Noname manuscript No.
(will be inserted by the editor)

Multilabel Graph-based Classification for Missing Labels

Yasunobu Sumikawa · Tatsurou Miyazaki

Received: date / Accepted: date

Abstract Assigning several labels to digital data is becoming easier as this can be achieved in a collaborative manner with Internet users. However, this process is still a challenge, especially in cases where several labels are assigned to each datum, as some suitable labels may be missed. The missing labels lead to inaccuracies in classification. In this study, we propose a novel graph-based multi-label classifier that exhibits stability for obtaining high-accuracy results; this is achieved even where there are missing labels in training data. The core process of our algorithm is to smoothen the label values of the training data from their top-k similar data by propagating their values and averaging them to generate values for the missing labels in the training data. In experimental evaluations, we used multi-labeled document and image datasets to evaluate classifiers, and then measured micro-averaged F-scores for eight classifiers. Even though we incrementally removed correct labels from the two datasets, the proposed algorithm tended to maintain the F-scores, whereas other classifiers decreased the scores. In addition, we evaluated the algorithm using Wikipedia, which comprises a real dataset that includes missing labels, in order to determine how well the algorithm predicted the correct labels and how useful it was for manual annotations, as initial decisions. We have confirmed that LPAC is useful for not only automatic annotation, but also the facilitation of decision making in the initial manual category assignment.

Keywords Multi-label classification · label propagation · digital document classification · digital image classification

Y. Sumikawa
University Education Center, Tokyo Metropolitan University
E-mail: ysumikawa@acm.org

T. Miyazaki
Depart. of Information Sciences, Tokyo University of Science, Japan
E-mail: tatsumiya05@gmail.com

1 Introduction

Thanks to the growing size of the Web and digital archiving technology, we can now access numerous digital documents, images, and other types of data. This situation is good for enhancing our experiences of using the Web; for example, it is easy to study the history of any country, to find big pictures about relationships between people, and so on. On the other hand, it is becoming increasingly demanding to organize digital data to access them quickly. Defining categories and dividing digital data into these categories play key roles in digital data organization. For example, the categorization of digital documents is useful for constructing thematic timelines or event lists.

As the amount of data increases, the categorization schemes dynamically change due to the revision of the hierarchical structure and the definition of new categories. When these categorization schemes change, it is necessary to re-assign categories to existing data, which requires a huge cost in manual work. For example, in Wikipedia, there are numerous articles and categories. These data are organized by Wikipedia editors who are able not only to edit them, but also to add new articles and categories using the discussions on them. Thus, when some Wikipedia categories are updated, some articles can be assigned new categories. As another example, if it is necessary to create a new dataset from combining several existing datasets such as the New York Times and Japan Times, it is sometimes required to define new category schemes and assign them to their data.

The increasing number of Internet users makes annotations easier because many people can work on such tasks in parallel following the same policies used on sites, e.g., editing Wikipedia. Nevertheless, this task is still quite challenging. For example, some Wikipedia articles report several

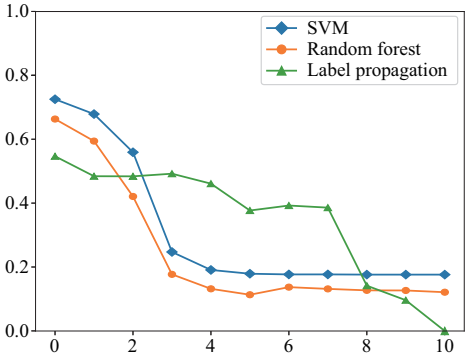


Fig. 1 Effect of missing labels in multi-label classification. x- and y-axes represent number of missing labels and micro-averaged F-scores of SVM, RF, and LP, respectively.

types of natural disasters[1]. In this category, there are two sub-categories: `Avalanches` and `Earthquakes`. Many articles in the `Avalanches` category have the `Natural disasters` category, whereas few articles in the `Earthquakes` have it. Thus, even though these articles report the same topic, the attached categories are not the same.

The missing labels is a very serious issue with respect to obtaining good classifier accuracy because almost all of the classifiers assume that labeled data prepared by people are correct. Fig. 1 shows how missing labels worsen the accuracies of support vector machine (SVM), random forest (RF), and label propagation (LP) on a multi-label dataset called the SIAM 2007 Text Mining Competition dataset[2], whose documents are assigned 3.4 labels on average. We can see that if only one label is missing, all their micro-averaged F-scores worsen by about 5%. If more than two labels are missing, the scores can decrease by about 10%.

Contributions: In this study, we propose a novel graph-based algorithm for multi-label classification (MLC) that is named LP using amendable clamping (LPAC). The core contribution of this study is to make label values for missing labels in labeled data. For this purpose, LPAC effectively utilizes *cluster assumption* [31], which implies that similar nodes tend to have common labels, by propagating label values of top similar data, called *local-propagation*, and updating scores of the labeled data by averaging from the top similar data, called *dynamic clamping*. These two processes add the values of the missing labels from their similar data. There are past works proposing LP-based algorithms that enhance the cluster assumptions. However, their objectives are different from that of these studies; for example, using the assumption for constructing graphs [28] and training kernel [5].

[1] `https://en.wikipedia.org/wiki/Category:Natural_disasters_by_country`

[2] `https://catalog.data.gov/dataset/siam-2007-text-mining-competition-dataset`

We evaluated LPAC by using three types of datasets: SIAM 2007 TextMining Competition dataset, digital images, and Wikipedia articles. We used the former two datasets to evaluate the accuracy in the term of stability for LPAC outputs with reducing the number of the correct labels. We incrementally removed correct labels of the two datasets and trained all classifiers on them. We confirmed that LPAC is more stable in terms of micro-averaged F-score than baselines. Next, in the third experimental evaluation, we trained the classifiers on Wikipedia articles and confirmed that LPAC is the best algorithm in terms of micro-averaged F-score and is useful for manual annotations as initial decisions.

The contributions of this study can be summarized as follows:

1. We proposed a novel LP-based algorithm using local-propagation and dynamic clamping for smoothing the effects of missing labels.
2. We performed quantitative evaluations on document classification by reducing correct labels, and confirmed that our algorithm is the best compared with widely used classifiers and other LP-based algorithms that dynamically propagate label values.
3. We also evaluated our algorithm on image classification and confirmed that our algorithm is the best as well in document classification.
4. We applied our algorithm to Wikipedia articles with several missing suitable Wikipedia categories. Our algorithm achieved the best results and will be useful for manual annotations as initial decisions.

Problem Statement. Let $\mathcal{L}$ be a finite and non-empty set of labels $\{l_1, l_2, ..., l_m\}$. Let $\mathcal{X}$ and $\mathcal{Y}$ be the input and output spaces, respectively. Given a dataset $\mathcal{D} = (\mathcal{D}^l, \mathcal{D}^u, \mathcal{D}^t)$ that includes labeled data $\mathcal{D}^l = \{(\boldsymbol{x_i}, \boldsymbol{y_i})\}_{i=1}^{P} \subset \mathcal{X} \times \mathcal{Y}$, $\boldsymbol{y_i} \in \{0,1\}^m$ and unlabeled data $\mathcal{D}^u = \{\boldsymbol{x_i}\}_{i=P+1}^{Q} \subset \mathcal{X}$, MLC predicts labels $\hat{y} = \{y_k \mid 1 \leq k \leq m\}$ for test data $\mathcal{D}^t = \{\boldsymbol{x_i}\}_{i=Q+1}^{R} \subset \mathcal{X}$. In this study, we define two labeled datasets $\mathcal{D}^l_{comp}$ and $\mathcal{D}^l_{missed}$ that are completely and partially assigned suitable labels, respectively. In other words, for all $i \in [1, P]$, $\boldsymbol{x_i} = \boldsymbol{x'_i}$ and $|\boldsymbol{y_i}| \geq |\boldsymbol{y'_i}|$ where $\boldsymbol{y_i} \in \mathcal{D}^l_{comp}$ and $\boldsymbol{y'_i} \in \mathcal{D}^l_{missed}$. In this work, it is assumed that classifiers are trained on $\mathcal{D}^l_{missed}$ and $\mathcal{D}^u$.

Outline. The remainder of this paper is organized as follows: Sec. 2 provides summaries of several related works. We describe the fundamental algorithm of LP in Sec. 3 and extend this algorithm to LPAC in Sec. 4. Sec. 5 performs evaluations on document and image classification, and qualitative analysis for Wikipedia articles. Sec. 6 concludes the remarks.

2 Related Works

In this paper, we first summarize classification studies in Sec. 2.1, to give a big picture of the classification problem. As LPAC is a semi-supervised learning (SSL) graph-based approach, we next summarize the SSL classification (Sec. 2.2) and previous graph-based studies (Sec. 2.3). Finally, in Sec. 2.4, we summarize event classification studies as we evaluate LPAC using events in Sec. 5.2.3.

2.1 Single- and Multi-label Classification

Single-label classification studies constitute the most important studies in classification research because many algorithms proposed as MLC and SSL are based on single-label classification. A special case of single-label classification is binary classification, that assigns one of two categories to each datum. RF and SVM are popular algorithms that perform binary classification. If there are two or more categories, and we apply classifiers to evaluate categories as one vs. the rest, then the classification problem can be extended from binary classification to multi-class classification that assigns a label to data from the categories. These studies play key roles in natural language processing (NLP), information retrieval (IR), machine learning (ML), and other research fields. Therefore, several researchers have published survey papers [1,3,21,23].

