GB$_{RAIN}$: Combating Textual Label Noise by Granular-ball based Robust Training

Zeli Wang
Key Laboratory of Cyberspace Big Data Intelligent Security, Ministry of Education, Chongqing University of Posts and Telecommunications
Chongqing, China
zlwang@cqupt.edu.cn

Tuo Zhang
Key Laboratory of Cyberspace Big Data Intelligent Security, Ministry of Education, Chongqing University of Posts and Telecommunications
Chongqing, China
s210231260@stu.cqupt.edu.cn

Shuyin Xia*
Key Laboratory of Cyberspace Big Data Intelligent Security, Ministry of Education, Chongqing University of Posts and Telecommunications
Chongqing, China
xiasy@cqupt.edu.cn

Longlong Lin
College of Computer and Information Science, Southwest University
Chongqing, China
longlonglin@swu.edu.cn

Guoyin Wang
Key Laboratory of Cyberspace Big Data Intelligent Security, Ministry of Education, Chongqing University of Posts and Telecommunications
Chongqing, China
wanggy@cqupt.edu.cn

ABSTRACT

Most natural language processing tasks rely on massive labeled data to train an outstanding neural network model. However, the label noise (i.e., wrong label) is inevitably introduced when annotating large-scale text datasets, which significantly degrades the performance of neural network models. To overcome this dilemma, we propose a novel Granular-Ball based t$_{RAIN}$ing framework, named GB_{RAIN}, to realize robust coarse-grained representation learning, thus combating label noises in diverse text tasks. Specifically, considering that most samples in the dataset are precisely labeled, GB_{RAIN} first proposes a dynamic granular-ball clustering algorithm to blend seamlessly into the traditional neural network model. A striking feature of the clustering algorithm is that it can adaptively group the embedding vectors of similar data into the same set (hereafter referred to as a granular-ball). The embedding vectors and labels of all samples from the same set will be coarse-grainedly represented by the center vector and the label of the granular-ball, respectively. Consequently, noise labels can be rectified through the labels of most of the labeled data. Moreover, we introduce a new gradient backpropagation mechanism compatible with our framework, which can help optimize coarse-grained embedding vectors with iterative training. Empirical results on text classification and name entity recognition tasks demonstrate that our proposal GB_{RAIN} is indeed effective in contrast to the state-of-the-art baselines.

*Corresponding author

ICMR '24, June 10–14, 2024, Phuket, Thailand

ACM ISBN 979-8-4007-0619-6/24/06
https://doi.org/10.1145/3652583.3658084

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**.

KEYWORDS

Natural processing language, Label noise, Clustering, Representation learning

ACM Reference Format:
Zeli Wang, Tuo Zhang, Shuyin Xia, Longlong Lin, and Guoyin Wang. 2024. GB$_{RAIN}$: Combating Textual Label Noise by Granular-ball based Robust Training. In *Proceedings of the 2024 International Conference on Multimedia Retrieval (ICMR '24), June 10–14, 2024, Phuket, Thailand.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3652583.3658084

1 INTRODUCTION

With the continuous growth of computing power, the emergence of large-scale datasets has led to significant success for Deep Neural Networks (DNNs) in various fields. It is well known that the performance of DNNs heavily relies on enormous high-quality annotated datasets. However, The labeling process will unavoidably introduce errors, due to human oversights in manual annotation or inaccuracies in the automatic annotation. Label noise seriously drags down DNNs' performance since the models learn incorrect associations and patterns. Research on label noise has received widespread attention from both academia and industry, particularly in the computer vision (CV) field [4, 11, 12, 19, 24, 29, 39, 44].

However, the solutions in the CV cannot be applied directly in the natural language processing (NLP) tasks, due to the discontinuity and ambiguity of texts. Few studies strive to enhance the resilience of DNN models to text label noises, mainly through robust optimization [6, 25] and noise filtering [20, 26]. The former aims at enhancing the model robustness by introducing adversarial loss [25] or optimizing objectives [6] in the training process. The latter focuses on identifying and eliminating label noise. This method first leverages the model prediction probabilities [12], sample uncertainties [9], or performance metrics [14] to assess whether labels are affected by noise, then removes or corrects annotations with

potential noises [20]. However, these methods still face limitations. For example, they typically handle low-dimensional samples, but difficult to deal with high-dimensional vectors that frequently appear in DNN applications [8, 13, 40]; they may have poor scalability when filtering large-scale label noise [41].

To mitigate these limitations, we aim to propose a novel framework that can provide robustness for NLP tasks. However, we face two significant challenges. ***How can we improve the accuracy of text representation learning under label noise?*** The precision of representation learning determines the effectiveness of NLP tasks. Specifically, natural language is complex with diverse contextual and semantic information. It is ambiguous with vocabulary and grammatical structures that can have multiple interpretations, making it difficult to learn the precise embedding representation. Moreover, identifying whether the labels are correct for NLP is tough. In computer vision, accuracy can be assessed by comparing the generated and real images. In contrast, the meanings and labels of the texts are more subjective, and establishing exact measurement standards of label quality is impossible. Hence, performing representation learning after filtering all label noise is not very realistic. In summary, ensuring the accuracy of representation learning underlying noise is significant and challenging for NLP tasks. ***How can we guarantee the scalability of DNN models, to efficiently process enormous datasets with much label noise?*** Deep learning relies on large annotated datasets to train an excellent neural network. As introduced above, an unanticipated amount of label noise may be introduced due to manual errors or tool limitations in the labeling procedure. The model must be scalable to handle large datasets and potentially substantial amounts of label noise.

In this paper, we propose GB_{RAIN}, a coarse-grained representation learning framework, for training a robust neural model on datasets with label noise. For the above two challenges, GB_{RAIN} realizes a multigranularity representation learning method based on Granular-Ball Computing (GBC) [33]. GBC can self-adaptively group close low-dimensional objects with the preservation of the data distribution characteristics. It then performs kinds of computing on the clusters to guarantee robustness [5, 34, 36, 37]. Inspired by this, GB_{RAIN} designs a granular-ball clustering algorithm for high-dimension text embedding. The Euclidean distance is used to measure the proximity between vectors to help cluster similar texts. However, each embedding has hundreds of dimension feature vectors in NLP. Even if the vectors in certain dimensions between two embeddings are far away, it is easy to be evened by the vectors of all dimensions. To accommodate clustering in high-dimensional vectors, GB_{RAIN} enhances the traditional GB clustering method by automatically allocating different weights to various dimensions when calculating distances. The farther the distance, the larger the weight of the dimension. Therefore, the distances between text embeddings are clearer. Then the clustering algorithm can adaptively gather similar texts based on such weighted Euclidean distances, producing granular-balls. Since all samples in a ball have close semantics, their label should be consistent. GB_{RAIN} regards the most prevalent label in a ball as the label of all samples in the ball. Then all samples in a ball will be substituted with a sample, whose embedding and label are the center vector and the label of the ball respectively. So the label noise in the training set can be corrected by the most precisely annotated data in the ball, and the coarse representation (the center vector of the ball) also reflects typical features of the text. When trained in such an environment, the model will be more robust. Hence, the first challenge is mitigated. For the second challenge, because the center vector of the ball will present all samples in a granular-ball, the training dataset will drop sharply. Therefore, the GB_{RAIN} will have excellent scalability.

