamath, G.; Wu, Y.; Liu, X. Distributed least-squares iterative methods in networks: A survey. *arXiv* **2017**, arXiv:1706.07098.
45. Frommer, A.; Renaut, R.A. A unified approach to parallel space decomposition methods. *J. Comput. Appl. Math.* **1999**, *110*, 205–223. [CrossRef]
46. Renaut, R.A. A parallel multisplitting solution of the least squares problem. *Numer. Linear Algebra Appl.* **1998**, *5*, 11–31. [CrossRef]
47. Yang, L.T.; Brent, R.P. Parallel MCGLS and ICGLS methods for least squares problems on distributed memory architectures. *J. Supercomput.* **2004**, *29*, 145–156. [CrossRef]
48. Sayed, A.H.; Lopes, C.G. Distributed recursive least-squares strategies over adaptive networks. In Proceedings of the 2006 IEEE Fortieth Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA, 29 October–1 November 2006; pp. 233–237.
49. Chung, F.R.K. *Spectral Graph Theory*; CBMS Regional Conference Series, Conference Board of the Mathematical Sciences; American Mathematical Society: Providence, RI, USA, 1997.
50. Wang, X.; Zhou, J.; Mou, S.; Corless, M.J. A distributed algorithm for least squares solutions. *IEEE Trans. Autom. Control.* **2019**, *64*, 4217–4222. [CrossRef]
51. Turnbull, D.; Barrington, L.; Torres, D.; Lanckriet, G. Semantic annotation and retrieval of music and sound effects. *IEEE Trans. Audio, Speech, Lang. Process.* **2008**, *16*, 467–476. [CrossRef]
52. Duygulu, P.; Barnard, K.; de Freitas, J.F.; Forsyth, D.A. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Computer Vision—ECCV 2002*; Lecture Notes in Computer Science; Heyden, A., Sparr, G., Nielsen, M., Johansen, P., Eds.; Springer: Berlin/Heidelberg, Germany, 2002; Volume 2353, pp. 97–112.

53. Trohidis, K.; Tsoumakas, G.; Kalliris, G.; Vlahavas, I.P. Multi-label classification of music into emotions. In Proceedings of the ISMIR, Philadelphia, PA, USA, 14–18 September 2008; Volume 8, pp. 325–330.
54. Pestian, J.; Brew, C.; Matykiewicz, P.; Hovermale, D.J.; Johnson, N.; Cohen, K.B.; Duch, W. A shared task involving multi-label classification of clinical free text. In Proceedings of the Biological, Translational, and Clinical Language Processing, Prague, Czech Republic, 29 June 2007; pp. 97–104.
55. Dietterich, T.G.; Becker, S.; Ghahramani, Z. *Advances in Neural Information Processing Systems 14: Proceedings of the 2001 Conference*; MIT Press: Cambridge, MA, USA, 2002; Volume 2.