An extension of single-label classification to assign one or more categories to a datum is called MLC. Based on the manner of approach, MLC algorithms can be divided into two: transformation and algorithm adaptation [26]. The transformation approach transforms data into a form that can be applied to traditional single-label classifiers. This approach independently trains several classifiers for each label and predicts labels by combining [6] or chaining [22] the classifiers. As a type of the transformation approach, the label powerset is a popular MLC approach. This approach transforms the label representation to apply multi-class classifiers after creating all combinations of the labels. On the other hand, the algorithm adaptation extends an existing single-label classifier to treat multi-label data. MLkNN is one of the most well-known algorithms using this approach [30]. These two types of approaches can be used as ensemble style approaches, such as random k-labelsets (RAKEL) [13] and ensembles of classifier chains [22], that combine results from several classifiers based on the transformation and algorithm adaptation approaches. Usually, a voting scheme where every category is predicted by taking the probability of votes from individual classifiers [14] is employed to combine the results. Frameworks of MLC studies are also presented as survey papers [3,21,29], as well as those of single-label classifications.

2.2 Semi-supervised Learning Style Classification

It is important to prepare high-quality labeled datasets to train both single- and multi-label classifiers. However, the preparation is usually costly; in many real applications, the available labeled data are scarce and assigning suitable labels to unlabeled data is time-consuming. If we can obtain numerous unlabeled data, SSL-style classification is useful to reduce the cost of preparing labeled data, because this approach incrementally adds labeled data from unlabeled ones by applying classifiers trained on labeled data and retraining the classifiers on the new labeled data, including the results of the classifiers [4].

One of the most popular implementations of the SSL classification is to use single- and multi-label classifiers with the expectation-maximization (EM) algorithm to train the classifiers [7,8,18]. The detail is given by survey papers [20, 33]. Like in the SSL case, Kong *et. al.* proposed a transductive algorithm that solves an optimization problem to estimate label concept compositions, by utilizing information from both labeled and unlabeled data [10]. In a related study, Chapelle *et. al.* proposed kernel training for SSL-style classification by cluster assumption [5]. Zhou *et al.* proposed an algorithm that constructs a smooth function taking care of the intrinsic structure revealed by known labeled and unlabeled data [31].

2.3 Graph-based Classification

LP has been proposed as SSL classification in [32]. This algorithm spreads labels from a small-sized labeled data for unlabeled data. This is done by using a graph whose nodes and edges represent labeled and unlabeled data, and their similarities, respectively. This algorithm is based on two fundamental assumptions. First, the initial values of the labeled data should be kept during the spread from the unlabeled data. Second, similar data should tend to have the same label.

The original algorithm is designed for single-label classification, especially for the multi-class classification. However, this algorithm has been extended to MLC to take care of several issues associated with conventional MLC; for example, taking label correlation [9,27] as detailed in the survey paper [35].

Kang *et. al.* exploited the correlation between labels on the assumption that labels in multi-label classification are often correlated respectively [9]. Wang and Tsotsos proposed the DLP [27] that performs label fusion and diffusion to take label correlations while propagating label values, in order to extend the traditional single-labeled LP to multi-labeled LP. The label fusion assumes that two data points with highly correlated label vectors tend to have high similarities in the input data space. To preserve the intrinsic structure of the

input data during the label fusion process, DLP uses KNN. The label diffusion emphasizes the intrinsic structure between all the input data, using the KNN matrix. Wang and Zhang [28] proposed the LNP that efficiently constructs graphs to perform graph-based SSL classifications. LNP applies KNN to incorporate similarity into a probability matrix during the graph construction phase. The DLP and LNP are related to this study. The reason is that DLP performs dynamic label propagation by label fusion that utilizes label values from top-k, whereas, LNP propagates label values from the labeled data and their neighborhoods. On the other hand, our proposed algorithm performs dynamic clamping by incorporating label values of top-k similar data calculated by KNN.

The aforementioned studies utilize cluster assumptions to extend domains of the initial algorithm of the LP while respecting the first assumption (the values of the initial labeled data should not be affected by the spread from the unlabeled data). This assumption should be kept if there are no missing labels in the training data; in other words, it is fine if the classifiers are trained on $\mathcal{D}^l_{comp}$. However, if there are missing labels in the training data, the assumption is no longer valid, because classifiers regard data without labels as not representative of the labels. Thus, in this study, we tackle this problem to improve the soundness of the assumption; in other words, our proposed algorithm is designed to be trained on $\mathcal{D}^l_{missed}$.

2.4 Event Classification

Event classification is becoming a popular research topic, especially over the last decade. Kosmerlj *et al.* proposed event categories that were originally defined by Wikipedia editors [11]. They then use TF-IDF trained from news articles to classify the news into the event categories. This study trains a classifier on the whole text of the news articles; however, several events can be mentioned with a few sentences, such as news articles containing references to related events, historical accounts or biographies. To classify the past event described with short texts, Sumikawa and Jatowt proposed a feature selection method [25]. This study assumed each event to have only one label defined in the Wikipedia Current portal; thus, this study is designed as a single-label classification.

The main differences between this study and previous event classifications lie in the availability or non-availability of SSL classifiers and whether or not, missing labels are assumed. The previous studies are designed as non-SSL classifiers and have no missing labels in their training data; thus, their objectives are different from those of this study.

Algorithm 1 Traditional LP

1: Construct a probabilistic transition matrix P.
2: Let $Y_0 = [Y^l_0 : \mathbf{0}]$
3: **for** $t = 1$ to $T - 1$ **do**
4: $\quad Y_{t+1} = P \cdot Y_t$
5: $\quad Y^l_{t+1} = Y^l_0$
6: **end for**
7: **return** Y_t

3 Traditional LP

In this section, we describe the well-known LP algorithm [32]. We first present a formal definition of a graph $\mathcal{G} = (N, E)$, as LP invokes classification on a graph. For each data $\boldsymbol{x} \in \mathcal{X}$, a node $n \in N$ is created. Then, a positive value $w_{i,j}$ is assigned to an edge $(n_i, n_j) \in E$ as a weight between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ where $\boldsymbol{x_i}, \boldsymbol{x_j} \in \mathcal{X}$. The value of $w_{i,j}$ is determined as follows:

$$w_{i,j} = \exp(-\frac{\|v_i - v_j\|^2}{\alpha^2}) \tag{1}$$

where v_i is a vector of $\boldsymbol{x_i}$ and α is a hyper-parameter.

Because a graph can be represented as a matrix, LP is usually invoked by multiplying a probabilistic transition matrix P by a label matrix Y to propagate labels. The probabilistic transition matrix P represents the probability of transition from n_i to n_j, and is defined by normalizing the similarity metrics as follows:

$$P(i, j) = \frac{w_{i,j}}{\sum_{k=1}^{n} w_{i,k}} \tag{2}$$

The label matrix Y is defined as $[Y^l : Y^u] \in \mathbb{R}^{|D|\times|C|}$, where $|\ D\ |$ is the number of data, $|\ C\ |$ is the number of categories, Y^l is the label matrix for labeled data, and Y^u is the label matrix for unlabeled data. The value of $Y^l(i, c)$ is 1 if $\boldsymbol{x}_i$'s categories include c; otherwise, it is 0.

Algorithm 1 shows the entire algorithm of LP. First, the two initial matrices, P and Y, are constructed. Then, they are multiplied repeatedly until all elements are fixed, or the iterative process converges to a certain number. During the iteration, values of Y^l are reset by assigning their initial values (Y^l_0) before invoking the next iteration. This is known as *clamping*. This assigning process removes any effects of the propagating labels for all labeled data in order to propagate the same values for all the iterations.

4 LP using Amendable Clamping

We extend the traditional LP to create propagation of missing labels by adding the following two steps:

1. *Local-propagation*: Enhancing the propagation of label values of similar data.
2. *Dynamic clamping*: Updating Y^l from labels of similar data.

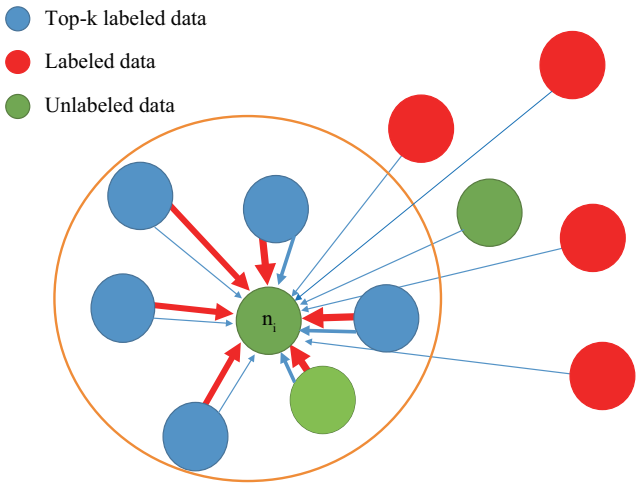


Fig. 2 Example of local-propagation. Red arrows represent local-propagation; LPAC additionally propagates the weights of the top-k data. Blue arrows represent propagations used in traditional LP.

In this algorithm, we assume that there are no labels wrongly assigned to training data. Thus, in the training data there are missing labels, while all the attached labels are correct.