GB_{RAIN} consists of three main parts. First, the labeled data are fed into the text encoder of GB_{RAIN}, mapped as high-dimensional feature vectors. Second, a granular-ball clustering layer adaptively clusters close feature vectors, forming feature balls with similar contexts and generating central vectors and labels for each ball. Finally, the central vectors and labels of all balls are regarded as coarsely represented instances and labels, used to train a neural network. The experiments are performed on multiple DNN models with diverse datasets. We evaluate GB_{RAIN} on two popular NLP tasks, including Named Entity Recognition (NER) and text classification. For the NER task, the GB_{RAIN} framework is assessed on the pre-trained masked language model RoBERTa. For text classification tasks, GB_{RAIN} is evaluated on traditional sequence models such as BiLSTM, GRU, and TextCNN. Furthermore, we compare GB_{RAIN} with state of arts, i.e. BOND [20], SCDL [42], and RoSTER [25]. Experimental results demonstrate that GB_{RAIN} presents good robustness and outperforms existing work under label noise. In a nutshell, we highlight our main contributions as follows:

- We put forward an effective granular-ball clustering algorithm for high-dimensional vectors, which can be embedded into diverse DNN models to cluster similar texts with data feature preserving.
- We propose a robust training framework based on coarse-grained representation to help DNN models fight against label noise in NLP.
- Broad evaluations on popular NLP tasks demonstrate that our proposal provides better robustness than existing work. Our codes and datasets are released for reproduction: https://github.com/zhangtuo723/GBRAIN.

2 RELATED WORK

2.1 Countermeasures against Label Noise in CV

Sukhbaatar et al. [29] incorporate an additional noise layer in a neural network to adjust the output to match the noise distribution. Manwani et al. [24] demonstrate that risk minimization using the loss function 0-1 exhibited noise tolerance characteristics, while the square error loss only tolerates uniform noise. Jindal et al. [18] enhance a standard deep network by incorporating a SoftMax layer that models label noise statistics. Zhuang et al. [44] propose a robust, weakly supervised deep learning framework to combat noisy labels in network images. Wang et al. [31] develop a symmetric cross-entropy learning method, symmetrically improving the cross-entropy using reverse cross-entropy corresponding to robust noise. Some researchers use noise filtering methods to reduce the impact of label noise on the model. Han et al. [12] introduce a co-learning noise memory method, which involves two networks collaboratively learning on small batches of data, effectively filtering noise samples. Yao et al. [39] propose a learning method called "search to explore", to adaptively determine the forgetting rate compared to the manual setting in [12]. Azadi et al. [1] introduce an auxiliary image regularization technique that automatically uses mutual contextual information between training samples to encourage models

to select reliable samples to improve learning. Jiang et al. [17] supervise the training of basic deep networks (i.e., StudentNet) by providing a course (sample weight scheme) to focus on samples with potentially correct labels. Jie et al. [15] suggest periodically changing the learning rate, causing the model to oscillate between overfitting and underfitting to detect noise label samples. Jindal et al. [19] introduce a non-linear processing layer to model data with incorrect labels, preventing the model from overfitting noise.

2.2 Countermeasures against Label Noise in NLP

Most researchers always mix robust optimization and noise filtering to improve the robustness of the model against label noise in NLP. Zhang et al. [42] propose self-cooperative noise reduction learning, training two teacher-student networks, with each network using reliable labels through self-denoising and exploring unreliable annotations through collaborative denoising. Liang et al. [20] first use a pre-trained language model adapted to NER tasks to improve accuracy, then employ distant label removal and self-training to enhance robustness. Meng et al. [25] propose a noise-robust learning scheme comprising a new loss function and a noise label deletion process, training the network model to label data robustly. Qiao et al. [27] propose a method called SelfMix, which effectively addresses label noise in text classification tasks by utilizing the Gaussian mixture model, semi-supervised learning, dropout mechanism, and a textual-level mixup training strategy. Cheng et al. [6] first conduct a warm-up process on the training set to dynamically terminate based on the model's adaptability to different noise scenarios. A hybrid training stage with various generalization strategies is then used to gradually correct mislabeled instances. Garg et al. [9] introduce a two-component beta mixture model, assigning probability scores with clean or noisy labels to each sample before training the classifier and noise model using denoising losses.

2.3 Granular-Ball Computing

Combining granular computing and research in the magazine of Science [3], Wang proposes granular cognitive computing [30]. Based on this, Xia [33] proposes granular-ball computing (GBC), which generates multiple Granular-Balls (GB) that cover the entire dataset based on the distribution of the data. Each GB is described by a center and a radius, where the center is the mean of the vectors of all objects in the ball, and the radius is the maximum or average distance from all objects in the GB to its center. All GBs that replace the original objects are used as input for downstream tasks, achieving robustness. Cheng et al. [5] propose the introduction of the granular-ball representation and present a novel clustering algorithm called GB-DP (granular-ball-based Density Peaks), aimed at addressing the challenge of clustering large-scale data. Xie et al. [37] propose a granular ball-based spectral clustering algorithm to reduce time and resource costs in large-scale data. Compared with traditional methods, the algorithm is more efficient and robust. Xia et al. [35] propose the Granular-Ball Neighborhood Rough Sets (GBNRS) method, addressing the efficiency and effectiveness issues caused by the fixed neighborhood radius in neighborhood rough sets. By adaptively generating different neighborhoods, GBNRS significantly improves generality and flexibility.

