# Extremely Weakly-supervised Text Classification with Wordsets Mining and Sync-Denoising

**Lysa Xiao**
East China Jiao Tong University / Nanchang, Jiangxi
lissssaxiao@gmail.com

## Abstract

Extremely weakly-supervised text classification aims to classify texts without any labeled data, but only relying on class names as supervision. Existing works include prompt-based and seed-based methods. Prompt-based methods prompt language model with instructions, while seed-based methods generate pseudo-labels with word matching. Both of them have significant flaws, including zero-shot instability and context-dependent ambiguities. This paper introduces SetSync, which follows a new paradigm, i.e. wordset-based, which can avoid the above problems. In SetSync, a class is represented with wordsets, and pseudo-labels are generated with wordsets matching. To facilitate this, we propose to use information bottleneck to identify class-relevant wordsets. Moreover, we regard the classifier training as a hybrid learning of semi-supervised and noisy-labels, and propose a new training strategy, termed sync-denoising. Extensive experiments on 11 datasets show that SetSync outperforms all existing prompt and seed methods, exceeding SOTA by an impressive average of 8 points.

## 1 Introduction

As a fundamental task in NLP, text classification has a wide range of real-world applications. However, the substantial annotation costs present significant challenges and obstacles to its practical implementation. As a result, extremely weakly-supervised text classification (WTC) has garnered considerable attention (Wang et al., 2023a), which requires no human-annotated datasets, but relies solely on the class names as supervision signals to perform the text classification task.

Generally, weakly-supervised text classification involves two main steps: pseudo-labels generation and text classifier training, as shown in Fig. 1. 1) Firstly, pseudo-labels for unlabeled texts are generated according to class names, which can be roughly divided into two major mainstream, namely *prompt* methods (Han et al., 2022a) and *seed* methods (Wang et al., 2021). The prompt methods generate pseudo-labels by prompting language models (e.g. GPT) with instructions, and the seed methods first expand the seed class names into a larger set of related words and then generate pseudo-labels with a matching strategy. Fig. 1 shows a specific example. 2) In the subsequent text classifier training step, the text classifier will be trained on generated pseudo-labels. The training strategies of the classifier mainly include fully-supervised training and self-training, where the former directly trains the classifier on pseudo-labeled data, and the latter utilizes self-training to exploit texts that are not assigned pseudo-labels.

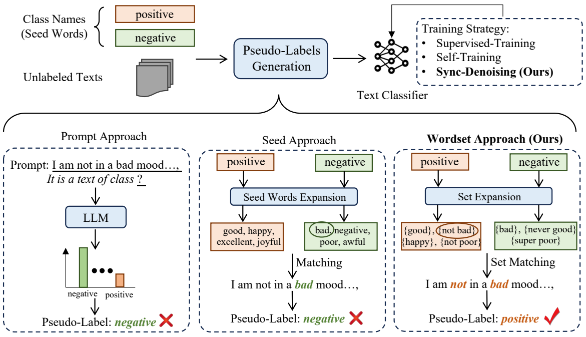


Figure 1: WTC consists of two major steps, namely pseudo-labels generation and classifier training. Firstly, we utilize wordset to generate pseudo-labels, which can avoid the problem of zero-shot instability in prompt methods and the context-dependent in seed methods. Moreover, we regard classifier training in WTC as a hybrid learning of semi-supervised and noisy-labels, and propose a new training strategy called sync-denoising.

Although previous methods have achieved some success, we observe that they are still sub-optimal: 1) Prompt methods tend to have a lower performance, as reported in a recent review (Wang et al., 2023a). This is because language models are known to exhibit bias towards text sequences more common in their pre-training data, which can lead

*Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7167–7179
June 16-21, 2024 ©2024 Association for Computational Linguistics

to instability in zero-shot settings. 2) Seed methods are known to be context-dependent and prone to ambiguity. For example, in the case of Fig. 1, the text “I am not in a bad mood” is an example of positive emotion being mislabeled as negative due to the presence of "bad" seed keywords in the negative class. In this case, the combination of “bad” and “not” can express the emotion accurately. Although there have been some methods to solve the context-dependency problem before (Mekala and Shang, 2020; Zhang et al., 2021), we believe that they are still not thorough and perfect enough. 3) Additionally, existing methods do not consider the noise of pseudo-labels in the training of classifier. Specifically, existing WTC works extensively utilize self-training to make full use of samples without any pseudo-labels (Meng et al., 2020b; Zhang et al., 2021). However, during the self-training, incorrect pseudo-labels will lead to the propagation and expansion of errors, ultimately contributing to a decrease in classifier performance.

In this paper, we solve the above problem with a novel framework called **SetSync**, where we embark on enhancing WTC from two primary perspectives: pseudo-labels generation based on word**sets** and classifier training with **sync**-denoising. To begin with, considering the context-dependency ambiguities in seed methods, we believe that individual seed words alone cannot effectively represent a category. Thus, we propose to utilize wordsets to represent a category and generate pseudo-labels, where a wordset is a collection that contains some keywords. For example, {not, bad} and {good} are two wordsets for class positive, while {never, good} is a wordset for class negative, as shown in Fig. 1. The previous seed word expansion can be seen as a special case of ours, where each wordset contains only one word. Thus, with the help of information bottleneck (IB) (Bayat and Wei, 2019) theory, we propose wordsets information bottleneck (WIB) to mine category-related wordsets. Information bottleneck was originally proposed for signal processing, attempts to find a short code of the input signal but preserve maximum information. Contrasting with the continuous nature of signal space, the space of words is inherently discrete. Applying IB directly to wordset mining will face problems such as high time complexity and difficulty in optimization. Therefore, we propose WIB in discrete space, which uses high-frequency filtering combined with frequent itemset mining for optimization.

Moreover, observing pseudo-labels, we can find it is a hybrid setting of semi-supervised and noisy-label learning, i.e. only a portion of the texts have pseudo-labels and they contain errors. Therefore, we propose a new training strategy, named sync-denoising to train the classifier, where noisy pseudo-labels and unlabeled data participate in training synchronously under a unified denoising framework. We assume that the noise levels of pseudo-labels and unlabeled data obey two independent Gaussian distributions and dynamically sample weights from them for denoising learning.

We conducted extensive experiments on 11 text benchmarks. Results show that our method substantially outperforms all existing seed-based and prompt-based methods, improving the accuracy of all 11 datasets by about 8 points on average.

## 2 Related Work

### 2.1 Weakly-supervised Text Classification

Weakly-supervised text classification (WTC) aims to use various weakly supervised signals to perform text classification. There are many sources of weak supervision signals, including: 1) external knowledge bases(Gabrilovich et al., 2007; Yin et al., 2019), 2) seed words(Meng et al., 2020b; Zhang et al., 2021; Mekala and Shang, 2020; Wang et al., 2021; Zhao et al., 2023a), 3) heuristic rules(Badene et al., 2019; Shu et al., 2020), 4) language models (prompt methods)(Holtzman et al., 2022; Han et al., 2022b). Among these, the most popular ones at present are seed-words and prompt methods, where the former generates pseudo-labels based on word matching, and the latter generates the class probability distribution of each text by prompting a large language model.

Different from existing seed-based and prompt-based methods, our SetSync belongs to a new paradigm, named wordset-based method, which can avoid the problem of zero-shot instability and context-dependent ambiguities in previous works.