4.1 Local-propagation

The objective of the first extension is to emphasize the cluster assumption in the label propagation phase. Fig. 2 presents an example to provide an intuitive understanding of the local-propagation process. Circles represent the nodes of the graph. Blue and red circles indicate that they are labeled data whereas green ones are unlabeled data. In this example, we focus on the green node n_i within five blue nodes and one green node in the orange circle and, put them close to each other to ease the presentation. Arrows represent flows of the propagating label weights. The blue arrows represent the flows for all nodes, whereas the red arrows represent flows for only top-k similar nodes. The traditional algorithm performs the propagation shown with the blue arrows only. LPAC propagates not only the values represented by the blue arrows, but also those represented by red arrows.

For this local-propagation, we add $(M \cdot P) \cdot Y_t$ to the matrix multiplication in the traditional algorithm where $M \in \mathbb{N}^{|N|\times|N|}$ is a matrix representing the KNN of the data; M_{ij} is 1 if the ith data is in the top-k similar ones of the jth data; otherwise, it is 0. As $M \cdot P$ allows the propagation of label values only for top-k similar data, we call this procedure *local-propagation*. In contrast, we call $P \cdot Y_t$, which is used in traditional LP, *global-propagation*. To adjust the balance between local- and global-propagations, we introduce a hyper-parameter β as follows:

$$Prop_t = \beta P \cdot Y_t + (1-\beta)(M \cdot P) \cdot Y_t \quad (3)$$

Algorithm 2 LP using amendable clamping

1: Construct a probabilistic transition matrix P and a KNN matrix M.
2: Let $Y_0 = [Y_0^l : \mathbf{0}]$
3: **for** $t = 1$ to $T-1$ **do**
4: $\quad Prop_t = \beta P \cdot Y_t + (1-\beta)(M \cdot P) \cdot Y_t$ # global- and local-propagation
5: $\quad Y_{t+1} = M \cdot Prop_t / k$
6: $\quad Y_{t+1}^l = M \cdot Y_{t+1}^l / k$ # Dynamic clamping
7: **end for**
8: **return** Y_t

We then take the cluster assumption; we calculate the averages from the top-k similar data and set them as the result of the propagation. The following equation invokes this process:

$$Y_{t+1} = \frac{M \cdot Prop_t}{k} \quad (4)$$

4.2 Dynamic Clamping

The second extension is the dynamic clamping. We define this clamping as updating values of Y^l from the top-k similar data in each iteration after propagating labels. Thus, LPAC may propagate different (dynamic) values from labeled data in each iteration, whereas the traditional LP algorithm propagates the same (static) label values from the labeled data. To perform the dynamic clamping, LPAC uses the M representing KNN to calculate the average values of the top-k similar data for each label. These values are set before invoking the next iteration as follows:

$$Y_{t+1}^l = \frac{M \cdot Y_{t+1}^l}{k} \quad (5)$$

4.3 Overview of the Algorithm

Finally, we show our entire algorithm in Algorithm 2. The first line defines a probabilistic transition matrix P and a KNN matrix M. After defining a label matrix at line 2, LPAC iteratively applies to the two extensions (from lines 4 to 6). Finally, LPAC returns the label matrix.

5 Experimental Evaluations

We performed two types of evaluations. The first one is a quantitative evaluation to measure the accuracies of classifiers on document and image datasets. The second one is a case study evaluation to observe the usefulness of LPAC in practical situations. All of these evaluations were performed on a macOS High Sierra computer with an Intel Core i5 (3.1 GHz) CPU and 8 GB of memory.

5.1 Experimental Setting

Algorithms. For all the evaluations, we compared the LPAC with the following seven baselines:

1. RF. RF is one of the most popular algorithms for MLC.
2. SVM. SVM is also one of the popular algorithms for MLC. We trained the SVM with a linear kernel.
3. SSL-RF. This classifier is an RF trained as SSL. This classifier applies the EM algorithm to train the RF classifier iteratively.
4. SSL-SVM. This classifier is an SVM-linear kernel trained as SSL. This classifier applies the EM algorithm to iteratively train the SVM classifier.
5. LP. This is the fundamental algorithm described in Sec. 3.
6. Dynamic LP (DLP). DLP dynamically propagates label values from top similar nodes during the iteration process. As this dynamic propagation is useful to address error propagation [24] that smoothens effects of wrong labels, we implemented DLP as a baseline.
7. LP through linear neighborhoods (LNP). This algorithm is an extension of LP for discovering the structure of an entire dataset through the linear neighborhoods of each data. As this algorithm updates label values from the top similar data, we implemented LNP as a baseline.

Because SSL may generate weak results compared with supervised learning [34], we trained the RF and SVM classifiers as supervised learning method to compare SSL with supervised learning. The EM algorithm iterates classifier training and label assignments to increase the size of the labeled data; thus, it is necessary to define a strategy for the label assignment procedure. We used the same strategy as that used in training the RF and SVM in the supervised learning manner.

Evaluation Measures. The algorithms were evaluated using the F-score, defined as follows:

$$\mathrm{MiP} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{6}$$

$$\mathrm{MiR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{7}$$

$$\mathrm{F-score} = \frac{2\mathrm{MiPMiR}}{\mathrm{MiP} + \mathrm{MiR}} \tag{8}$$

where TP is true positive, FP is false positive, FN is false negative, MiP (micro-average precision) is the number of TPs in each category divided by the total number of TPs and FPs in each category, and MiR (micro-average recall) is the number of TPs in each category divided by the total number of TPs and FNs in each category. The trade-off between the micro-average precision and micro-average recall is formalized by their harmonic mean, known as the micro-average F-score. Hereinafter, these terms are simply referred to as precision, recall, and F-score.

Table 1 Statistics of document dataset.

Num. of categories	22
Num. of labeled data	4,819
Num. of unlabeled data	4,819
Num. of test data	4,819
Ave. length per document	1191.3
Ave. num. of categories per document	3.41

5.1.1 Quantitative Evaluation

As for quantitative evaluation, we perform document and image classifications under the following research questions.

- **RQ1**. How well does each classifier correctly predict test data without missing labels on training data?
- **RQ2**. How stable does each classifier predict correct labels even though missing labels exist?

Parameters. We set T (the iteration number of LP), the dimension of the latent Dirichlet allocation (LDA) [2] that is used to create feature vectors in document classification, k, and β to 1,000, 1,000, 5, and 0.1 respectively. All these values were empirically selected based on analyzing the results on a small held-out development dataset. It is noteworthy that DLP uses two additional parameters (λ and α). We set them to the same values used in [27]. Finally, we set 0.2 as a threshold for label assignment after the iteration of LP-based algorithms because these algorithms only assign a score for each label to each data point.

Dataset for Document Classification. We used the SIAM 2007 Text Mining Competition dataset, which is a subset of the Aviation Safety Reporting System dataset. It provides various types of aviation safety events reported by pilots, controllers, mechanics, flight attendants, and dispatchers. This dataset defines 22 categories and assigns 3.41 labels for each document on average.

Next, to create missing labels for analyzing **RQ2**, we intentionally removed the attached labels of the documents. We first selected 9,638 labeled data from the SIAM 2007 Text Mining Competition dataset. Subsequently, we removed all labels from 4,819 documents. Therefore, we prepared 4,819 and 4,819 labeled and unlabeled data, respectively. Tab. 1 summarizes the statistics of the dataset that we used for the evaluation. Next, to create different sizes of missing labels, we selected several labeled documents from the 4,819 labeled data to remove their labels. We first extracted some data from this dataset to remove the labels; subsequently, we removed the labels attached to the data. Both the extraction ratio and removal ratio increased from 10% to 100%, in 10% increments, and were selected randomly. For all the cases, we inspected the F-scores, which were measured by the 10-fold cross-validation of classifiers that were trained on the dataset.

Table 2 Statistics of image dataset.

Num. of object categories	8
Num. of shape classes	3
Num. of labeled data	48
Ave. num. of categories per document	2

Dataset for Image Classification. We evaluated the six algorithms, which were RF, SVM, LP, DLP, LNP, and LPAC for image data using a toy dataset[3]. This dataset defines two categories: objects (helmet, kettle, joystick, keyboard, mouse, stapler, barrel, and mug) and shapes (triangle, square, and circle). Each image is classified into 2 categories: one from the object category and the other one from the shape category. This dataset contains only 48 data; therefore, it is a challenging dataset. We decided to omit SSL-RF and SSL-SVM in this evaluation as this dataset is insufficient for preparing unlabeled data. Tab. 2 shows the statistics of the image dataset.

We created different sizes of missing labels dataset by the same procedure as the one used in the document classification. The detail is described in Sec. 5.2.2.

5.1.2 A Case Study: Classification on Wikipedia Category

Finally, we evaluated how well LPAC assigned correct labels to the English version of Wikipedia articles, which were data comprising missing labels.

Research Questions. Wikipedia comprises numerous categories. The broad range of categories is suitable for encompassing a significant number of topics; however, as described in Sec. 1, several articles contained missing labels. Hence, we evaluated RF, SVM, and all LP-based algorithms based on the following research questions.

- **RQ3**. How accurately does each algorithm predict the correct categories?
- **RQ4**. How is each algorithm useful in automatic annotations or providing hints for manual annotations?

Parameters. We used the same values as in the document classification experiment (Sec. 5.1.1) for the parameters of this experiment.

Data Collection Procedure. To the best of our knowledge, no ground truth dataset exists for this case study; therefore, we developed a dataset from the topic of natural disaster. To collect these articles, we performed a Wikipedia-category-based article crawling. First, we collected all types of disasters that were stored in "*Natural_disasters_by_country*" category[4]. Subsequently, we obtained five categories of natural disasters: `Avalanches`, `Floods`, `Tornadoes`, `Earthquakes`, and `Landslides`. For each disaster d, we further collected all the Wikipedia pages stored in *d_by_country*[5] [6] [7] [8] [9]. Tab. 3 shows the statistics for the results obtained by crawling. From the Wikipedia-category-based article crawling, we obtained 1,372 Wikipedia pages. Tab. 4 shows the statistics of the training dataset.