3 OUR PROPOSED SOLUTION: GB$_{\mathbf{RAIN}}$

Figure 1 presents an overview of our framework. GB_{RAIN} takes as input a training dataset with label noise. Each sample in the dataset is a sentence with a label, which is not necessarily correct. Then the input texts are first processed by an encoder, which is a representation network. These inputs are mapped into a high-dimensional representation space, called feature vectors. Subsequently, GBC layer cluster close inputs based on their feature vector, to construct multi-granularity corpuses (i.e., granular-balls). The representation and label of all samples will be re-expressed based on granular-balls, to correct label noise while preserving data features. The represented instances are further fed into a classifier, which relies on the output probability distribution to predict their labels. Then a loss value can be computed based on the prediction and re-represented label, which reflects the model's performance on the training data. Through backpropagation and reducing loss in the iterative training, the model can be optimized, and figure out the relationship between coarse-grained representation and the corrected labels. Therefore, the adverse impact of label noise on model performance in the training can be alleviated. The model can be more generalized and robust in the inference procedure. As introduced above, GB_{RAIN} includes three modules: encoder, GBC-layer, and classifier. We claim that GB_{RAIN} is a general framework, so the encoder module can be any DNN model that can represent input texts as feature vectors, such as LSTM, CNN, and BERT. Similarly, the classifier can be any classifier network. The core of our framework is the GBC layer, which mainly consists of input re-representation (Section 3.1) and representation optimization (Section 3.2). We will detailedly introduce them in the next subsections.

3.1 Input Re-representaion by Granular-ball

This module aims at correcting noise labels while preserving data features based on clustering similar samples. To maintain the raw features of data, this module will re-represent inputs through coarse-grained representation learning. Therefore, the main tasks of this module are clustering and re-representing samples.

As for the clustering task, the samples have been mapped into high-dimensional feature vectors, after being processed by the text encoder as shown in Figure 1. Traditional clustering methods are either limited to low-dimensional data or not correct enough because of predefining the size of sets. Based on granular-ball computing, we design an adaptive clustering method for high-dimension data without configuring the set size. The whole process is shown in algorithm 1. Specifically, The algorithm takes a batch of feature vectors (i.e. the embedding of some input texts), denoted as $\mathcal{D}$, and a user-defined purity threshold T as input. The output of the algorithm is all GBs that cover all samples in the input dataset, together with the label and center vector of these GBs. A GB is a collection of similar input samples as shown in Definition 3.1. Purity is a metric to assess the quality of a GB, seeing Definition 3.2. The algorithm begins by initializing a queue Q as an empty set. Q maintains a list of intermediate sets, the purity of which does not reach the threshold T, waiting for further optimization. Subsequently, it adds the input dataset $\mathcal{D}$ to the empty queue Q. Next, the algorithm iteratively processes every set Q_i in the queue to create satisfying granular-balls from lines 3 to 11. For each set, the algorithm first calculates its purity value through the function

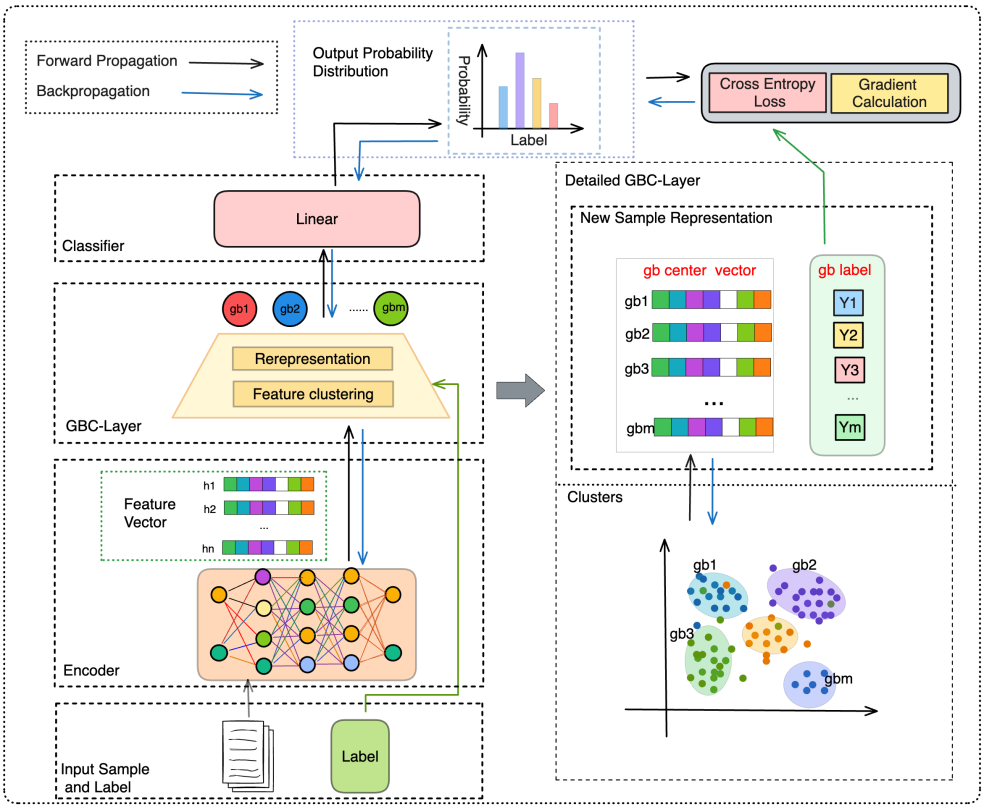


Figure 1: Overview of GB_{RAIN}.

getPurity() (line 6), by computing the ratio of the number of samples of the group with the most samples and the number of all samples in Q_i as presented in Definition 3.2. If the purity value is less than the threshold, the function *splitSet*() will further split Q_i into m subsets (line 8), whose values of purity all satisfy the threshold T. Otherwise, the current set Q_i satisfies the requirements of granular-balls and will be recorded in the output *GBs*. Finally, when the algorithm converges, almost all input samples are in certain granular-balls, and the union of all balls is approximately equal to the entire input. The details of this function are shown from lines 17 to 31 in algorithm 1. The *splitSet*() function iteratively splits each unsatisfying cluster Q in its waiting queue S, until the purity of all clusters satisfies the threshold T. Detailedly, for each cluster, it randomly chooses K samples as new cluster centers. k is equal to the number of label types. A sample in Q will be assigned to a new cluster Q_{c_j}, if the center vector of Q_{c_j} is the closest to the sample than other centers (lines 23-26). After all samples in Q are reorganized into new clusters, the function will update the center vector of each cluster based on their contained samples. The whole clustering procedure is adaptive without predefining the number of final clusters.