### 2.2 Noisy and Semi-supervised Learning

Noisy-label learning and semi-supervised learning are two research areas that have received widespread attention. Noisy-label learning (NLL) mainly studies the learning of labels with errors or noise in a fully-supervised scenario. While semi-supervised learning (SSL) studies that only a part of the samples have labels, and these labels are all correct. For NLL, existing methods can be divided into two major categories, i.e. loss correction (Han et al., 2020; Liu et al., 2020) and sample selection (Li et al., 2020; Albert et al., 2021). For SSL, pseudo-labeling methods with confidence thresholds have gained widespread adoption(Sohn et al., 2020b; Chen et al., 2023b), which train models using pseudo-labels with prediction confidences above thresholds while discarding others.

However, existing noisy-label learning assumes that all samples have a label, and semi-supervised learning assumes that the labels in the labeled data are totally correct. Neither is adequate for the data we are faced with, where only a small part of the data has labels, and these labels are noisy. In this paper, we propose a novel training strategy, named sync-denoising, to solve the above data setting, which jointly optimizes unlabeled and noisy labeled samples with a unified denoising framework.

# 3 Method

## 3.1 Problem Definition

The input of WTC consists of two parts: 1) a series of $N$ unlabeled text $\mathcal{X} = \{x_1, x_2, ..., x_N\}$. 2) Class names for $C$ categories, represented as $\mathcal{M} = \{m_1, m_2, ..., m_C\}$. Our goal is to train a text classifier, with only class names as supervision signals, whose performance is evaluated on an additional test set.

## 3.2 Framework

The framework of SetSync is shown in Fig. 2, which follows an iterative paradigm. The system is initialized with class names as wordsets. In each iteration, we first generate pseudo-labels for unlabeled texts with set matching. Then, we train the text classifier with the proposed sync-denoising. Finally, we perform wordset mining with the assistance of information bottleneck. The mined wordsets will be used to generate pseudo-labels for the next iteration. When the total number of iterations reaches the threshold, the iteration stops.

## 3.3 Pseudo-Labels Generation

Given the unlabeled texts $\mathcal{X} = \{x_1, x_2, ...x_N\}$ and wordsets $\mathcal{S} = \{\mathcal{S}^1, \mathcal{S}^2, ...\mathcal{S}^C\}$ for $C$ classes, where $\mathcal{S}^c = \{s_1^c, s_2^c, ...s_T^c\}$ is $T$ wordsets for class $c$, the pseudo-label for text $x_i$ is generated with wordset matching as follows:

$$\hat{y}_i^l = \arg\max_c \{\sum_{j=1}^{T} \mathbb{I}(s_j^c \in x_i) | \forall(s_j^c \in \mathcal{S}^c)\} \quad (1)$$

where $\mathbb{I}(\cdot)$ is indicator function. Here, we count the number of wordsets of each category contained in text $x_i$. Typically, a text includes wordsets from a single category, which becomes the pseudo-label. While sometimes, a text contains wordsets from multiple classes, and we take the category with the most occurrences as the pseudo-label.

**Initialization:** At the beginning of the first round of training, we initialize the wordset $\mathcal{S}$ using the provided class names $\mathcal{M}$, that is $\mathcal{S}^c = \{s_1^c\}$, and $s_1^c = \{m_c\}$, i.e. for class $c$, it is initialized with one wordset that contains only a class name.

## 3.4 Sync-Denoising Training

According to Eq.(1), some texts will be assigned with pseudo-labels. However, there are still some texts that do not contain any wordsets, i.e. remain unlabeled, whose proportion of first-round is visualized in Fig. 3.

Such training data will bring two important issues: 1) How to leverage unlabeled texts without pseudo-labels. 2) How to learn in the presence of errors and noise in pseudo-labels. The former is a problem addressed by semi-supervised learning (SSL), while the latter is tackled by noisy-label learning (NLL). However, in SSL, it is assumed that all labels are correct, whereas in NLL, it is assumed that all samples have noisy labels. This highlights a difference in our assumptions.

Existing WTC works directly use self-training (an SSL method) to make use of unlabeled texts. However, we observe that such a strategy will lead to the further expansion and spread of pseudo-labeling errors, as shown in Fig. 3.

In this paper, we try to solve the learning in this data setting, where the training data consists of pseudo-labeled data $\hat{\mathcal{D}}_L = \{x_i^l, \hat{y}_i^l\}_{i=1}^{N_L}$ and unlabeled data $\mathcal{D}_U = \{x_i^u\}_{i=1}^{N_U}$, and the pseudo-labels $\{\hat{y}_i^l\}_{i=1}^{N_L}$ are noisy. We proposed a sync-denoising training strategy, which jointly optimize pseudo-labeled and unlabeled data through a unified denoising framework, where the sample weights of both are sampled through two independent Gaussian distributions.

### 3.4.1 Revisit Semi-supervised Learning

In recent research on semi-supervised learning (Sohn et al., 2020b; Chen et al., 2023a), the labeled data $\mathcal{D}_L$ and unlabeled data $\mathcal{D}_U$ are optimized with objective $\mathcal{L} = \mathcal{L}_s + \mathcal{L}_u$, where $\mathcal{L}_s$ is the supervised loss with ground-truth labels $\{y_i^l\}_{i=1}^{N_L}$ in

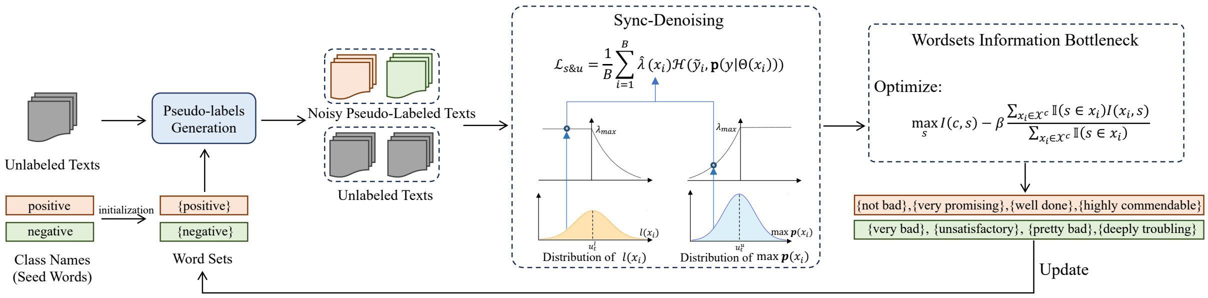

Unlabeled Texts
Pseudo-labels Generation
Noisy Pseudo-Labeled Texts
Unlabeled Texts
positive
negative
Class Names (Seed Words)
initialization
{positive}
{negative}
Word Sets
Sync-Denoising
$\mathcal{L}_{s\&u} = \frac{1}{B}\sum_{i=1}^{B} \hat{\lambda}(x_i)\mathcal{H}(\tilde{y}_i, \mathbf{p}(y|\Theta(x_i)))$
$\lambda_{max}$
$\lambda_{max}$
$l(x_i)$
$u_t^l$
Distribution of $l(x_i)$
$\max \boldsymbol{p}(x_i)$
$u_t^u$
Distribution of $\max \boldsymbol{p}(x_i)$
Wordsets Information Bottleneck
Optimize:
$\max_s I(c,s) - \beta \frac{\sum_{x_i \in \mathcal{X}^c} \mathbb{I}(s \in x_i) I(x_i, s)}{\sum_{x_i \in \mathcal{X}^c} \mathbb{I}(s \in x_i)}$
{not bad},{very promising},{well done},{highly commendable}
{very bad}, {unsatisfactory}, {pretty bad},{deeply troubling}
Update

Figure 2: In SetSync, an iteration starts with class names as initial wordsets. Each iteration involves generating pseudo-labels via set matching, training the classifier with sync-denoising, and conducting wordset mining with wordsets information bottleneck. The mined wordsets are then updated to the initial sets for the next cycle.