Table 3 Statistics of Wikipedia dataset.

Num. of training articles	1,347
Num. of test articles	25
Num. of Wikipedia categories	6
Ave. length per document	7,878.4
Ave. num. of categories per document	352.5

Table 4 Statistics of Wikipedia category. Average lengths and numbers of Wikipedia articles for each Wikipedia category used in training dataset.

Wiki. Cat.	Ave. len.	Num. of Wiki. Art.
`Avalanches`	3,193.3	28
`Floods`	6,359.2	328
`Tornadoes`	6,733.8	57
`Earthquakes`	3,151.1	772
`Landslides`	5,967.7	158
`Natural disasters`	7,082.0	144

Test Data Creation. Tab. 5 shows the test data we used in this evaluation. The first column assigns abbreviated indexes for the test data to ease the presentation. The second column lists the names of the test data. We randomly selected five Wikipedia articles for each Wikipedia category related to the natural disasters. The third column shows the lengths of the test data. The last six columns represent whether each test data point was part of the Wikipedia categories. These six columns present results of our manual review for evaluating whether each test data belonged to suitable Wikipedia categories. If a suitable Wikipedia category is correctly assigned to the article, we then use the checkmark (✓); otherwise, we use a hyphen (-). For the manual review of the categories, we asked three volunteers to evaluate the Wikipedia categories that should be assigned to the test data. All these subjects have Ph. D. degrees and had worked in the field of machine learning. The three volunteers provided the same results; hence, we used the results as the test data, as shown in Tab. 5.

Evaluation Criteria. For **RQ3**, we evaluated all the classifiers by the micro-average precision, recall, and F-score as defined in Eqs. 6, 7, and 8, respectively.

[3] `https://github.com/AliAbbasi/Multilabel-Image-Classification-with-Softmax`

[4] `https://en.wikipedia.org/wiki/Category:Natural_disasters_by_country`

[5] `https://en.wikipedia.org/wiki/Category:Avalanches_by_country`

[6] `https://en.wikipedia.org/wiki/Category:Floods_by_country`

[7] `https://en.wikipedia.org/wiki/Category:Tornadoes_by_country`

[8] `https://en.wikipedia.org/wiki/Category:Earthquakes_by_country`

[9] `https://en.wikipedia.org/wiki/Category:Landslides_by_country`

Table 5 Assigned Wikipedia categories. Checkmark (✓) and hyphen (-) denote categories assigned to Wikipedia article and missed to be assigned, respectively. Abbreviated name of Wikipedia category is for `Natural disasters` (`N.D.`)

Wiki. Article	Title of the Wiki. Article	Len.	`Avalanches`	`Floods`	`Tornadoes`	`Earthquakes`	`Landslides`	`N.D.`
WA1	Tornado outbreak of May 15–17, 2013	5,272			✓			✓
WA2	1986 Vrancea earthquake	2,939				✓		-
WA3	2015 Afghanistan avalanches	885	✓					✓
WA4	2015 Guatemala landslide	2,236		-			✓	-
WA5	Pantai Remis landslide	1,379		✓			✓	-
WA6	1978 Singapore flood	438		✓				-
WA7	2016 Geier avalanche	1,334	✓					✓
WA8	2009 Schalfkogel avalanche	1,932	✓					✓
WA9	1985 Puerto Rico floods	9,144		✓			✓	✓
WA10	1983 Sea of Japan earthquake	3,877				✓		-
WA11	1981 Sirch earthquake	720				✓		-
WA12	1983 Erzurum earthquake	133				✓		-
WA13	Southern Ontario Tornado Outbreak of 2009	9,634			✓			✓
WA14	Tornado outbreak of June 5–6, 2010	4,450			✓			✓
WA15	1981 Irian Jaya earthquake	2,320				✓		-
WA16	2015 East Malaysian floods	7,294		✓				-
WA17	2014 Hiroshima landslides	5,334					✓	-
WA18	2015 Argentina floods	998		✓				-
WA19	November 1989 tornado outbreak	9,439			✓			✓
WA20	2015 South Indian floods	76,909		✓				-
WA21	Valfréjus avalanche	2,540	✓					✓
WA22	Tornado outbreak of May 26–31, 2013	16,127		-	✓			✓
WA23	2015 Colombian landslide	2,960		-			✓	-
WA24	1954 Blons avalanches	730	✓					✓
WA25	2015 Bahia landslide	122					✓	-

For **RQ4**, we evaluated the number of suitable categories that were assigned by each classifier to the test Wikipedia articles.

5.2 Result Analysis

5.2.1 Document Classification

Feature Vector Creation. To train the classifiers, feature vectors must be created at first. In this document classification, we compared three types of feature vectors, namely: bag-of-words (BoW), LDA[10], and Doc2Vec[11] [12]. These three feature vectors were selected as they are widely used in document classification. BoW indicates that feature vectors were created by counting the number of words in each document; hence, the dimensional size is equivalent to the size of the word set. LDA is an unsupervised and probability-based topic detection algorithm. For a specified number of topics, LDA inferences all the topics provided by the entire dataset; subsequently, it calculates the distribution of the relationship of each document to each topic. As the distribution can be used as a feature vector, we used them to train the classifiers. To train an LDA model, we used both labeled and unlabeled data. Doc2Vec is a neural-network-based algorithm extended from Word2Vec [15–17]. Word2Vec assigns vectors to words by embedding words into vector spaces by employing CBOW and Skip-gram models. Doc2Vec introduces a paragraph vector to the two models. Similar to LDA, we trained the Doc2Vec model on both the labeled and unlabeled data.

[10] `https://radimrehurek.com/gensim/models/ldamodel.html`

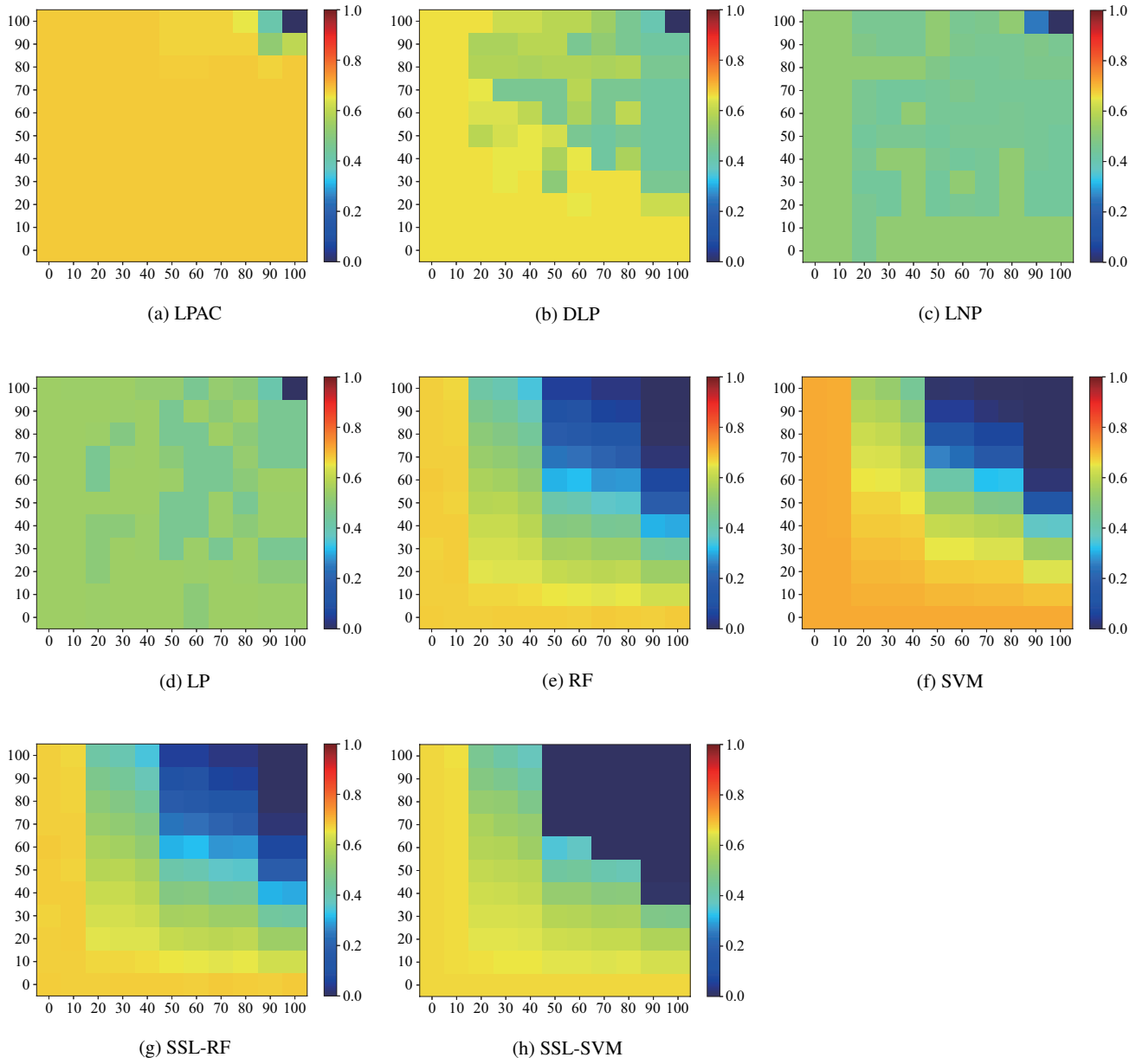
[11] `https://radimrehurek.com/gensim/models/doc2vec.html`

Table 6 Feature selection results. Precision, recall, and F-scores of LPAC equipped with BoW, LDA, and Doc2Vec.