Definition 3.1 (Granular-ball). Given a batch of samples $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, x_i is the feature vector of a sample, y_i is the label of the sample,

$$Granular\ Ball = \{(x_{s1}, y_{s1}), ..., (x_{sm}, y_{sm})\} \tag{1}$$

$$Y_{GB} = \arg\max_{p} \left(\frac{\text{count}(y_{si}) = p)}{m} \right) \tag{2}$$

$$C_{GB} = \frac{1}{N} \sum_{i=1}^{N} x_{si} \tag{3}$$

where $(x_{si}, y_{si}) \in \mathcal{D}$, i.e., $Granular\ Ball \subseteq \mathcal{D}$, m is the size of *granular ball*, and all feature vectors x_{si} are close within a predefined distance. C_{GB} and Y_{GB} are the center vector and the label of current *granular-ball*.

Definition 3.2. Given a GB, which is made up of k classes of samples, that is, $\bigcup_{i=1}^{k} c_i$ = GB and $\bigcap_{i=1}^{k} c_i = \phi$, where c_i represents the i-th class of samples in GB. The purity T_{GB} of GB can be expressed as

$$T_{GB} = \frac{max|c_i|}{|GB|}, i = 1, 2, ..., k \tag{4}$$

where $|.|$ represents the number of samples in a set.

As introduced above, our clustering method is more robust and compatible with a high-dimensional feature space, compared to existing work. Unlike traditional clustering methods such as k-means that require predefining the number of sets, the GB clustering algorithm adaptly decides whether splitting a set and how much subsets are split out based on purity and the number of label types, enabling the algorithm to automatically partition and cluster without limiting the set number. Hence, our method is flexible

Algorithm 1: The clustering algorithm of GB_{RAIN}

Data: $\mathcal{D}$: a batch of samples, T: the threshold of purity
Result: GBs: all granular balls that cover the whole input data, together with their center vector and label

Initializing: $Q = \emptyset, GBs_{init} = \emptyset$
$Q = Q \cup \mathcal{D}$
while Q *is not empty* **do**
  for Q_i ***in*** Q **do**
    $Q \leftarrow Q - Q_i$
    $T_{Q_i} \leftarrow getPurity(Q_i)$
    if $T_{Q_i} < T$ **then**
      $[Q_{i-child1}, \dots Q_{i-childm}] \leftarrow splitSet(Q_i)$
      $Q = Q \cup [Q_{i-child1}, \dots Q_{i-childm}]$
    else
      $GBs_{init} = GBs_{init} \cup Q_i$
for *gb in* ***in*** GBs_{init} **do**
  $C_{gb} = getCenter(gb)$ (using Eq 3)
  $Lable_{gb} = getLabel(gb)$ (using Eq 2)
  $GBs = GBs \cup gb(C_{gb}, Lable_{gb})$
return GBs
Function splitSet(Q_i):
$\mathcal{S} = Q_i$
  repeat
    for *each cluster* $Q \in \mathcal{S}$ *and* $T_Q < T$ **do**
    $\mathcal{S} = \mathcal{S} - Q$
    Randomly select K data points from Q_i as new cluster centers: $C_1, C_2, \dots, C_K$
      for *each data point* $x_i \in Q_i$ **do**
        Calculate the distance to each C_j using Eq 6:
        $d_{ij} = Dis(x_i, C_j)$ for $j = 1, 2, \dots, K$
        Assign x_i to the nearest cluster:
        $c_j = \arg\min_j d_{ij}, Q_{c_j} \leftarrow x_i$
      for *each new cluster* Q_{C_j} **do**
        update its center:
        $C_j = \frac{1}{|\{x_m | C_{x_m} = C_j\}|} \sum_{x_m \in C_j} x_m$
        $\mathcal{S} = \mathcal{S} \cup Q_{C_j}$
until *Convergence*;
return $\mathcal{S}$

enough to create multi-granularity sample sets with context-similar, which facilitates the generated clusters to better fit the data feature distribution, resulting in more robust clusters. This characteristic endows GB clustering with broader applicability, particularly in scenarios with an unknown number of clusters. In addition, the traditional clustering method cannot necessarily be applied to high-dimensional spaces. It is because traditional distance computations may face the challenge of the "curse of dimensionality", where distance calculations become complicated in high-dimensional data. Our method addresses this issue by introducing the concept of weighted distance. It assigns various weights to different dimensions during distance calculations, improving influences of the dimensions with large differences as shown in Definition 3.3. This strategy is better compatible with high-dimensional spaces, enhancing performance and efficiency in such environments. Overall, our input re-representation will be more robust and precise under the support of such a multi-granularity clustering method.

Definition 3.3. Given two high-dimensional vectors $\mathbf{x}(x_1, x_2, ..., x_n)$ and $\mathbf{y}(y_1, y_2, ..., y_n)$, the distance between the two vectors is calculated as

$$\mathbf{d} = \mathbf{x} - \mathbf{y} = ((x_1 - y_1), (x_2 - y_2), ..., (x_n - y_n)) \tag{5}$$

$$Dis(\mathbf{x}, \mathbf{y}) = \|(\alpha_1, \alpha_2, ..., \alpha_n)\mathbf{d}^T\|_2 \tag{6}$$

where n is the dimensions of vectors, $\mathbf{d}$ is a vector with dimension n to maintain the difference between x and y, $Dis()$ represents weighted distance, α_i represent hyper-parameters with different weights, if $d_i > d_j$, then $\alpha_i > \alpha_j$.

As for the re-representing task, all samples have been divided into granular-balls after the clustering procedure. According to the principles of GB clustering, all samples in a ball are close and semantic-similar, so they should have the same label. We use the label with the most samples in the ball as the label for all samples and the ball as presented in Equation (2). Therefore, the label noise can be corrected by the majority of the accurately-annotated data. In addition, we use the center vector of the ball as the new feature vector for each sample in the ball. The center vector is the average of all sample vectors within the GB calculated as Equation (3). There are two main reasons for such embedding re-representation. First, if the GB contains some noisy data points, taking the average of the samples within the ball helps mitigate the negative impact of noise. Secondly, the clustering procedure is not accurate enough, that is, all samples in a granular-ball do not necessarily have the same label. If the samples maintain the original embeddings but with the new re-represented labels, the abnormal training samples will be created, whose embeddings do not correspond with their labels. However, the center vector contains all features of the samples in a granular-ball, and the errors can be degraded by averaging all samples. Therefore, we use the center vectors as the new embeddings for the samples. In summary, this module organizes data into multiple granularity levels during the training process. It aggregates samples with similar features into granular-balls, ultimately reducing the impact of label noise on model training.