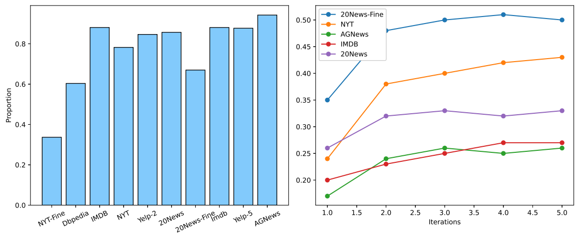


Figure 3: Left: The proportion of texts that keep unlabeled. Right: The proportion of errors in pseudo-labels gradually increases with iterations of self-training.

$\mathcal{D}_L$, and $\mathcal{L}_u$ is unsupervised loss with consistency between weak and strong augmentation in $\mathcal{D}_U$.

Formally, the supervised loss $\mathcal{L}_s$ of the cross-entropy loss $\mathcal{H}$ on $B_L$-sized batch is formulated as:

$$\mathcal{L}_s = \frac{1}{B_L}\sum_{i=1}^{B_L} \mathcal{H}(y_i^l, \mathbf{p}(y|x_i^l)) \tag{2}$$

where $\mathbf{p}(y|x_i^l) \in \mathbb{R}^C$ is the model's prediction.

For the unsupervised loss $\mathcal{L}_u$, the consistency between weak augmentation $\omega(x_i^u)$ and strong augmentation $\Omega(x_i^u)$ on $B_U$-sized batch of unlabeled data is constrained by the following formula:

$$\mathcal{L}_u = \frac{1}{B_U}\sum_{i=1}^{B_U} \lambda(x_i^u)\mathcal{H}(\widehat{\mathbf{p}}_i, \mathbf{p}(y|\Omega(x_i^u))) \tag{3}$$

where $\widehat{\mathbf{p}}_i = \arg\max \mathbf{p}(y|\omega(x_i^u))$, i.e. the one-hot pseudo-label generated from weak augmentation. $\lambda(x_i^u)$ is the binary or continuous weight of $x_i^u$, which is used to filter out samples that may be predicted incorrectly.

### 3.4.2 Synchronous Denoising

Here, we impose label-denoising training on SSL. Coincidentally, Eq.(3) utilize $\lambda(x_i^u)$ to filter unconfidence predictions, which inspires us to use the same paradigm to denoise noisy labeled data $\hat{\mathcal{D}}_L$. Hence, we employ a unified framework to simultaneously conduct denoising learning of labeled data and unsupervised learning of unlabeled data, both sharing the same optimization formula as follows:

$$\mathcal{L}_{s\&u} = \frac{1}{B}\sum_{i=1}^{B} \widehat{\lambda}(x_i)\mathcal{H}(\tilde{y}_i, \mathbf{p}(y|\Theta(x_i))) \tag{4}$$

In Eq.(4), we optimize both labeled and unlabeled data with weighted cross-entropy loss, where $B$ is the batch size, and $\widehat{\lambda}(x_i)$ is the sample weight. $\Theta(x_i)$ is the mixture of weak and strong augmentation of sample $x_i$, formally $\Theta(x_i) = \{\omega(x_i), \Omega(x_i)\}$. In our unified framework, we simultaneously predict weak and strong augmentation samples, with different target $\tilde{y}_i$.

For target $\tilde{y}_i$, according to whether $x_i$ is pseudo-labeled or unlabeled, it is formulated as:

$$\tilde{y}_i = \begin{cases} \hat{y}_i^l, & \text{if } x_i \in \hat{\mathcal{D}}_L \\ \arg\max \mathbf{p}(y|\omega(x_i)), & \text{if } x_i \in \mathcal{D}_U \end{cases} \tag{5}$$

where the pseudo-labels from Eq.(1) and prediction from weak augmentation is utilized as a target for pseudo-labeled and unlabeled data, respectively.

### 3.4.3 Dynamic Gaussian Weight

For the sample weight $\widehat{\lambda}(x_i)$, existing works of NLL measure the probability of label errors based on the magnitude of the loss function value (Kim et al., 2022). While the semi-supervised learning derives weight $\widehat{\lambda}(x_i)$ from the prediction confidence of $x_i$ (Sohn et al., 2020a; Chen et al., 2023a).

Here, we go one step further. We assume that the underlying probability mass function of loss values in pseudo-labeled samples $\hat{\mathcal{D}}_L$ and the prediction confidence in unlabeled samples $\mathcal{D}_U$ obey two dynamic Gaussian distributions respectively, where the mean and variance are $\mu_t^l$, $\sigma_t^l$ for loss values in $\hat{\mathcal{D}}_L$, and $\mu_t^u$, $\sigma_t^u$ for prediction confidence in $\mathcal{D}_U$. Since samples with higher loss values in $\hat{\mathcal{D}}_L$ are more prone to be erroneous than those with lower loss values, we can convert the deviation of the sample loss values from the mean $\mu_t^l$ of the Gaussian distribution into the probability of label correctness, which will be used as weight $\widehat{\lambda}(x_i)$. Similarly, the greater the prediction confidence in $\mathcal{D}_U$, the greater the probability of being correct. Therefore, $\widehat{\lambda}(x_i)$ can be derived as:

$$\widehat{\lambda}(x_i) = \begin{cases} \lambda_{\max} \exp(-\frac{(l(x_i)-\mu_t^l)^2}{2\sigma_t^{l\,2}}), \\ \qquad\qquad \text{if } x_i \in \mathcal{D}_L \text{ and } l(x_i) > \mu_t^l \\ \lambda_{\max} \exp(-\frac{(\max \mathbf{p}(x_i)-\mu_t^u)^2}{2\sigma_t^{u\,2}}), \\ \qquad\qquad \text{if } x_i \in \mathcal{D}_U \text{ and } \max \mathbf{p}(x_i) < \mu_t^u \\ \lambda_{\max}, \qquad\qquad \text{otherwise} \end{cases} \tag{6}$$

where $l(x_i)$ is the loss value of labeled sample $x_i$, and $l(x_i) = \mathcal{H}(\hat{y}_i^l, \mathbf{p}(y|\Theta(x_i)))$. $\max \mathbf{p}(x_i)$ is the prediction confidence of unlabeled sample, and is an abbreviation of $\max \mathbf{p}(y|\Theta(x_i))$. $\lambda_{\max}$ is the max value of weight, which is a hyperparameter.

In Eq.(6), we applied distinct truncations to two Gaussian distributions. For pseudo-labeled data $\hat{\mathcal{D}}_L$, we truncate the part of the Gaussian distribution where the loss value $l(x_i)$ is less than the mean $\mu_t^l$, making it $\lambda_{\max}$. For loss value that larger than $\mu_t^l$, since it's more likely to have the wrong label, we down-weight it according to how far it deviates from $\mu_t^l$. For unlabeled data $\mathcal{D}_U$, it is the opposite; Since higher confidence corresponds to lower error probability, we truncate in the opposite direction.