	BoW	LDA	Doc2Vec
MiP	62.6%	**65.7**%	50.0%
MiR	60.7%	**68.8**%	36.1%
F-score	61.7%	**67.2**%	42.0%

Tab. 6 shows the results of LPAC trained on the entire dataset. As shown, the most suitable method to generate the feature vector was by using the LDA. As applying the LDA reduces the dimensions of the feature vectors, it is better than BoW. Additionally, Doc2Vec was useful for reducing dimensions; however, its result was worse than that of BoW. As Doc2Vec is a neural network-based approach, it requires a significant amount of training data. However, we used less than 10,000 training data. If more training data are used, then Doc2Vec may be as useful as the LDA. Therefore, we only used the LDA in the remainder of the document classification experiments.

Discussions of Accuracies. The results of the document classification are shown in Fig. 3. Figs. 3 (a) ∼ (h) show the accuracies of each classifier, which had been trained on datasets wherein 0% ∼ 100% of the documents did not contain any labels. Based on the accuracies of the LPAC that were trained on a dataset wherein 0% ∼ 90% of the documents contained missing labels, the scores were similar to



(a) LPAC (b) DLP (c) LNP

(d) LP (e) RF (f) SVM

(g) SSL-RF (h) SSL-SVM

Fig. 3 F-scores for document classification. Each sub-figure shows micro-averaged F-scores for each classifier. The x-axis represents ratio of missing labels. The y-axis represents ratio of documents containing missing labels. Each cell color represents the F-score. Red cell indicates a better accuracy than blue cell.

those of the SVM and RF, which were trained on a dataset wherein 10% of the documents contained missing labels. By contrast, all baselines tended to yield deteriorated accuracies once the ratio of missing-label documents exceeded 30%. In particular, once the missing-label ratio reached 50%, the accuracies of DLP, RF, SVM, SSL-RF, and SSL-SVM decreased rapidly from approximately 10% to 50%. Furthermore, LNP and LP tended to maintain their accuracies, but their scores were lower than those obtained by LPAC.

For a better comparison among all classifiers, Figs. 4 and 5 show their F-scores for cases wherein we extracted 50% and 70% of documents for removing labels, respectively. When the ratio of missing labels reached over 40%, the F-scores of five baselines (DLP, RF, SVM, SSL-RF, and SSL-SVM) began to decrease earlier compared with the case for $0 \sim 40\%$. Moreover, when the ratio was over 80%, the accuracies of RF, SVM, SSL-RF, and SSL-SVM decreased significantly. In the case wherein 70% of the documents were

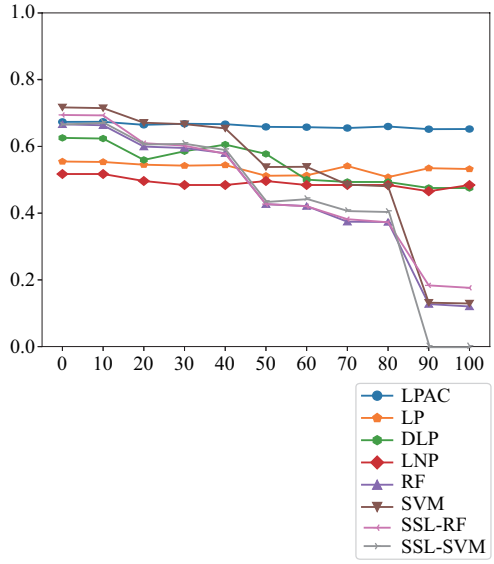


Fig. 4 F-scores when 50% of documents contained missing labels. The x-axis represents ratio of missing labels whereas the y-axis represents F-score.

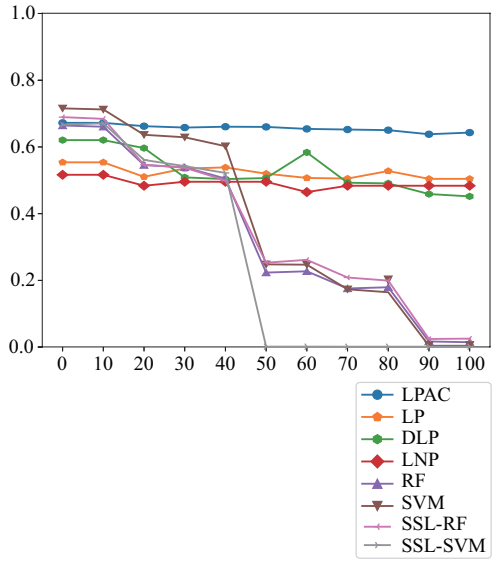


Fig. 5 F-scores when 70% of documents contained missing labels. The x-axis represents ratio of missing labels whereas the y-axis represents F-score.

extracted, the accuracies of almost all the baselines began to decrease when the ratio of missing labels reached 20%. Conversely, LPAC maintained a stable accuracy for all missing label ratios.

Based on the discussions above, we summarize the answers for the two research questions here.

An answer for RQ1. By observing the two figures (Figs. 4 and 5), the initial F-score of SVM was the best among all the classifiers as it exceeded 70%. LPAC performed worse than the SVM and SSL-RF; however, the *F-score for LPAC was also approximately 70%. This result indicates that LPAC can be used practically.*

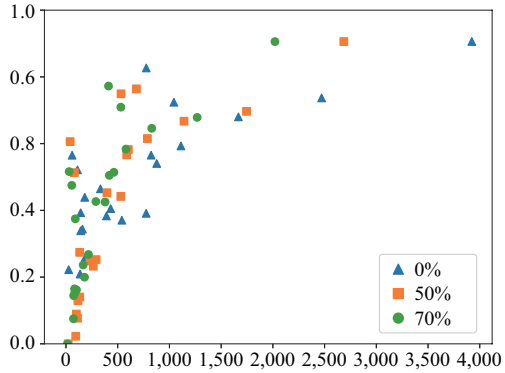


Fig. 6 Correlation between number of training labels and F-scores of LPAC. The x-axis represents number of training data. The y-axis represents value of F-scores. Blue, yellow, and green points represent F-scores when 0%, 50%, and 70% of documents contained missing labels, respectively.

An answer for RQ2. *LPAC maintained its F-scores even when correct labels were removed for 70% of the training data.* This indicates that LPAC is useful as a semi-supervised learning method. This property was observed for other LP-based algorithms; however, LPAC achieved better scores than other LP-based algorithms. By contrast, the scores of the SVM, RF, and their SSL-style classifiers decreased when the number of missing labels increased rapidly.

We subsequently evaluated LPAC to obtain a better understanding of the amount of data required to obtain high accuracies. Fig. 6 shows the correlation between the F-scores and the number of labeled documents for LPAC. We plotted the results of three cases, where 0%, 50%, and 70% of the documents contained missing labels. Fig. 6 shows that a positive correlation existed among them. In fact, their correlation coefficiencies were 0.76, 0.72, and 0.75, respectively. Hence, we concluded that the number of correct labels that were assigned to the training data was directly proportional to the achievable accuracy.

Finally, we compared the analysis times of the LP-based algorithms in terms of the time complexity and practical measured times.

To compare the time complexities, we separated all LP-algorithms into two phases: matrix construction and label propagation. First, LP, DLP, and LPAC constructed a probabilistic transition matrix (P) whose time complexity is $O(n^2)$. In addition, DLP and LPAC construct a KNN matrix whose time complexity is $O(n)$. Furthermore, DLP constructs a KNN matrix corresponding to P that can be created by averaging the top-k of each data point in $O(kn)$. By contrast, LNP solves standard quadratic programming problems to construct a weight matrix. It is well known that solving standard quadratic programming problems is NP-hard [19]; this

Table 7 Analysis times. Times for creating matrices and propagation until achieving convergence (s).

	LP	DLP	LNP	LPAC
Creating Matrices	10.4	20.2	1186.7	16.8
Propagation	8.9	2197.8	0.8	14.9
Total	*19.3*	*2218.0*	*1187.5*	*31.7*

indicates that the time complexity for the matrix construction of LNP is not polynomial time. We solved the problem using the CVXOPT[12] library, which uses a Newton method whose order is $O(n^3)$[13] t times. As this process is the most expensive, we regarded the time complexity for constructing a weight matrix in LNP as $O(tn^3)$. Hence, in the matrix construction phase, LPAC is the second fastest among the four algorithms.

Based on the propagation phase, all algorithms iterate the propagation T times or until convergence is achieved. LP performs only a matrix multiplication wherein the time complexity is $O(Tn)$. LPAC performs the Hadamard product and the averaging scores of top-k of all data points twice, in addition to the LP process. As the two additional processes are performed to obtain the top-k of each data point, the time complex of each additional process is $O(kn)$. Therefore, the time complexity of LPAC is calculated as $O(Tkn)$. DLP performs fusion and diffusion, wherein the time complexities are $O(n^2)$ and $O(kn^2 + kn)$, respectively; in addition, it accomplishes a matrix multiplication identical to that of LP. Hence, the complexity of DLP is $O(Tkn^2 + Tkn)$. LNP performs a matrix multiplication and a matrix addition, and its complexity is $O(Tn)$. Therefore, in the propagation phase, LPAC is the third fastest among the four algorithms.