3.2 Representation Optimization

Through the re-representation process in Section 3.1, we express all samples as new embeddings and labels. However, these representations need further optimization in multiple training rounds. To achieve this goal, we introduce the Granular-Ball Computing (GBC) layer and integrate it into both the forward and backward propagation of the model training process.The forward propagation of the GBC layer corresponds to the workflow of the *Input Re-representation* module (Section 3.1). This module acquires coarse-grained feature vectors and corresponding labels for the samples, feeding them into the classifier to train the model. Let $\mathbf{X}$ represent a batch of input vectors, where each vector is denoted as $\mathbf{x}_i$. After passing the GBC layer, multiple granular-balls containing these inputs are generated. The GB_{RAIN} network finally represents a sample as the center vector of its subordinate granular-ball.

In the backward propagation phase, we design a method to ensure the effective transmission of gradient information from the

classifier to each granular-ball, based on the backpropagation of average pooling in a convolutional neural network (CNN). Specifically, The model needs to compute the partial derivative of the loss function L concerning each input vector $\mathbf{x}$, i.e., gradients, helping optimize the model parameters. But in our scenario, the representation of each sample has been re-expressed by the center vector $\mathbf{c}_j$ of its subordinate granular-ball gb_j, so the backpropagation must first pass through the balls, then reach the single sample. To realize this propagation, we directly compute the gradients of loss function L concerning the center vector of granular-balls, and regard them as the gradient of their contained samples as shown in Definition 3.4. In other words, for samples belonging to a granular-ball gb_j, their gradient is determined by the gradient of the loss concerning the center vector $\mathbf{c}_j$ of gb_j. This process is applied to all granular-balls and samples in the entire training process. This ensures that gradients in backpropagation are transmitted from granular-balls to each single sample, akin to copying the residual to each sample in backpropagation of average pooling. The biggest difference from it is that we do not use the mean of granular-ball gradient (i.e., $(\frac{\partial \mathbf{L}}{\partial \mathbf{c}_j})/|gb_j|$) as the gradient of each sample. Instead, we employ a weighting strategy. Specifically, our gradient calculations are assigned different weights (γ) based on the current number of samples in the granular ball (gb). We set a threshold, currently set to 3; if the number of samples in a gb is less than 3, then γ is set to 0, otherwise, γ is set to 1.

Definition 3.4. Given an input vector $\mathbf{x}_i$, and a granular-ball gb_j with a center vector $\mathbf{c}_j$, the gradients of loss function L concerning $\mathbf{x}_i$ is computed as

$$\frac{\partial \mathbf{L}}{\partial \mathbf{x}_i} = \gamma \frac{\partial \mathbf{L}}{\partial \mathbf{c}_j}, \text{ for } \mathbf{x}_i \in gb_j, \gamma = (0 \| 1)$$

4 EXPERIMENTS

We conducted sufficient experiments to validate the effectiveness of our proposal method on label noise. Moreover, to measure versatility, we assess our framework GB_{RAIN} on commonly encountered NLP classification tasks (i.e., text classification and named entity recognition). We will detailly introduce the details of evaluations on the two tasks in the next subsections.

4.1 Evaluations on Text Classification

This section will assess the effectiveness of GB_{RAIN} on the text classification task. To make evaluations sufficient and convincing, we assess six DNN models on two datasets. The details are as follows.

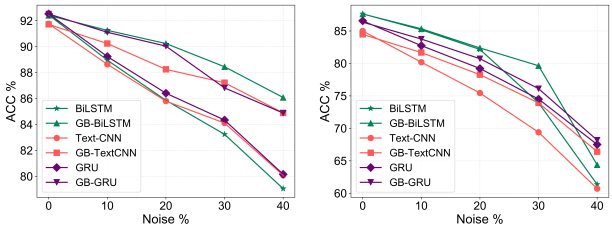
4.1.1 Experimental Setups. **Datasets.** Experiments were carried out on two prevalent datasets. The *AGNews* dataset [43] consists of 120,000 training samples and 7,600 testing samples, sourced from more than 2,000 news outlets. Each sample is a short text, labeled with one of four types indicating its category or topic. The *IMDB* dataset [23], a benchmark for sentiment analysis, comprises 50,000 movie reviews from the Internet Movie Database (IMDb) [23], divided into 25,000 training and testing samples. Each review is labeled with a binary sentiment tag, indicating positive or negative sentiments. **Models.** GB_{RAIN} is a universal framework, which can be integrated with any DNN model to improve the robustness of the current model. Therefore, we apply GB_{RAIN} to three different text classification models, BiLSTM [40], GRU [8], and Text-CNN [13], to evaluate its effectiveness against label noise. We call the models using the GB_{RAIN} framework GB-BiLSTM, GB-GRU, and GB-Text-CNN. Hence, we get six models in this evaluation. For BiLSTM and GRU, we used a two-layer network with 128 hidden units. In the case of Text-CNN, we use three convolutional layers with kernel sizes of 3, 4, and 5 respectively, and the dimension of word vectors was initialized randomly as 100. We set the maximum sequence length to 400 tokens to maintain sequence information, because the longest sample only contains 500 tokens. **Environment configurations.** The learning rate is 1e-3 during training. The experiments are run on two NVIDIA GeForce RTX 3090 GPUs.

4.1.2 Implementation Details: To validate the effectiveness of GB_{RAIN} in a noisy environment, we first introduce random noise by deliberately modifying labels in the two kinds of training sets. We implement a Python script to automatically modify the labels of a certain percentage of the datasets. The percentage ranges from 0% to 40%, which also reflects the level of noise in the related modified dataset. In such a case, we can comprehensively assess our proposal performance under different levels of noise environment, providing insights into its robustness in dealing with label errors and uncertainties. Secondly, we integrate our framework with the original evaluation models. Specifically, we insert the GBC layer of GB_{RAIN} between the encoder and the classifier of the raw models (BiLSTM, GRU, and Text-CNN) during the training process, resulting in the GB-BiLSTM, GB-GRU, and GB-Text-CNN models. Meanwhile, we use the original models as baselines for a detailed comparative study. After preparing the datasets and models, we start training the six models on the datasets with label noise. This procedure is stopped when the accuracy rate of the models remains stable. During the inference process, we assess the performance of the six models on clean datasets and datasets with varying degrees of noise. We use the accuracy rate on the testing set to represent their performance, which is computed as the number of correctly predicted samples divided by the number of all samples. In addition, to delve deeper into the factors affecting model performance, we monitor the probability of failure prediction in the whole experimental procedure at different noise levels. Specifically, each time the model completes training on one hundred batches of training samples, its performance is assessed on the test datasets. This process is repeated until the test performance becomes stable. This evaluation is evaluated only on the Bi-LSTM base model with 20% and 40% noise levels in the Agnews dataset, which are more representative in practice. We use the error rate to represent the performance of different models, which is calculated as the difference between 100% and the accuracy rate.