#### 3.4.4 Parameter Estimation

Then, we estimate the parameters of two Gaussian distributions, i.e. $\mu_t^l$, $\sigma_t^l$ and $\mu_t^u$, $\sigma_t^u$, separately. Since the training process is a dynamic process, the two Gaussian distribution are also changing dynamically. Therefore, our parameters are also updated with momentum. In particular, for $\mu_t^l$, $\sigma_t^l$ of labeled data, we calculate the empirical mean and variance of the loss values in the batch of $t^{th}$ iteration as:

$$\mu_b^l = \frac{1}{B}\sum_{i=1}^{B} l(x_i) \qquad \sigma_b^l = \frac{1}{B}\sum_{i=1}^{B}(l(x_i) - \mu_b^l)^2 \tag{7}$$

Then, the parameters at $t^{th}$ is estimated with EMA with unbiased variance as follows:

$$\begin{aligned} \mu_t^l &= \epsilon \cdot \mu_b^l + (1-\epsilon)\cdot \mu_{t-1}^l \\ \sigma_t^l &= \epsilon \cdot \frac{B}{B-1} \cdot \sigma_b^l + (1-\epsilon)\cdot \sigma_{t-1}^l \end{aligned} \tag{8}$$

where $\epsilon$ is a hyperparameter. Similarly, the parameters $\mu_t^u$, $\sigma_t^u$ for unlabeled data are also estimated in the same way. According to the real-time estimated Gaussian parameters, we employ Eq.(6) for the dynamic weight generation and utilize Eq.(4) to optimize the classifier.

### 3.5 Wordsets Information Bottleneck

As mentioned before, we found that seed words alone are not enough to accurately characterize a category, so we utilize wordsets to represent a category and generate pseudo-labels. In this paper, we propose to mine class-relevant wordsets with information bottleneck (IB) (Bayat and Wei, 2019).

Given the input signal $X$ and label $Y$, the objective of IB is maximized to find the most informative yet compressed representation $Z$ by optimizing:

$$\max_Z I(Y, Z) \quad s.t. I(X, Z) \leq I_x \tag{9}$$

where $I(\cdot,\cdot)$ is mutual information, $I_x$ is information constraint between $X$ and $Z$. By introducing a Lagrange multiplier $\beta$, we can get the unconstrained form:

$$\max_Z I(Y, Z) - \beta I(X, Z) \tag{10}$$

where the hyperparameter $\beta$ can be used to tradeoff informativeness and compression.

Here, in wordset mining, we focus on mining the wordsets that is compressed with minimum information loss in terms of class properties, and we propose the wordsets information bottleneck (WIB). In particular, for a class $c$, the maximally informative yet compressed wordsets can be obtained with WIB by optimizing the following objective:

$$\max_s I(c, s) - \beta \frac{\sum_{x_i \in \mathcal{X}^c} \mathbb{I}(s \in x_i) I(x_i, s)}{\sum_{x_i \in \mathcal{X}^c} \mathbb{I}(s \in x_i)} \tag{11}$$

where $s$ is the wordset to be solved. $\mathcal{X}^c$ is the set of texts predicted by the classifier to be class $c$. We use the average mutual information of a wordset and the text containing it to calculate $I(X, Z)$.

Unlike continuous space in signal processing, words are discrete space and non-derivable. To solve Eq.(11), an intuitive idea in discrete space is to enumerate all possible wordsets $s$ and find the top ones. However, due to huge enumeration combinations in discrete space, it is not feasible.

Here, we combine high-frequency screening and frequent itemset algorithm to solve the approximate optimal solution of Eq.(11). Our method is based on the prior knowledge that a category-specific wordset occurs with high frequency in this class, and the words included in the high-frequency wordset are also high-frequency. Therefore, we first use high-frequency screening to filter out low-frequency words, then use the frequent item set mining algorithm to mine high-frequency wordsets, and finally utilize Eq.(11) to evaluate the wordsets.

### 3.5.1 High-frequency Screening

Firstly, we need to build a high-frequency vocabulary $\Phi^c$ for each category $c$. To achieve this, we aggregate the texts predicted with class $c$ by the trained classifier, which is denoted as $\mathcal{X}^c$. Then, we count the frequency of each word $w_i$ in $\mathcal{X}^c$ as $TF(w_i)^c = \sum_{x_j \in \mathcal{X}^c} \mathbb{I}(w_i \in x_j)$. For each class $c$, we keep top $Z_1$ words with the highest $TF(w_i)^c$, which is denoted as $\Phi^c$.

According to the high-frequency vocabulary $\Phi^c$, we simplify the original texts by filtering low-frequency words. For each text $x_i$ of class $c$, we keep only those words that appear in the vocabulary $\Phi^c$, and the result is denoted as $\tilde{x}_i = \{w_i | w_i \in x_i \text{ and } w_i \in \Phi^c\}$. Through filtering operations, we can greatly reduce the number of words in each text, thus speeding up the mining process.

### 3.5.2 Frequent Wordsets Mining

According to the simplified texts $\tilde{x}_i$, we mine frequent itemsets in each class $c$. Treat each $\tilde{x}_i$ of class $c$ as a transaction, and we use the FP-Growth (Borgelt, 2005) algorithm to mine frequent itemsets that appears in as many $\tilde{x}_i$ as possible. We choose FP-Growth for its better time complexity. Then, $Z_2$ itemsets with the highest support will be output from FP-Growth for class $c$, which is denoted as $\widetilde{\mathcal{S}}^c = \{\tilde{s}_1^c, \tilde{s}_2^c, ..., \tilde{s}_{Z_2}^c\}$, where the support of a itemset refers to the probability of occurrence of that itemset in all $\tilde{x}_i$ of class $c$.

### 3.5.3 Information Bottleneck Ranking

With the results $\widetilde{\mathcal{S}}^c$ from FP-Growth, we score each $\tilde{s}_i^c \in \widetilde{\mathcal{S}}^c$ with Eq.(11), where the mutual information is calculated using the representations of the last layer of trained classifier. For each class $c$, we keep the top $T$ wordsets with the highest score and update them into the initialized wordsets. Then, with updated wordsets, we restart a new round of training. When the total number of iterations reaches $M$, the iteration stops.

## 4 Experiments

For experimental settings and hyperparameter, please refer to the Appendix.C.

### 4.1 Datasets

All our experimental settings and datasets follow a recent WTC benchmark(Wang et al., 2023b), which replicated WTC methods using standardized evaluation criteria, including seed-based and prompt-based approaches. Following the benchmark, we conducted experiments on 11 text datasets from diverse domains. and the input class names are also the same. More details can refer to Appendix.A.

### 4.2 Compared Methods

We compared seed-based and prompt-based methods following the benchmark(Wang et al., 2023b). Seed methods included LoT-Class (Meng et al., 2020b), X-Class (Wang et al., 2021), ClassKG (Zhang et al., 2021) and NPPrompt (Zhao et al., 2023a), while prompt methods included prompt baseline (Wang et al., 2023a), prompt+DCPMI (Holtzman et al., 2021) and prompt+ProtoCal (Han et al., 2023). Prompt methods use GPT-2, while seed methods use BERT (except NPPrompt-Roberta) as classifier. More details of compared methods can refer to Appendix.B.