Next, we present the measured practical analysis time of the LP-based algorithms in Tab. 7. We categorized the analysis times into two types, namely: for creating matrices and propagating labels. These times were measured for the original dataset and no labels were removed. In total, LP was the fastest and was approximately twice as fast as LPAC. Comparing LPAC with the other two algorithms, LPAC was at least 37-fold faster. Based on the propagation time of LNP, it was the fastest of the four algorithms; this was because its the number of iterations was the smallest. Next, we discuss how the iteration times increase with the number of iterations. Fig. 7 plotted the y-axis as a log scale to compare LPAC with all the baselines; this is because DLP requires a significant amount of time for the propagation, rendering it difficult to observe the differences between LP, LNP, and LPAC. We observed that all these algorithms increased linearly with the analysis time.

[12] http://cvxopt.org/userguide/coneprog.html#quadratic-programming

[13] http://www.seas.ucla.edu/~vandenbe/publications/mlbook.pdf

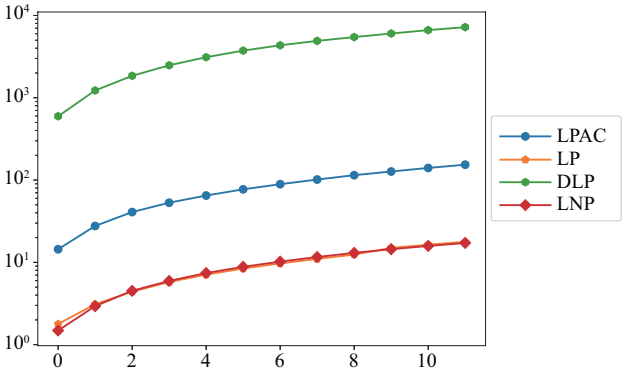


Fig. 7 Analysis times of iterations. The x-axis represents number of iterations. The y-axis represents log-scaled analysis time (s).

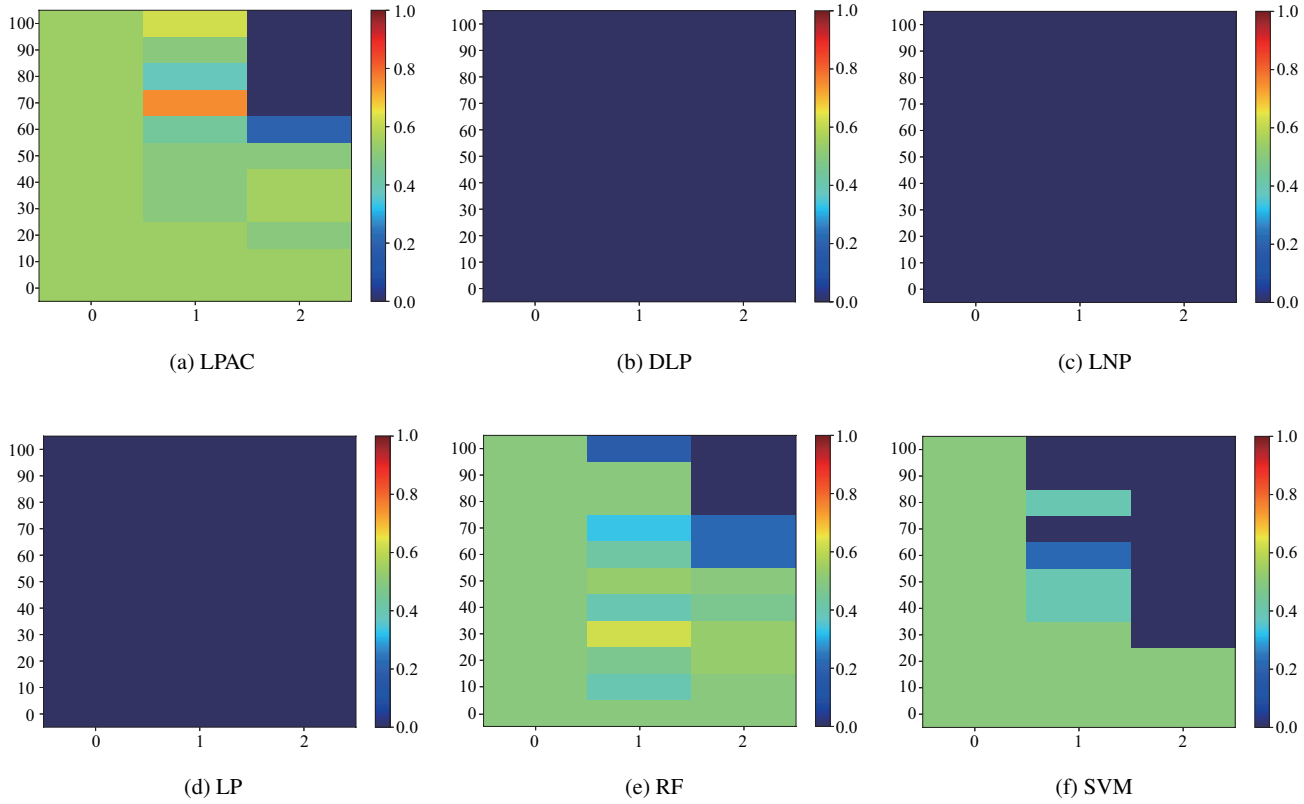
5.2.2 *Image Classification*

Feature Vector Creation. In this image classification, we used the pixel data to convert all the images into feature vectors.

Discussions of Accuracies. Figs. 8 (a) ~ (f) show the F-scores of each classifier that was trained on datasets wherein 0% ~ 100% of the images did not contain correct labels. Based on these results, we discovered that LPAC outperformed the comparative algorithms as all LP-based baselines completely failed to predict the correct labels for all the cases. RF and SVM successfully assigned correct labels for several cases; however, LPAC operated at better accuracies. Subsequently, we evaluated how LPAC maintained its F-scores even when the number of attached labels decreased as well as the document classification. Figs. 9 and 10 show the F-scores in cases where we extracted 40% and 60% of the images to remove the labels, respectively. When no labels were removed, three algorithms (LPAC, RF, and SVM) achieved more than 50% F-scores. In particular, LPAC achieved the best score (approximately 54.5%). As shown in Fig. 9, both LPAC and RF maintained their scores, whereas SVM demonstrated a linear decrease. Based on Fig. 10, all the three classifiers failed to maintain their scores. When all the labels for 60% of the images were removed, RF emerged as the best algorithm; however, the difference between LPAC and RF was small as their F-scores were 20.0% and 22.2%, respectively.

An answer for RQ1. Based on the two figures (Figs. 9 and 10), *the initial F-scores of LPAC were the best among those of all the classifiers discussed in the experimental result.*

An answer for RQ2. *LPAC maintained its F-scores even on when the correct labels for 40% of the training data were removed.* However, upon the removal of the correct labels for 60% of the training data, the scores became difficult to maintain.



(a) LPAC (b) DLP (c) LNP

(d) LP (e) RF (f) SVM

Fig. 8 F-score for image classification. Each sub-figure shows micro-averaged F-scores for each classifier. The x-axis represents ratio of missing labels. The y-axis represents ratio of images containing missing labels. Each cell color represents the F-score. Red cell indicates a better accuracy than blue cell.

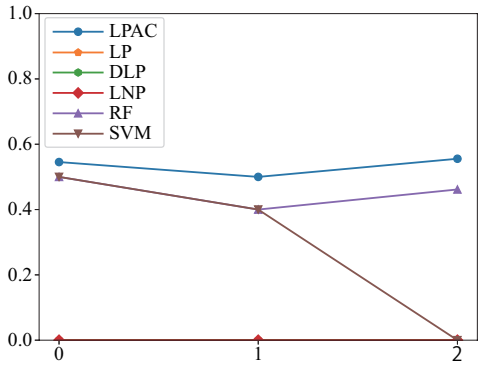


Fig. 9 F-score when 40% of images contained missing labels. The x-axis represents number of missing labels whereas the y-axis represents F-score.

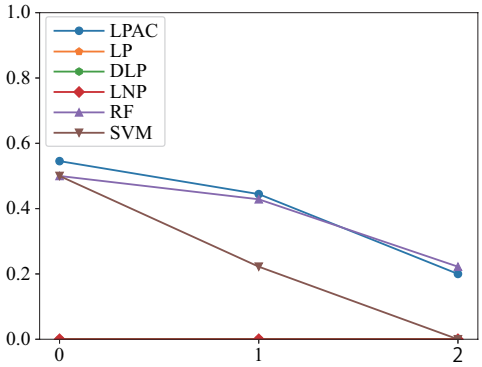


Fig. 10 F-score when 60% of images contained missing labels. The x-axis represents number of missing labels whereas the y-axis represents F-score.

5.2.3 A Case Study

Feature Vector Creation. We created feature vectors using the process used for document classification (Sec. 5.2.1) including the LDA usage.

Analysis for RQ3. Tab. 8 shows the precision, recall, and F-score obtained from this experimental evaluation. As

Table 8 Accuracies of the case study.