4.1.3 Results and Analysis: The experimental results on effectiveness are presented in Table 1. When the training datasets are clean with noise level 0%, as shown in the second and seventh columns of Table 1, the performance of our proposal is lower than the corresponding original models. For example, the accurate rate of GB-BiLSTM is inferior to that of BiLSTM on both datasets; similarly, GB-TextCNN is below Text-CNN, and GB-GRU is below GRU. Due to the clustering of vectors in a batch to generate Granular-Balls (GBs), multiple samples within a GB are replaced by a center vector. This substitution tends to reduce the diversity of samples, leading to

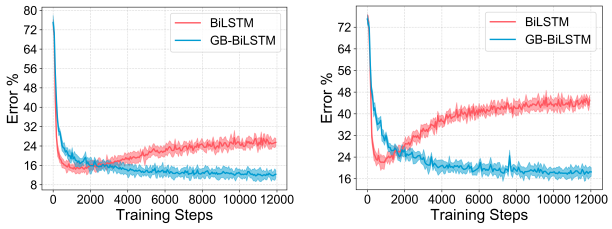
Table 1: Experimental results on the text classification task. The highlights represent the best performance (%).

Datasets	Agnews					IMDB				
Noise(%)	0	10	20	30	40	0	10	20	30	40
BiLSTM	92.45	88.96	85.86	83.24	79.05	**87.63**	85.18	82.15	73.94	61.36
Text-CNN	91.76	88.64	85.80	84.11	81.08	85.00	80.19	75.44	69.40	60.71
GRU	**92.53**	89.24	86.41	84.34	80.16	86.55	82.74	79.21	74.51	67.51
GB-BiLSTM(ours)	92.41	**91.26**	90.24	**88.45**	**86.08**	87.61	**85.33**	**82.38**	**79.64**	64.38
GB-TextCNN(ours)	91.72	90.24	88.25	87.22	84.89	84.47	81.67	78.27	73.91	66.40
GB-GRU(ours)	92.52	91.11	**90.04**	86.82	84.88	86.53	83.75	80.72	76.11	**68.18**



(a) Accuracy rate in Agnews dataset (b) Accurate rate in IMDB dataset

Figure 2: Experimental results on the accuracy rate of models with different noise level environments



(a) 20% noise in the training dataset (b) 40% noise in the training dataset

Figure 3: Experiment observations on model performance changing with training steps under various noise level environments

a slight decrease in model performance. When noises are applied to the training datasets, our GBC-based method always outperforms the original models. As with increasing noise levels, the performance of all models continues to decline, but the performance gap between the original model and our proposal becomes larger in general as Figure 2 shows. Figure 2 presents the tendency to vary in the accuracy of models with different noise levels of datasets, where different colors represent different base models, while the same color indicates the same base model. The patterns on the curve are used to distinguish different schemes using the same base model. This suggests a significant improvement in our approach GB_{RAIN} to address label noise. Specifically in highly noisy environments, the performance of GB_{RAIN} is more outstanding. It is because our model enhances robustness and generalization through coarse-grained representation learning in noisy environments so that GB_{RAIN} can better adapt to the noise of real inputs.

Moreover, Figure 3 presents the experimental results on the observation of model performance during training. As the figure illustrates, under different noise levels, the original model BiLSTM (the red line) outperforms our improved model GB-BiLSTM (the blue line) at the beginning, but after around 2000 steps, its performance starts to get worse and worse than GB-BiLSTM. When the training process converges, our model performs much better than the original model. Compared with Figure 3(a) and Figure 3(b), the performance difference between GB-BiLSTM and BiLSTM is more significant in 40% noise environments. This phenomenon suggests that BiLSTM overfits the training dataset. Due to the influence of noisy training samples, the decision boundary of the model diverts from the normal data feature distribution. When such a model runs on the normal testing sample, its prediction may make mistakes. On the contrary, GB-BiLSTM reduces the influence of noisy samples through granular-ball-based coarse-grained representation learning. That is, the embedding and labeling of abnormal samples can be optimized by the majority of correct training samples based on granular-balls. In such a case, GB-BiLSTM can avoid overfitting noisy samples and gain good robustness and performance on test samples. In summary, the evaluations demonstrate the effectiveness of our framework on text classification.

4.2 Evaluations on Named Entity Recognition

Named Entity Recognition (NER) is a sequence labeling task in NLP, focusing on identifying entities from a text sequence. To effectively perform this task, large-scale annotated corpora are usually required. However, due to the high cost of acquiring and annotating such corpora, distant supervised named entity recognition is usually used to generate distant labels for unlabeled datasets, based on external knowledge bases and dictionaries. Despite the advantages of this approach in reducing annotation costs, the limitations of knowledge bases always introduce label noise. Hence, we apply our framework GB_{RAIN} to the distant supervised NER task to assess the effectiveness. The evaluations are conducted on RoBERTa models. Further, we compare our proposal with three states of arts, BOND [20], SCDL [42] and RoSTER [25]. The details are as follows.

4.2.1 Experimental setups. **Dataset.** The evaluations are conducted on three prevalent public datasets. The *CONLL2003* dataset [28] is a well-known open domain NER dataset from the CONLL shared task [7], comprising annotated data from 1,393 English news articles with four main entity types, including persons, locations, organizations, and miscellaneous entities. The *Twitter* dataset [10] is an open-domain NER dataset from the WNUT 2016 NER shared task [38], containing annotated data from 2,400 tweets with 34,000

Table 2: Experimental results on the named entity recognition task.