### 4.3 Performance Comparison

The evaluation results are summarized in Tab.1. Our method achieved the highest average results, surpassing SOTA on most datasets regardless of whether BERT-base or BERT-large was used as the classifier. SetSync outperformed ClassKG by 9.0 points and the best prompt method (Prompt+DCPMI) by 24.70 points with BERT-base. With BERT-large, it exceeded previous SOTA (X-class) by 8.54 points on average. SetSync demonstrated strong performance on both short and long texts, highlighting its effectiveness and generalizability.

### 4.4 Ablation Study

We perform further module inspections, all experiments use BERT-base as the classifier.

### 4.4.1 Effect of Different Modules

We investigate the effectiveness of sync-denoising training and wordsets mining, and present the results in Fig. 4. We compared sync-denoising

| Method | Model | IMDB | Yelp-2 | Yelp-5 | AGNews | 20News | 20News-Fine | NYT-S | NYT-S-Fine | NYT | NYT-Loc | DBpedia | Average |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | PROMPT | | | | | | |
| Prompt | GPT2-small | 56.42 | 47.36 | 7.62 | 38.42 | 36.32 | 28.76 | 22.45 | 38.90 | 33.44 | 60.32 | 13.93 | 34.90 |
| | GPT2-medium | 35.80 | 33.57 | 25.87 | 69.36 | 55.16 | 46.03 | 54.08 | 46.14 | 24.92 | 79.00 | 24.52 | 44.95 |
| Prompt + DCPMI | GPT2-small | 70.13 | 65.34 | 23.01 | 72.67 | 61.64 | 37.45 | 73.93 | 63.19 | 55.20 | 70.40 | 51.10 | 58.55 |
| | GPT2-medium | 63.24 | 87.00 | 11.34 | 74.13 | 61.15 | 52.74 | 79.80 | 67.66 | 58.44 | 87.35 | 57.30 | 63.65 |
| Prompt + ProtoCal | GPT2-small | 70.35 | 65.89 | 23.77 | 72.66 | 58.62 | 36.77 | 53.69 | 29.82 | 55.15 | 65.80 | 51.97 | 53.14 |
| | GPT2-medium | 70.58 | 88.60 | 36.62 | 75.26 | 62.58 | 48.55 | 51.97 | 46.85 | 59.04 | 72.45 | 66.46 | 61.54 |
| | | | | | | | SEED | | | | | | |
| LoT-Class | BERT-base | 58.56 | 67.96 | 24.92 | 73.94 | 70.57 | 9.40 | 61.36 | 23.05 | 48.59 | 67.13 | 57.98 | 51.2 |
| | BERT-large | 81.03 | 77.03 | 25.17 | 68.25 | 65.71 | 45.51 | 44.00 | 37.11 | 43.08 | 80.55 | 58.04 | 56.86 |
| X-Class | BERT-base | 82.89 | 85.44 | 28.80 | 81.81 | 76.98 | 58.78 | 91.94 | 61.06 | 67.19 | 86.38 | 89.50 | 73.71 |
| | BERT-large | 82.05 | 90.39 | 31.02 | 85.91 | 77.52 | 59.98 | 87.53 | 68.40 | 68.73 | 85.77 | 87.91 | 75.02 |
| ClassKG | BERT-base | 88.08 | 92.21 | 32.33 | 88.10 | 81.72 | 52.29 | 84.12 | 49.59 | 60.79 | 92.81 | **94.75** | 74.25 |
| | BERT-large | 90.96 | 93.10 | 39.41 | 87.30 | 83.84 | 51.62 | 80.95 | 59.95 | 56.31 | 91.03 | 72.74 | 73.38 |
| NPPrompt | Roberta-base | 85.19 | 81.17 | 14.20 | 80.42 | 68.92 | 48.64 | 77.76 | 55.23 | 64.46 | 53.85 | 60.36 | 62.75 |
| | Roberta-large | 85.67 | 93.58 | 23.45 | 83.62 | 69.82 | 43.33 | 77.93 | 35.91 | 59.96 | 65.83 | 47.11 | 62.38 |
| | | | | | | | WORD SET | | | | | | |
| **SetSync (Ours)** | BERT-base | **90.18** | **93.31** | **45.94** | **89.76** | **83.40** | **68.99** | **92.23** | **89.79** | **78.67** | **93.83** | 89.67 | **83.25** |
| | BERT-large | **92.52** | **94.12** | **51.65** | **88.91** | **84.01** | **67.29** | **89.50** | **91.05** | **76.73** | **94.82** | **88.60** | **83.56** |

Table 1: Performance comparison. All methods take class names as input. Accuracy on the test set is reported.

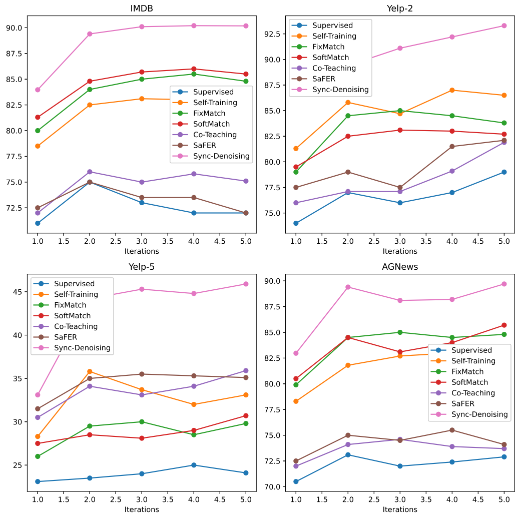


Figure 4: The classifier performance of each iteration when changing the training strategies.

with various training strategies, including fully-supervised, semi-supervised (self-training (Pseudo-Label, 2013), fixmatch (Sohn et al., 2020a), softmatch (Chen et al., 2023a)), and noisy-label learning (co-teaching (Han et al., 2018), SaFER (Qi et al., 2023)), reporting the performance of each round for each training strategy. From Fig. 4, we can see that: 1) Our sync-denoising training outerformed all other strategies. Fully-supervised training on pseudo-labeled data had the worst performance. Semi-supervised and noisy-label learning were slightly better than fully-supervised, but with limited improvement. 2) Classifier performance gradually improved with iterations in all strategies, which demonstrating the effectiveness of our word-sets mining algorithm. In addition,we observed that the $2^{nd}$ round typically showed the most significant improvement. This is mainly due to the use of mined wordsets to generate high-quality pseudo-labels, greatly improving the performance.

| Pseudo-Labeled | Unlabeled | IMDB | Yelp-2 | Yelp-5 | AGNews |
|---|---|---|---|---|---|
| $\max \mathbf{p}(x_i)$ | $\max \mathbf{p}(x_i)$ | 88.83 | 91.97 | 40.09 | 87.55 |
| $\max \mathbf{p}(x_i)$ | $l(x_i)$ | 87.89 | 91.89 | 41.24 | 86.20 |
| $l(x_i)$ | $l(x_i)$ | 89.36 | 92.10 | 42.98 | 88.65 |
| $l(x_i)$ | $\max \mathbf{p}(x_i)$ | **90.18** | **93.31** | **45.94** | **89.76** |

Table 2: Results of different noise estimation methods.