	LPAC	DLP	LNP	LP	RF	SVM
MiP	78.2%	42.9%	26.0%	26.0%	83.3%	**100.0**%
MiR	**79.6**%	38.9%	24.1%	24.1%	18.2%	52.7%
F-score	**78.9**%	40.8%	25.0%	25.0%	29.9%	69.0%

Table 9 Results of the case study. Checkmark (✓) and X mark (×) denote the correct and wrong predictions of the categories, respectively. The hyphen (-) indicates that the classifier misses assigning the correct category.

Alg.	Wiki. Article	Avalanches	Floods	Tornadoes	Earthquakes	Landslides	Natural disasters
LPAC	**WA1**			✓			-
	WA2		×		✓		-
	WA3	✓					✓
	WA4		✓		×	✓	✓
	WA5		✓		×	✓	✓
	WA6		✓		×		✓
	WA7	✓					✓
	WA8	✓					✓
	WA9		✓		×	✓	-
	WA10				✓	×	✓
	WA11				✓		-
	WA12				✓		-
	WA13			✓			✓
	WA14			✓			✓
	WA15				✓		-
	WA16		✓				-
	WA17		×		×	✓	✓
	WA18		✓		×		-
	WA19			✓			✓
	WA20		✓				-
	WA21	✓					✓
	WA22		-	✓			✓
	WA23		✓			✓	✓
	WA24	✓	×				✓
	WA25				×	✓	-
DLP	**WA1**	×	×	✓	×	×	✓
	WA2				✓		-
	WA3	-	×		×		-
	WA4		-		×	-	-
	WA5		✓		×	-	-
	WA6		✓		×		-
	WA7	-	×		×		-
	WA8	-	×		×		-
	WA9		-		×	-	-
	WA10				✓		-
	WA11				✓		-
	WA12				✓		-
	WA13		×	✓			✓
	WA14		×	✓			✓
	WA15				✓		-
	WA16		✓		×		-
	WA17				×	-	-
	WA18				×		-
	WA19		×	✓			✓
	WA20		✓		×		-
	WA21	-	×		×		-
	WA22		✓	✓			✓
	WA23		✓		×	-	-
	WA24	-	×		×		-
	WA25		×		×	-	-

all scores of LPAC were greater than 78%, this classifier yielded outstanding predictions; hence, we can conclude that *LPAC is useful for automatic annotations*. In particular, the recall was approximately 80%; this score was the most important of the three in this case study. Furthermore, LPAC was the best among all the classifiers in terms of the F-score; therefore, we can conclude that LPAC was the best classifier in this case study. In terms of precision, SVM was the best as it achieves 100%. However, in terms of the recall, its score was approximately 53%; therefore, its F-score was less than 70%. This tendency was observed in RF as well, wherein only the precision was high but the other scores were low. Additionally, we observed that the three LP-based baselines achieved low scores for all observations.

Analysis for RQ4. We analyzed the categories predicted by each classifier. Tabs. 9, 10, and 11 show the categories predicted by all the classifiers. In the tables, the checkmark (✓) and X mark (×) denote the correct and wrong predictions of the classifier, respectively. A hyphen (-) signifies that the classifier did not assign a suitable category.

First, we verified the number of correctly assigned categories. We observed that LPAC correctly predicted almost all the test data. In particular, for two categories (`Avalanches` and `Tornadoes`), LPAC yielded perfect predictions.

Table 10 Results of the case study. Checkmark (✓) and X mark (×) denote the correct and wrong predictions of the categories, respectively. The hyphen (-) indicates that the classifier misses assigning the correct category.

Alg.	Wiki. Article	Avalanches	Floods	Tornadoes	Earthquakes	Landslides	Natural disasters
LNP	**WA1**		×	-	×		-
	WA2		×		✓		-
	WA3	-	×		×		-
	WA4		✓		×	-	-
	WA5		✓		×	-	-
	WA6		✓		×		-
	WA7	-	×		×		-
	WA8	-	×		×		-
	WA9		✓		×	-	-
	WA10		×		✓		-
	WA11		×		✓		-
	WA12		×		✓		-
	WA13		×	-	×		-
	WA14		×	-	×		-
	WA15		×		✓		-
	WA16		✓		×		-
	WA17		×		×	-	-
	WA18		✓		×		-
	WA19		×	-	×		-
	WA20		✓		×		-
	WA21	-	×		×		-
	WA22		✓	-	×		-
	WA23		✓		×	-	-
	WA24	-	×		×		-
	WA25		×		×	-	-
LP	**WA1**		×	-	×		-
	WA2		×		✓		-
	WA3	-	×		×		-
	WA4		✓		×	-	-
	WA5		✓		×	-	-
	WA6		✓		×		-
	WA7	-	×		×		-
	WA8	-	×		×		-
	WA9		✓		×	-	-
	WA10		×		✓		-
	WA11		×		✓		-
	WA12		×		✓		-
	WA13		×	-	×		-
	WA14		×	-	×		-
	WA15		×		✓		-
	WA16		✓		×		-
	WA17		×		×	-	-
	WA18		✓		×		-
	WA19		×	-	×		-
	WA20		✓		×		-
	WA21	-	×		×		-
	WA22		✓	-	×		-
	WA23		✓		×	-	-
	WA24	-	×		×		-
	WA25		×		×	-	-

In addition, for the `Natural disasters` category, LPAC was the best among all the classifiers in terms of the number of labels assigned. The results for `Landslides` indicated only a single mispredicted datum (`WA10`). For these categories, LPAC demonstrated suitable accuracies. However, for the `Floods` and `Earthquakes` categories, the number of mispredictions were higher compared to those of other categories. This tendency was also observed in other LP-based algorithms; however, the other three LP-based baselines demonstrated more mispredictions for these two categories. This was because the sizes of the Wikipedia articles that were based on these two categories were larger than those based on the other categories, as shown in Tab. 4. Unexpectedly, SVM and RF demonstrated better results than all the LP-based classifiers in these two categories. RF demonstrated errors in the prediction for two data (`WA7` and `WA25`) as `Earthquakes`, but no mispredictions were observed for the category of `Floods`.

Next, we analyzed the missing labels. In the four categories (`Avalanches`, `Tornadoes`, `Earthquakes`, and `Landslides`), LPAC did not demonstrate any missing predictions. In the `Floods` category, the classifier missed the prediction for only one datum (`WA22`). It is noteworthy noting that `WA22` did not belong to a category in the original

Table 11 Results of the case study. Checkmark (✓) and X mark (×) denote the correct and wrong predictions of the categories, respectively. The hyphen (-) indicates that the classifier misses assigning the correct category.

Alg.	Wiki. Article	Avalanches	Floods	Tornadoes	Earthquakes	Landslides	Natural disasters
RF	**WA1**			✓			-
	WA2				✓		-
	WA3	-					-
	WA4		✓			-	-
	WA5		-			-	-
	WA6		-				-
	WA7	-			×		-
	WA8	-					-
	WA9		✓			-	-
	WA10				✓		-
	WA11				✓		-
	WA12				✓		-
	WA13			-			-
	WA14			-			-
	WA15				✓		-
	WA16		✓				-
	WA17					-	-
	WA18		-				-
	WA19			✓			-
	WA20		✓				-
	WA21	-					-
	WA22		-	-			-
	WA23		✓			-	-
	WA24	-					-
	WA25				×	-	-
SVM	**WA1**			✓			-
	WA2				✓		-
	WA3	✓					-
	WA4		-			✓	-
	WA5		-			✓	-
	WA6		✓				-
	WA7	✓					✓
	WA8	✓					✓
	WA9		✓			✓	-
	WA10				✓		-
	WA11				✓		-
	WA12				✓		-
	WA13			✓			-
	WA14			✓			-
	WA15				✓		-
	WA16		✓				-
	WA17					-	-
	WA18		✓				-
	WA19			✓			-
	WA20		✓				-
	WA21	✓					✓
	WA22		-	✓			-
	WA23		-			✓	-
	WA24	✓					✓
	WA25					✓	-

Table 12 Number of Wikipedia articles under category of `Natural disasters` (`N.D.`). Third column indicates ratios of Wikipedia articles in training data having `Natural disasters` categories.

Wiki. Cat.	Num. of Articles having `N.D.`	Raio
`Avalanches`	16	57.1%
`Floods`	59	18.0%
`Tornadoes`	22	38.6%
`Earthquakes`	11	1.4%
`Landslides`	33	20.9%

Wikipedia article (see Tab. 5); in other words, this classifier, at least for this category, can assign categories to Wikipedia articles as effectively as a human. Furthermore, no missed predictions were observed in the results of LNP and LP for this category; however, the two classifiers predicted these two categories for all the data; this implies that they were overfitting for these two categories. In the `Natural disaster` category, LPAC could not assign a category to 10 test data. To determine the reason, we investigated the ratio of labeled data in the category. Tab. 12 shows the ratios of the number of Wikipedia articles that were categorized under `Natural disasters`. We observed that the percentage of articles on the `Floods` and `Earthquakes` categories were under 20%; in particular, only 1.4% of the Wikipedia arti-

cles under the Earthquakes category were classified under Natural disasters. This low score may have caused LPAC to wrongly assign Earthquakes category to several test data. We believe that it is better to increase these ratios to achieve a more effective automatic classification.