Method	CoNLL2003			WikiGold			Twitter		
	F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall
Distant Match	0.714	81.13	63.75	0.478	47.9	47.63	0.358	40.34	32.22
RoBERTa	0.759	82.29	70.47	0.525	47.67	58.59	0.464	50.97	42.66
BOND	0.814	82.05	80.92	0.600	53.44	68.58	0.480	53.16	43.76
SCDL	0.836	87.96	79.82	0.640	62.25	66.12	0.510	**59.87**	44.57
RoSTER	0.854	**85.90**	84.90	0.678	64.90	71.00	-	-	-
GB-RoBERTa(ours)	**0.866**	85.50	**87.70**	**0.724**	**69.60**	**75.40**	**0.538**	53.50	**54.20**

tokens and ten entity types related to common named entities in Twitter. The *Wikigold* dataset [2] consists of a collection of articles, randomly selected from the 2008 English Wikipedia dump [32]. This dataset contains 40,000 tokens and is manually annotated with four types of entities (people, locations, organizations, and miscellaneous) from Wikipedia articles. **Model.** We choose RoBERTa [21] as the base model, a pre-trained model fine-tuned on data labeled through distant supervision in Hugging Face [16]. We apply GB_{RAIN} to RoBERTa to get our model GB-RoBERTa. We also compare GB-RoBERTa with four baselines, *Distant Match* [20], BOND [20], SCDL [42] and RoSTER [25]. *Distant Match* is a base model to realize NER with distant supervision. BOND initially trains a RoBERTa model on data labeled through distant supervision with early stopping. Subsequently, it employs a teacher-student framework for iterative training to enhance the model's performance. SCDL training two teacher-student networks in a mutually beneficial manner to iteratively refine noisy labels. RoSTER first uses a generalized cross-entropy loss function and tag noise removal, then leverages a context-enhanced language model to train the network model's generalization. **Environment configurations.** The maximum sequence lengths of *CoNLL2003*, *Twitter*, and *Wikigold* are set as 150, 100, and 120 tokens respectively. During training, the batch size of the three datasets is 128, and the learning rates of the pre-trained model and classifier are 1e-6 and 1e-4. AdamW [22] is chosen as the optimizer. The experiments are conducted on servers equipped with NVIDIA GeForce RTX 3090 GPUs.

4.2.2 Implementation Details: To evaluate our proposal, we first integrate GB_{RAIN} with the fundamental NER model (referred to as RoBERTa) [21] to obtain our enhanced models GB-RoBERTa, by inserting the GBC layer of GB_{RAIN} between the pre-trained masked language model and the NER classifier. We do not evaluate the *Distant Match* model, because its results are directly extracted from the work of Liang et al. [20], which are calculated based on comparisons between the results of distant labeling and the ground truth. In the training procedure, to simulate distant supervised NER, we train the five models, i.e., RoBERTa, BOND, SCDL, RoSTER and GB-RoBERTa on the three datasets without their annotated labels. Instead, we use the tool [20] to generate labels for training samples automatically, which acquires distant labels based on matching a distant knowledge base with the contents of the sample. Label noise will be introduced to the training dataset in the automatic annotation process. We use the F1 score, *Precision*, and *recall* of models on the testing dataset set to show model performance.

4.2.3 Results and Analysis: The experimental results are presented in Table 2. Comparing *Distant Match* and the base model RoBERTa, RoBERTa performs better on both three metrics. This validates that distant supervised NER introduces many noises, which demonstrates our noisy training environment. However, compared to four improved models, RoBERTa performs worse in all datasets. This indicates the serious influence of label noise on DNN model performance, which may cause models to overfit label noise and generalization decrease. Research on mitigating label noise is urgent. When comparing our GB-RoBERTa solution with existing work, GB-RoBERTa has the best F1 score and *recall* on all datasets. However, it performs worse given the *precision* on *CONLL2003* an *Twitter* dataset. In particular, *recall* of GB-RoBERTa has a distinct increase. In general, these phenomena suggest our proposal has great effectiveness in capturing entities within noisy texts, which can identify entities correctly and reduce false negatives at the same time. Our framework also always outperforms existing work. The advantages of GB-RoBERTa may be sourced from the multi-granularity representation method in our robust training framework. All in all, these evaluations illustrate the effectiveness of our framework in distant-supervised NER.

5 CONCLUSION

In this paper, we propose a new Granular-Ball based t$_{RAIN}$ing framework, named GB_{RAIN}, to mitigate the impact of label noises on the performance of neural network models. The main advantage of GB_{RAIN} is its ability to learn coarse-grainedly the embedding vectors for all samples and then adaptively rectify their labels. On top of that, GB_{RAIN} also introduces a new backpropagation mechanism to further refine the coarse-grained embedding vectors with iterative training. Experimental results on entity recognition and text classification tasks show that our proposed solutions significantly reduce the impact of label noises on the traditional neural network model when contrasted with the state-of-the-art baselines.

6 ACKNOWLEDGEMENT

This work was supported by the National Natural Science Foundation of China under Grant Nos. 62222601, 62176033, 62221005 and 61936001, key cooperation project of Chongqing municipal education commission No. HZ2021008 and Natural Science Foundation of Chongqing (cstc2022ycjh-bgzxm0128, cstc2021ycjh-bgzxm0013, CSTB2023NSCQ-JQX0034 and CSTB2022NSCQ-MSX1588).

REFERENCES

[1] Samaneh Azadi, Jiashi Feng, Stefanie Jegelka, and Trevor Darrell. 2015. Auxiliary image regularization for deep cnns with noisy labels. *arXiv preprint arXiv:1511.07069* (2015).

[2] Dominic Balasuriya, Nicky Ringland, Joel Nothman, Tara Murphy, and James R Curran. 2009. Named entity recognition in wikipedia. In *Proceedings of the 2009 workshop on the people's web meets NLP: Collaboratively constructed semantic resources (People's Web)*. 10–18.

[3] Lin Chen. 1982. Topological structure in visual perception. *Science* 218, 4573 (1982), 699–700.

[4] Robert S Chen, Brendan Lucier, Yaron Singer, and Vasilis Syrgkanis. 2017. Robust optimization for non-convex objectives. *Advances in Neural Information Processing Systems* 30 (2017).

[5] Dongdong Cheng, Ya Li, Shuyin Xia, Guoyin Wang, Jinlong Huang, and Sulan Zhang. 2023. A fast granular-ball-based density peaks clustering algorithm for large-scale data. *IEEE Transactions on Neural Networks and Learning Systems* (2023).

[6] Shaohuan Cheng, Wenyu Chen, Fu Mingsheng, Xuanting Xie, and Hong Qu. 2023. Adaptive Textual Label Noise Learning based on Pre-trained Models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. 3174–3188.

[7] CoNLL. 2003. *Conference on Natural Language Learning*. https://www.conll.org/.