### 4.4.2 Different Noise Estimation

In sync-denoising training of classifier, we estimated pseudo-labeled data and unlabeled text's prediction noise using loss value and prediction confidence, respectively. We try to change it, with results shown in Tab. 2. We can see that using loss values to evaluate noise on pseudo-labeled data and using prediction confidence to evaluate unlabeled data achieved the best results. This is because for pseudo-label data, the loss value can be a good measure of the consistency between the model's prediction and the pseudo-label. For abnormal samples with incorrect labels, the consistency will be greatly reduced, so that its noise level can be eval-

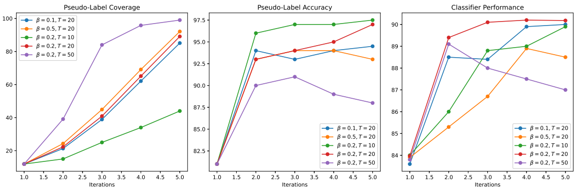


Figure 5: The influence of $\beta$ and $T$ of WIB.

| Pseudo-Labels Generation | IMDB | Yelp-2 | Yelp-5 | AGNews |
|---|---|---|---|---|
| Random Selection | 0.872 | 0.855 | 0.529 | 0.863 |
| Length Accumulation | 0.915 | 0.903 | 0.570 | 0.879 |
| Score Accumulation | 0.931 | **0.925** | 0.581 | 0.891 |
| Majority Voting | **0.934** | 0.919 | **0.595** | **0.892** |

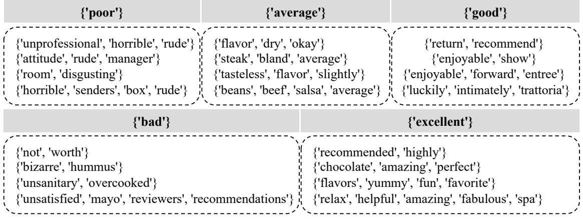
Table 3: Pseudo-label accuracy in the second round with different pseudo-label generation methods.

uated. For unlabeled samples, the loss value is a measure of the consistency between weak and strong augmentation. Since most of the strong augmentation are difficult samples, the results are likely to be wrong. Therefore, this consistency is not as reliable as the prediction confidence.

### 4.4.3 Ablation on WIB

Here, we perform experiments on $\beta$ and $T$ in wordsets information bottleneck, where $\beta$ is a lagrange multiplier that trade-off informativeness and compression, and $T$ is the number of reserved wordsets in each round. We report the accuracy and coverage of the pseudo-labels, and the performance of classifier over iterations in Fig. 5. We can see that: 1) A higher $T$ will bring higher pseudo-label coverage and also reduce pseudo-label accuracy. 2) A higher $\beta$ will lead to higher compression, thereby extracting higher coverage wordsets. Likewise, higher compression ratios reduce accuracy. 3) Our system is robust to these hyperparameters, and these parameter selections can achieve high performance.

### 4.4.4 Ablation on Pseudo-Label Generation



| {'poor'} | {'average'} | {'good'} |
|---|---|---|
| {'unprofessional', 'horrible', 'rude'}<br>{'attitude', 'rude', 'manager'}<br>{'room', 'disgusting'}<br>{'horrible', 'senders', 'box', 'rude'} | {'flavor', 'dry', 'okay'}<br>{'steak', 'bland', 'average'}<br>{'tasteless', 'flavor', 'slightly'}<br>{'beans', 'beef', 'salsa', 'average'} | {'return', 'recommend'}<br>{'enjoyable', 'show'}<br>{'enjoyable', 'forward', 'entree'}<br>{'luckily', 'intimately', 'trattoria'} |

| {'bad'} | {'excellent'} |
|---|---|
| {'not', 'worth'}<br>{'bizarre', 'hummus'}<br>{'unsanitary', 'overcooked'}<br>{'unsatisfied', 'mayo', 'reviewers', 'recommendations'} | {'recommended', 'highly'}<br>{'chocolate', 'amazing', 'perfect'}<br>{'flavors', 'yummy', 'fun', 'favorite'}<br>{'relax', 'helpful', 'amazing', 'fabulous', 'spa'} |

Figure 6: Wordsets example.The gray area represents the wordset initialized for the class, while the dashed lines indicate the wordsets updated after three iterations.

| Model | AG News | DBPedia | IMDB | Yelp-2 | Yelp-5 |
|---|---|---|---|---|---|
| GPT-3 | 83.4 | 82.5 | 88.8 | 92.6 | 42.9 |
| ChatGPT-3.5 | 83.8 | 92.0 | 92.7 | **97.2** | **73.8** |
| SetSync (BERT-base) | **88.5** | **90.1** | 90.9 | 93.3 | 45.9 |
| SetSync (BERT-large) | 87.3 | 89.3 | **92.3** | 94.1 | 51.6 |

Table 4: Performance comparison with ChatGPT.

We conduct further experiments on pseudo-label generation. When a text contains only one category of wordsets, its category is easy to determine. However, when a text contains multiple categories, we try to use different methods to determine its pseudo-labels, including random selection, score accumulation, length accumulation, majority voting. The length accumulation accumulates the total number of words in wordsets of each class. The score accumulation accumulates the scores of wordsets in each class, where the score is obtained from the previous round of WIB. Majority voting is formulated as Eq.(1). We report the second-round pseudo-labels accuracy in Tab.3. We can see that majority voting achieved relatively stable and good results in most cases, followed by score accumulation. Score accumulation may cause performance degradation in some cases due to the imbalance of scores between different categories.

## 4.5 Qualitative Analysis

Using the Yelp-5 dataset as an example, we initialized the wordset with class names and updated it over three iterations, as shown in Fig. 6. From the results, it is evident that our method employed a wordset to represent each category, allowing for a more accurate expression of nuanced sentiment compared to individual seed words. This explains our higher performance on the Yelp-5 dataset and enables a more granular and precise interpretation of sentiments, essential in effective dataset understanding and analysis.

## 4.6 Comparison with ChatGPT

We also conducted a comparative study with ChatGPT-3.5 and GPT-3. However, given the huge scale of the datasets, using ChatGPT-3.5 to predict all test sets of 11 datasets (about 60K texts) would incur a high cost. Therefore, we chose to conduct experiments on a part of the datasets. More experimental details can refer to Appendix.D, and the prompts for ChatGPT and GPT-3 can be found in Tab.7. The results are shown in Tab. 4. From the results, we can see that ChatGPT-3.5 can achieve the highest results on some datasets. Meanwhile, our method can surpass GPT-3 on all datasets and beat ChatGPT-3.5 on some datasets.

## 5 Conclusion

In this paper, we introduce SetSync, which utilize wordsets to represent categories and generate pseudo-labels. To achieve this, we propose a class-relevant wordsets mining algorithm with wordsets information bottleneck. Moreover, we revisit the classifier training in WTC and propose a new training strategy, called sync-denoising, which jointly optimize unlabeled and noisy labeled samples with a unified denoising framework. Extensive experiment results on 11 datasets shows that SetSync substantially outperforms all existing methods.

## References

Paul Albert, Diego Ortego, Eric Arazo, Noel O'Connor, and Kevin McGuinness. 2021. Addressing out-of-distribution label noise in webly-labelled data.

Sonia Badene, Kate Thompson, Jean-Pierre Lorré, and Nicholas Asher. 2019. Data programming for learning discourse structure. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 640–645, Florence, Italy. Association for Computational Linguistics.

Farhang Bayat and Shuangqing Wei. 2019. Information bottleneck problem revisited. In *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 40–47. IEEE.

Christian Borgelt. 2005. An implementation of the fp-growth algorithm. In *Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations*, pages 1–5.