Finally, we analyzed the wrongly assigned labels by LPAC. As discussed previously, LPAC assigned wrong labels to 10 Wikipedia articles in the Floods and Earthquakes categories. We asked three volunteers who have Ph. D. degrees and had worked on studies associated with machine learning the following question: *How useful is LPAC for supporting the initial decisions of assigning labels? As the list may include the wrong category and missing labels, please remove unnecessary categories or add new suitable categories if necessary.* We showed these volunteers all the results of the 10 Wikipedia articles (WA2, WA4, WA5, WA6, WA9, WA10, WA17, WA18, WA24, and WA25) to which LPAC had assigned the wrong labels. This was performed without informing them that those labels actually represented the wrong categories. We observed that all the volunteers removed the wrongly assigned categories by LPAC and categorized those articles as Natural disasters. This result indicates that the result of the wrongly assigned category can be corrected by human judgment; therefore, LPAC is useful for facilitating decision making with respect to manually assigning specific Wikipedia categories to Wikipedia articles.

Answer for RQ3. LPAC was useful for automatic annotations because all of its scores for precision, recall, and F-score exceeded 78%.

Answer for RQ4. LPAC is useful for facilitating decision making with respect to manually assigning Wikipedia categories to Wikipedia articles. As the accuracy of LPAC was high, the required human correction became affordable; therefore, the human annotators successfully removed the wrong categories and added the correct ones that were missed by LPAC.

5.2.4 Limitations and discussions of this case study.

In this study, we focused on data from five natural disasters collected from the Natural disasters category; however, more types of disasters exist, such as tsunamis. *We suggest improving problems associated with inaccessible categories to reduce problems related to missing labels.* After increasing the efficiency of the categorizing system, we can perform error analyses in detail; for example, how well does LPAC assign correct labels to articles in cases of causalities (e.g., between earthquakes and tsunamis) or correlations (e.g., heavy rains and landslides) between the categories.

In addition, Wikipedia comprises various articles that elucidate numerous categories, such as crimes, people, countries, and groups. As many events can be correlated to each other, other articles must be evaluated to improve the automatic identification of the relationships between different entity types. *If we can apply classifiers to obtain these relationships, we can then further improve the capabilities of Wikipedia, e.g., the development of thematic timelines of events that are listed by each person or group and the measurement of the importance of people based on their action when applying PageRank.*

5.3 Summary of Discussions

Finally, we summarize the key results, findings, and suggestions from our evaluations.

- The document classification showed that LPAC was the best algorithm among the eight algorithms used in this study if half or more training data contained missing labels.
- The document and image classifications showed that LPAC was the best in terms of stability.
- Although LPAC predicted correct labels using missing labels datasets, the number of correct labels that were assigned to the training data was directly proportional to the accuracy achieved by LPAC.
- The analysis time of LPAC increased linearly with the number of iterations.
- In the case study, LPAC was the best algorithm in terms of the micro-average recall and F-score for classifying articles under the category of Natural disasters, as defined in Wikipedia.
- LPAC was useful in facilitating manual annotations for obtaining the initial decisions of the process.
- The evaluation of other types of articles is an important future endeavor to automatically improve the identification of relationships between different entity types.

6 Conclusions & Future Studies

In this study, we proposed a novel graph-based multi-label classification (LPAC) to apply to a moderately challenging multi-labeling task. LPAC enforces propagations of missing labels by two extensions: propagating labels according to top-k similar data and updating labeled data. As demonstrated in our experiments, these two extensions rendered the F-score of the LPAC stable, as compared to previously proposed algorithms. In addition, a case study showed that LPAC achieved the best scores among all the classifiers used in this study, indicating its usefulness for manual annotations as initial decisions.

In future studies, we will identify (a) *the effective utilization of label correlation*. Once our algorithm is expanded to achieve label correlation, we can implement label recommendations. This suggestion should be helpful, in particular for cases wherein numerous labels exist, such as the

Wikipedia category system. In addition, we will (b) *establish an algorithm that can be trained on datasets that include both wrong and missing labels*. Although this study assumed that no data were attached to any wrong labels, real datasets might contain both kinds of labels simultaneously. Accordingly, we will investigate the method to reduce noise.

Acknowledgements This work was supported in part by MEXT Grant-in-Aid (#19K20631).

References

1. Barforoush, A., Shirazi, H., Emami, H.: A new classification framework to evaluate the entity profiling on the web: Past, present and future. ACM Comput. Surv. **50**(3), 39:1–39:39 (2017)
2. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
3. Branco, P., Torgo, L., Ribeiro, R.P.: A survey of predictive modeling on imbalanced domains. ACM Comput. Surv. **49**(2), 31:1–31:50 (2016)
4. Cardoso-Cachopo, A., Oliveira, A.L.: Semi-supervised single-label text categorization using centroid-based classifiers. SAC'07, pp. 844–851. ACM, New York, NY, USA (2007)
5. Chapelle, O., Weston, J., Schölkopf, B.: Cluster kernels for semi-supervised learning. NIPS'02, pp. 601–608. MIT Press, Cambridge, MA, USA (2002)
6. Cheng, W., Hüllermeier, E.: Combining instance-based learning and logistic regression for multilabel classification. Machine Learning **76**(2), 211–225 (2009)
7. Cong, G., Lee, W.S., Wu, H., Liu, B.: Semi-supervised text classification using partitioned em. Database Systems for Advanced Applications, pp. 482–493. Springer Berlin Heidelberg, Berlin, Heidelberg (2004)
8. Ghani, R.: Combining labeled and unlabeled data for multiclass text categorization. ICML'02, pp. 187–194. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2002)
9. Kang, F., Jin, R., Sukthankar, R.: Correlated label propagation with application to multi-label learning. CVPR'06, pp. 1719–1726. New York, NY, USA (2006)
10. Kong, X., Ng, M.K., Zhou, Z.: Transductive multilabel learning via label set propagation. IEEE Transactions on Knowledge and Data Engineering **25**(3), 704–719 (2013)
11. Košmerlj, A., Belyaeva, E., Leban, G., Grobelnik, M., Fortuna, B.: Towards a complete event type taxonomy. WWW'15 Companion, pp. 899–902. ACM, New York, NY, USA (2015)
12. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. ICML'14, pp. II–1188–II–1196. JMLR.org (2014)
13. Lo, H., Lin, S., Wang, H.: Generalized k-labelsets ensemble for multi-label and cost-sensitive classification. IEEE Transactions on Knowledge and Data Engineering **26**(7), 1679–1691 (2014)
14. Menc'ia, E.L., Park, S., Fürnkranz, J.: Efficient voting prediction for pairwise multilabel classification. Neurocomputing **73**(7-9), 1164–1176 (2010)
15. Mikolov, T., Kai, C., Suchanek Greg, C., Dean, J.: Linguistic regularities in continuous space word representations. NAACL-HLT'13, pp. 746–751 (2013)
16. Mikolov, T., Sutskever, I., Chen, K., S. Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. NIPS'13, pp. 3111–3119 (2013)
17. Mikolov, T., Yih, W.t., Zweig, G.: Efficient estimation of word representations in vector space. ICLR Workshop (2013)
18. Nigam, K., McCallum, A.K., Thrun, S., Mitchell, T.: Text classification from labeled and unlabeled documents using em. Mach. Learn. **39**(2-3), 103–134 (2000)
19. Pardalos, P.M., Vavasis, S.A.: Quadratic programming with one negative eigenvalue is np-hard. Journal of Global Optimization **1**, 15–22 (1991)
20. Pise, N.N., Kulkarni, P.: A survey of semi-supervised learning methods. In: 2008 International Conference on Computational Intelligence and Security, *CISIS'08*, vol. 2, pp. 30–34 (2008)
21. Qi, X., Davison, B.D.: Web page classification: Features and algorithms. ACM Comput. Surv. **41**(2), 12:1–12:31 (2009)
22. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. Machine Learning **85**(3), 333–359 (2011)
23. Sebastiani, F.: Machine learning in automated text categorization. ACM Comput. Surv. **34**(1), 1–47 (2002)
24. Seyedi, S.A., Lotfi, A., Moradi, P., Qader, N.N.: Dynamic graph-based label propagation for density peaks clustering. Expert Systems with Applications **115**, 314 – 328 (2019)
25. Sumikawa, Y., Jatowt, A.: Classifying short descriptions of past events. ECIR'18, pp. 729–736 (2018)
26. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining multi-label data pp. 667–685 (2010)
27. Wang, B., Tsotsos, J.: Dynamic label propagation for semi-supervised multi-class multi-label classification. Pattern Recognition **52**, 75 – 84 (2016)
28. Wang, F., Zhang, C.: Label propagation through linear neighborhoods. ICML'06, pp. 985–992. ACM, New York, NY, USA (2006)
29. Zhang, M., Zhou, Z.: A review on multi-label learning algorithms. IEEE Transactions on Knowledge and Data Engineering **26**(8), 1819–1837 (2014)
30. Zhang, M.L., Zhou, Z.H.: Ml-knn: A lazy learning approach to multi-label learning. Pattern Recogn. **40**(7), 2038–2048 (2007)
31. Zhou, D., Bousquet, O., Navin Lal, T., Weston, J., Scholkopf, B.: Learning with local and global consistency. NIPS'04, pp. 321–328. MIT Press (2004)
32. Zhu, X.: Semi-supervised learning with graphs. Ph.D. thesis, Pittsburgh, PA, USA (2005)
33. Zhu, X.: Semi-supervised learning literature survey. Comput. Sci., University of Wisconsin-Madison **2** (2008)
34. Zhu, X., Goldberg, A.B.: Introduction to semi-supervised learning. Intell. Mach. Learn (2009)
35. Zoidi, O., Fotiadou, E., Nikolaidis, N., Pitas, I.: Graph-based label propagation in digital media: A review. ACM Comput. Surv. **47**(3), 48:1–48:35 (2015)