[8] Rahul Dey and Fathi M Salem. 2017. Gate-variants of gated recurrent unit (GRU) neural networks. In *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*. IEEE, 1597–1600.

[9] Siddhant Garg, Goutham Ramakrishnan, and Varun Thumbe. 2021. Towards robustness to label noise in text classification via noise modeling. In *Proceedings Of The 30th ACM International Conference On Information & Knowledge Management*. 3024–3028.

[10] Fréderic Godin, Baptist Vandersmissen, Wesley De Neve, and Rik Van de Walle. 2015. Multimedia lab@ acl wnut ner shared task: Named entity recognition for twitter microposts using distributed word representations. In *Proceedings of the workshop on noisy user-generated text*. 146–153.

[11] Sheng Guo, Weilin Huang, Haozhi Zhang, Chenfan Zhuang, Dengke Dong, Matthew R Scott, and Dinglong Huang. 2018. Curriculumnet: Weakly supervised learning from large-scale web images. In *Proceedings of the European conference on computer vision (ECCV)*. 135–150.

[12] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems* 31 (2018).

[13] Tong He, Weilin Huang, Yu Qiao, and Jian Yao. 2016. Text-attentional convolutional neural network for scene text detection. *IEEE transactions on image processing* 25, 6 (2016), 2529–2541.

[14] Dan Hendrycks and Kevin Gimpel. 2016. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136* (2016).

[15] Jinchi Huang, Lie Qu, Rongfei Jia, and Binqiang Zhao. 2019. O2U-Net: a simple noisy label detection approach for deep neural networks. in 2019 IEEE. In *CVF International Conference on Computer Vision (ICCV)*, Vol. 80. 3325–3333.

[16] Hugging Face. 2024. Transformers: State-of-the-art Natural Language Processing. *https://huggingface.co/* (2024).

[17] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International conference on machine learning*. PMLR, 2304–2313.

[18] Ishan Jindal, Matthew Nokleby, and Xuewen Chen. 2016. Learning deep networks from noisy labels with dropout regularization. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 967–972.

[19] Ishan Jindal, Daniel Pressel, Brian Lester, and Matthew Nokleby. 2019. An effective label noise model for dnn text classification. *arXiv preprint arXiv:1903.07507* (2019).

[20] Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. 2020. Bond: Bert-assisted open-domain named entity recognition with distant supervision. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 1054–1064.

[21] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[22] Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam. (2018).

[23] Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*. 142–150.

[24] Naresh Manwani and PS Sastry. 2013. Noise tolerance under risk minimization. *IEEE transactions on cybernetics* 43, 3 (2013), 1146–1151.

[25] Yu Meng, Yunyi Zhang, Jiaxin Huang, Xuan Wang, Yu Zhang, Heng Ji, and Jiawei Han. 2021. Distantly-supervised named entity recognition with noise-robust learning and language model augmented self-training. *arXiv preprint arXiv:2109.05003* (2021).

[26] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. 2017. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1944–1952.

[27] Dan Qiao, Chenchen Dai, Yuyang Ding, Juntao Li, Qiang Chen, Wenliang Chen, and Min Zhang. 2022. SelfMix: Robust Learning Against Textual Label Noise with Self-Mixup Training. *arXiv preprint arXiv:2210.04525* (2022).

[28] Erik F Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050* (2003).

[29] Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. 2014. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080* (2014).

[30] Guoyin Wang. 2017. DGCC: data-driven granular cognitive computing. *Granular Computing* 2, 4 (2017), 343–355.

[31] Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. 2019. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE/CVF international conference on computer vision*. 322–330.

[32] Wikipedia. 2024. *Title of the Article*. https://en.wikipedia.org/ Accessed January 31, 2024.

[33] Shuyin Xia, Yunsheng Liu, Xin Ding, Guoyin Wang, Hong Yu, and Yuoguo Luo. 2019. Granular ball computing classifiers for efficient, scalable and robust learning. *Information Sciences* 483 (2019), 136–152.

[34] Shuyin Xia, Daowan Peng, Deyu Meng, Changqing Zhang, Guoyin Wang, Elisabeth Giem, Wei Wei, and Zizhong Chen. 2020. A fast adaptive k-means with no bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020).

[35] Shuyin Xia, Hao Zhang, Wenhua Li, Guoyin Wang, Elisabeth Giem, and Zizhong Chen. 2020. GBNRS: A novel rough set algorithm for fast adaptive attribute reduction in classification. *IEEE Transactions on Knowledge and Data Engineering* 34, 3 (2020), 1231–1242.

[36] Shuyin Xia, Shaoyuan Zheng, Guoyin Wang, Xinbo Gao, and Binggui Wang. 2021. Granular ball sampling for noisy label classification or imbalanced classification. *IEEE Transactions on Neural Networks and Learning Systems* (2021).

[37] Jiang Xie, Weiyu Kong, Shuyin Xia, Guoyin Wang, and Xinbo Gao. 2023. An Efficient Spectral Clustering Algorithm Based on Granular-Ball. *IEEE Transactions on Knowledge and Data Engineering* (2023).

[38] Wei Xu, Bo Han, and Alan Ritter. 2015. Proceedings of the workshop on noisy user-generated text. In *Proceedings of the Workshop on Noisy User-generated Text*.

[39] Quanming Yao, Hansi Yang, Bo Han, Gang Niu, and James Tin-Yau Kwok. 2020. Searching to exploit memorization effect in learning with noisy labels. In *International Conference on Machine Learning*. PMLR, 10789–10798.

[40] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. 2019. A review of recurrent neural networks: LSTM cells and network architectures. *Neural computation* 31, 7 (2019), 1235–1270.

[41] Qingjun Yuan, Gaopeng Gou, Yanbei Zhu, Yuefei Zhu, Gang Xiong, and Yongjuan Wang. 2023. MCRe: A Unified Framework for Handling Malicious Traffic with Noise Labels Based on Multidimensional Constraint Representation. *IEEE Transactions on Information Forensics and Security* (2023).

[42] Xinghua Zhang, Bowen Yu, Tingwen Liu, Zhenyu Zhang, Jiawei Sheng, Mengge Xue, and Hongbo Xu. 2021. Improving distantly-supervised named entity recognition with self-collaborative denoising learning. *arXiv preprint arXiv:2110.04429* (2021).

[43] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems* 28 (2015).

[44] Bohan Zhuang, Lingqiao Liu, Yao Li, Chunhua Shen, and Ian Reid. 2017. Attend in groups: a weakly-supervised deep learning framework for learning from web data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1878–1887.