H Chen, R Tao, Yue Fan, Y Wang, M Savvides, J Wang, B Raj, X Xie, and Bernt Schiele. 2023a. Softmatch: Addressing the quantity-quality tradeoff in semi-supervised learning. In *Eleventh International Conference on Learning Representations*. OpenReview. net.

Hao Chen, Ran Tao, Yue Fan, Yidong Wang, Jindong Wang, Bernt Schiele, Xing Xie, Bhiksha Raj, and Marios Savvides. 2023b. Softmatch: Addressing the quantity-quality trade-off in semi-supervised learning.

Evgeniy Gabrilovich, Shaul Markovitch, et al. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *International Joint Conference on Artificial Intelligence*, volume 7, pages 1606–1611.

Bo Han, Gang Niu, Xingrui Yu, Quanming Yao, Miao Xu, Ivor Tsang, and Masashi Sugiyama. 2020. Sigua: Forgetting may make learning with noisy labels more robust.

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.

Zhixiong Han, Yaru Hao, Li Dong, Yutao Sun, and Furu Wei. 2022a. Prototypical calibration for few-shot learning of language models. *arXiv preprint arXiv:2205.10183*.

Zhixiong Han, Yaru Hao, Li Dong, Yutao Sun, and Furu Wei. 2022b. Prototypical calibration for few-shot learning of language models.

Zhixiong Han, Yaru Hao, Li Dong, Yutao Sun, and Furu Wei. 2023. Prototypical calibration for few-shot learning of language models. In *The Eleventh International Conference on Learning Representations*.

Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. 2021. Surface form competition: Why the highest probability answer isn't always right. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7038–7051, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. 2022. Surface form competition: Why the highest probability answer isn't always right.

Youngwook Kim, Jae Myung Kim, Zeynep Akata, and Jungwoo Lee. 2022. Large loss matters in weakly supervised multi-label classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14156–14165.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on International Conference on Machine Learning*, ICML'95, page 331–339, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Junnan Li, Richard Socher, and Steven C. H. Hoi. 2020. Dividemix: Learning with noisy labels as semi-supervised learning.

Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. 2020. Early-learning regularization prevents memorization of noisy labels.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Andrew L. 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*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Dheeraj Mekala and Jingbo Shang. 2020. Contextualized weak supervision for text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 323–333, Online. Association for Computational Linguistics.

Yu Meng, Jiaxin Huang, Guangyuan Wang, Zihan Wang, Chao Zhang, Yu Zhang, and Jiawei Han. 2020a. Discriminative topic mining via category-name guided text embedding. In *Proceedings of The Web Conference 2020*, WWW '20. ACM.

Yu Meng, Yunyi Zhang, Jiaxin Huang, Chenyan Xiong, Heng Ji, Chao Zhang, and Jiawei Han. 2020b. Text classification using label names only: A language model self-training approach. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9006–9017, Online. Association for Computational Linguistics.

Dong-Hyun Lee Pseudo-Label. 2013. The simple and efficient semi-supervised learning method for deep neural networks. In *ICML 2013 Workshop: Challenges in Representation Learning*, pages 1–6.

Zhenting Qi, Xiaoyu Tan, Chao Qu, Yinghui Xu, and Yuan Qi. 2023. SaFER: A robust and efficient framework for fine-tuning BERT-based classifier with noisy labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 390–403, Toronto, Canada. Association for Computational Linguistics.

Kai Shu, Subhabrata Mukherjee, Guoqing Zheng, Ahmed Hassan Awadallah, Milad Shokouhi, and Susan Dumais. 2020. Learning with weak supervision for email intent detection. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM.

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. 2020a. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *Advances in Neural Information Processing Systems*, volume 33, pages 596–608. Curran Associates, Inc.

Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. 2020b. Fixmatch: Simplifying semi-supervised learning with consistency and confidence.

Zihan Wang, Dheeraj Mekala, and Jingbo Shang. 2021. X-class: Text classification with extremely weak supervision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3043–3053, Online. Association for Computational Linguistics.

Zihan Wang, Tianle Wang, Dheeraj Mekala, and Jingbo Shang. 2023a. A benchmark on extremely weakly supervised text classification: Reconcile seed matching and prompting approaches. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 3944–3962, Toronto, Canada. Association for Computational Linguistics.

Zihan Wang, Tianle Wang, Dheeraj Mekala, and Jingbo Shang. 2023b. A benchmark on extremely weakly supervised text classification: Reconcile seed matching and prompting approaches.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3914–3923, Hong Kong, China. Association for Computational Linguistics.

Lu Zhang, Jiandong Ding, Yi Xu, Yingyao Liu, and Shuigeng Zhou. 2021. Weakly-supervised text classification based on keyword graph.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2016. Character-level convolutional networks for text classification.

Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models.

Xuandong Zhao, Siqi Ouyang, Zhiguo Yu, Ming Wu, and Lei Li. 2023a. Pre-trained language models can be fully zero-shot learners. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15590–15606, Toronto, Canada. Association for Computational Linguistics.

Xuandong Zhao, Siqi Ouyang, Zhiguo Yu, Ming Wu, and Lei Li. 2023b. Pre-trained language models can be fully zero-shot learners. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15590–15606, Toronto, Canada. Association for Computational Linguistics.

# A Details of Datasets

These datasets can be divided into the following four categories:

1) Sentiment Analysis:

- IMDB(Maas et al., 2011): Contains movie reviews from the IMDB website.it's a binary classification dataset (positive or negative)

- Yelp-2 and Yelp-5(Zhang et al., 2016): These datasets are derived from Yelp reviews. Yelp-2 is a binary classification dataset (positive or negative reviews), while Yelp-5 has a finer classification with five classes, rating reviews from one to five stars.

2) Topic Classification:

- AGNews(Zhang et al., 2016): Classifies news articles into four main topics: World, Sports, Business, and Science/Technology.

- 20News and 20News-Fine(Lang, 1995): Topic classification of newsgroup articles, covering a range of topics, with the "Fine" version offering more detailed classification.

3) Entity Recognition and Classification:

- DBpedia(Zhang et al., 2016): Entities extracted from Wikipedia, categorized into 14 different classes, used for multi-class classification and entity recognition.

4) News Text and Fine-grained Analysis:

- NYT, NYT-Fine, NYT-Topics and NYT-Loc(Meng et al., 2020a): Articles from The New York Times, used for various text analysis tasks. NYT for broader classification tasks, NYT-Fine for detailed analysis of sub-topics, NYT-Topics for thematic categorization of news into specific areas like politics or sports, and NYT-Loc for geographic-focused analysis.

# B Details of Compared Methods

In this paper, the Compared Methods align with the published benchmark paper(Wang et al., 2023b) and include a comparison between the Seed and Prompt methods.

## B.1 Seed Approach

The Seed methods, except for NPPrompt, are all based on Bert (base and large).

- LotClass(Meng et al., 2020b) expand related words using masked language models and matches texts by fine-tuning.

- Xclass(Wang et al., 2021) retrieves related words by searching for words with similar representations and matches texts with clustering-enhanced similarity.

- ClassKG(Zhang et al., 2021) Transform the dependency among related words into a keyword graph annotation task.

- NPPrompt(Zhao et al., 2021) based on Roberta, obtains related words by extracting embedding similarity from pre-trained language models, using them as prompts for a generative language model to make predictions, which are then aggregated for the final match.

## B.2 Prompt Approach

All prompt methods are based on GPT-2 (small and medium).

- DC-PMI(Holtzman et al., 2022) uses empty prompts to gauge a language model's likelihood for predicting labels, refining predictions for each text.

- ProtoCal(Han et al., 2022b) works with unlabeled data, obtaining predictive likelihoods and clustering them to improve category predictions.

# C Experimental Setting

The model training and evaluation is performed on NVIDIA RTX 4090. The classifier is implemented with 'bert-base-uncased' and 'bert-large-uncased', which follows the setting in benchmark. The classifier is optimized with AdamW (Loshchilov and Hutter, 2019), and the learning rate is 2e-6. The batch size $B$ is set to 64. The number of wordsets mined each round $T$ is set to 20. $\lambda_{\max}$ in Eq.(6) is set to 1. Momentum coefficient $\epsilon$ in Eq.(8) is set to 0.001. The initial value of $\mu_t$ and $\sigma_t$ for labeled and unlabeled data is set to $\frac{1}{C}$ and 1.0. The trade-off coefficient $\beta$ in Eq.(11) is set to 0.2. The parameter $Z_1$ to limit the size of $\Phi^c$ is set to 2000, and $Z_2$ to

Table 5: Dataset statistics.

| Name | Domain | # Classes | ‖Unlabelled‖ | ‖Eval‖ | Imbalance |
|---|---|---|---|---|---|
| IMDB | Reviews/Sentiment | 2 | 5000 | 5000 | 1.0 |
| Yelp-2 | Reviews/Sentiment | 2 | 5600 | 3800 | 1.1 |
| Yelp-5 | Reviews/Sentiment | 5 | 6500 | 5000 | 1.1 |
| AGNews | News/Topic | 4 | 6000 | 7600 | 1.0 |
| 20News | News/Topic | 5 | 6254 | 5362 | 1.9 |
| 20News-Fine | News/Topic | 17 | 5589 | 4792 | 1.3 |
| NYT-S | News/Topic | 9 | 4578 | 3925 | 17.1 |
| NYT-S-Fine | News/Topic | 26 | 4034 | 3459 | 96.3 |
| NYT | News/Topic | 5 | 5119 | 6400 | 30.7 |
| NYT-Loc | News/Location | 10 | 5119 | 6400 | 17.1 |
| DBpedia | Wikipedia/Ontology | 14 | 5600 | 7000 | 1.3 |

Table 6: Initial class name of each dataset.

| Dataset | Seed Words/Initial class name |
|---|---|
| IMDB | positive; negative |
| Yelp-2 | positive; negative |
| Yelp-5 | poor; bad; average; good; excellent |
| AGNews | politics; sports; business; technology |
| 20News | computer; sports; science; politics; religion |
| 20News-Fine | atheism; graphics; Microsoft; IBM; Mac; motif; autos; motorcycles; baseball; hockey; encryption; electronics; medicine; space; Christian; guns; Arab |
| NYT | politics; art; business; science; sport |
| NYT-S-Fine | budget; gun; laws; gay; energy; environment; immigration; military; cosmos; insurance; stocks; bank; abortion; music; baseball; economy; television; golf; tennis; hockey; football; dance; movies; soccer; surveillance; basketball |
| NYT-S | business; politics; sports; health; education; estate; arts; science; technology |
| NYT-Loc | America; Iraq; Japan; China; Britain; Russia; Germany; Canada; France; Italy |
| DBpedia | company; education; artist; athlete; politician; transportation; place; nature; village; species; plant; album; movie; book |

limit the output number from FP-Growth is set to 200. The wordsets change threshold $\Delta$ is set to 0.1. More details can refer to our code.

## D Details of Compared ChatGPT

As the experiments of ChatGPT-3.5 on AG News, DBPedia, and IMDB have already been reported in a recent WTC work (Zhao et al., 2023b), we directly reference its results. To be fair, we reran our method on the test set they provided. Additionally, we have included additional experimental results for Yelp-2 and Yelp-5, where the prompts for ChatGPT is shown in Tab.7 and the experiments is based on the June 2023 version of ChatGPT.

## E Limitation

The running time of our method may be longer than the seed methods, especially the mining part of frequent itemsets. However, through more stringent high-frequency filtering, the mining of frequent itemsets can be significantly accelerated, but the accuracy may be slightly lost. Our method requires multiple rounds of iteration, which also means that it will consume more time than non-iterative algorithms. However, we need to state that this running time is negligible compared to the time required for large-scale annotation.

| **Prompts for GPT-3 and ChatGPT-3.5** |
|---|
| *AG News* :<br>*[Descriptions]* Definition: In this task, you are given a sentence.<br>Your job is to classify the following sentence into one of the four different categories.<br>The categories are: “politics”, “sports”, “business”, and “technology”. Input: `[x]`. Output: |
| *DBPedia*:<br>*[Descriptions]* Definition: In this task, you are given a sentence.<br>Your job is to classify the following sentence into one of the fourteen different categories.<br>The categories are: “company”, “school”, “artist”, “athlete”, “politics”,<br>“transportation”, “building”, “mountain”, “village”, “animal”, “plant”, “album”, “film”, and “book”. Input: `[x]`. Output: |
| *IMDB*:<br>*[Descriptions]* Definition: In this task, you are given a sentence.<br>Your job is to classify the following sentence into one of the two categories. The categories are: “bad" and “good". Input: `[x]`. Output: |
| *Yelp-2*:<br>*[Descriptions]* Definition: In this task, you are given a sentence.<br>Your job is to classify the following sentence into one of the two categories.<br>The categories are: “negative" and “positive". Input: `[x]`. Output: |
| *Yelp-5*:<br>*[Descriptions]* Definition: In this task, you are given a sentence.<br>Your job is to classify the following sentence into one of the five categories.<br>The categories are: “poor" , “bad", “average", “good", “excellent". Input: `[x]`. Output: |

Table 7: Prompts for GPT-3 and ChatGPT-3.5.

| **Descriptions** |
|---|
| *AG News*:<br>The politics category is related to politics, government, and law.<br>The sports category is related to sports, competition, and athletics.<br>The business category is related to business, portfolio, economics, and money.<br>The technology category is related to technology, software, system, and science. |
| *DBPedia*:<br>The company category is related to company, corporation, enterprise, brand, and business.<br>The school category is related to school, academy, university, and college.<br>The artist category is related to artist, art, painter, musician, singer, and creative.<br>The athlete category is related to athletes, sports, Olympic, and gym.<br>The politics category is related to politics, government, and law.<br>The transportation category is related to transportation, transport, vehicle, and traffic.<br>The building category is related to buildings, construction, and structure.<br>The mountain category is related to river, lake, bay, and mountain.<br>The village category is related to village, town, and rural.<br>The animal category is related to animal, wildlife, and nature.<br>The plant category is related to plant, shrub, tree, and forest.<br>The album category is related to album, lyrics, cd, and song.<br>The film category is related to film, movie, cinema, and video.<br>The book category is related to book, novel, and publication. |
| *IMDB*:<br>The bad category is related to negative and bad reviews.<br>The good category is related to positive and good reviews. |
| *Yelp-2*:<br>The negative category is related to negative and bad reviews.<br>The positive category is related to positive and good reviews. |
| *Yelp-5*:<br>The poor category is related to extremely negative and terrible reviews.<br>The bad category is related to negative and bad reviews.<br>The average category is related to moderate or average reviews.<br>The good category is related to positive and good reviews.<br>The excellent category is related to highly positive and exceptional reviews. |

Table 8: Descriptions in Tab. 7