Unsupervised Label Refinement Improves Dataless Text Classification

Zewei Chu[1] **Karl Stratos**[2] **Kevin Gimpel**[3]
[1]The University of Chicago, 5730 S Ellis Ave, Chicago, IL 60637, USA
[2]Rutgers University, 110 Frelinghuysen Road, Piscataway, NJ 08854, USA
[3]Toyota Technological Institute at Chicago, 6045 S Kenwood Ave, Chicago, IL 60637, USA
zeweichu@uchicago.edu, karlstratos@gmail.com, kgimpel@ttic.edu

Abstract

Dataless text classification is capable of classifying documents into previously unseen labels by assigning a score to any document paired with a label description. While promising, it crucially relies on accurate descriptions of the label set for each downstream task. This reliance causes dataless classifiers to be highly sensitive to the choice of label descriptions and hinders the broader application of dataless classification in practice. In this paper, we ask the following question: how can we improve dataless text classification using the inputs of the downstream task dataset? Our primary solution is a clustering based approach. Given a dataless classifier, our approach refines its set of predictions using k-means clustering. We demonstrate the broad applicability of our approach by improving the performance of two widely used classifier architectures, one that encodes text-category pairs with two independent encoders and one with a single joint encoder. Experiments show that our approach consistently improves dataless classification across different datasets and makes the classifier more robust to the choice of label descriptions.[1]

1 Introduction

Dataless text classification aims at classifying text into categories without using any annotated training data from the task of interest. Prior work (Chang et al., 2008; Song and Roth, 2014) has shown that with effective ways to represent texts and labels, dataless classifiers can perform text classification on unbounded label sets if suitable descriptions of the labels are provided.

There have been many previous efforts in dataless or zero-shot text classification (Dauphin et al., 2013; Nam et al., 2016; Li et al., 2016; Ma et al., 2016; Shu et al., 2017; Fei and Liu, 2016; Zhang et al., 2019; Yogatama et al., 2017; Mullenbach et al., 2018; Rios and Kavuluru, 2018; Meng et al., 2019). Several settings have been considered across this prior work, and some have used slightly different definitions of dataless classifiers. In this paper, we use the term "dataless text classification" to refer to methods that: (1) can assign scores to any document-category pair, and (2) do not require any annotated training data from downstream tasks. A dataless classifier can therefore be immediately adapted to a particular label set in a downstream task dataset by scoring each possible label for a document and returning the label with the highest score. Dataless classifiers are typically built from large-scale freely available text resources such as Wikipedia (Chang et al., 2008; Yin et al., 2019).

A well known problem with dataless classifiers is that the choice of label names has a significant impact on performance (Chang et al., 2008). As dataless classifiers rely purely on the label descriptions in a downstream task, there is typically no tailoring or fine-tuning of the classifier for a given dataset. A poor choice of label descriptions could jeopardize the performance of dataless classifiers on a particular text classification task, so prior work has addressed this with modifications to label descriptions. Chang et al. (2008) manually expanded label names for the 20 newsgroups dataset and Yin et al. (2019) expanded labels using WordNet.

To illustrate the problem, Table 1 shows various choices of label names when applying a dataless classifier to the 4-class AG News dataset. When we change the descriptions of the four labels, performance of our dataless text classifier[2] changes drastically. The broader application of dataless text classifiers is hindered by their fragility caused by the choice of label descriptions. It is unclear how

[1]Code and data available at https://github.com/ZeweiChu/ULR.

[2]A RoBERTa dual encoder architecture (Section 4).

choice of label names	**world** **sports** **business** **science technology**	international health finance technology	world politics sports business and finance science and technology	world news health business science and technology	world health and sports commerce science technology
dataless	69.9	55.9	71.6	49.9	55.0
dataless + ULR	70.7	78.3	78.4	70.2	70.9

Table 1: Accuracy (%) of a RoBERTa dual encoder dataless classifier (details in Section 4) on AG News with different choices of label names. The original label set (boldfaced) is "world", "sports", "business", and "sci/tech". The "dataless + ULR" row shows accuracies after applying unsupervised label refinement (details in Section 3).

practitioners should choose label descriptions for practical use.

In this paper, we ask the following question: how can we improve dataless text classification provided the *unlabeled* set of input texts for the downstream task in addition to its label descriptions? Our approach, which we refer to as unsupervised label refinement (ULR), is based on k-means clustering. We develop variations of our approach so that it can be applied to different styles of dataless text classifiers to improve their performance. Table 1 shows results when applying ULR to our dataless text classifier. In all cases, accuracies improve after applying ULR, with larger gains when using weaker label descriptions.

To summarize, our contributions in this paper are as follows:

- We propose unsupervised label refinement (ULR), a k-means clustering based approach to improve dataless classifiers.
- We develop variations of ULR that can be applied to different model architectures of dataless classifiers. Experiments on dual encoder and single encoder architectures show that ULR almost always improves performance.
- Experiments show that ULR improves robustness of dataless classification against choices of label names, making dataless classifiers more practically useful.

2 Dataless Text Classification

Dataless text classification (Chang et al., 2008; Chen et al., 2015; Song and Roth, 2014; Yin et al., 2019) aims at building a single, universal text classifier that can be applied to any text classification task with a given set of label descriptions. Dataless classifiers can be used on an unbounded set of categories. There is typically no tailoring or fine-tuning of the classifier for a dataset other than through specifying the label descriptions.

Since annotated data in the target task is not available for training, the choice of label descriptions plays a critical role in the performance of dataless classifiers (Chang et al., 2008; Yin et al., 2019). With a dataless classifier, a score is produced for each text-category pair, indicating their semantic relatedness. Text classification then becomes a ranking problem, i.e., picking the category that has the highest semantic relatedness with the text.

Several researchers have used Explicit Semantic Analysis (ESA) (Gabrilovich and Markovitch, 2007) as text representations in dataless text classification (Chang et al., 2008; Song and Roth, 2014; Wang et al., 2009). Both label descriptions and text are encoded into ESA vectors. Cosine similarity is used to compute scores between text and categories. Yin et al. (2019) directly compute text-category relatedness with a single BERT (Devlin et al., 2019) model. Chang et al. (2008) and Yin et al. (2019) exemplify two typical modeling choices for dataless classifiers, namely dual encoder and single encoder architectures, respectively. We will introduce them briefly and consider both types in our experiments.

Dual encoder model. With the dual encoder model, the category and text are fed into the encoder separately, each producing a vector representation. The text and category encoders could have shared or independent parameters. In our experiments, we always share parameters, i.e., we use the same encoder for both the categories and texts. A distance function takes both the category and text vectors and produces a scalar value. In our experiments, this scoring function can be either cosine distance or Euclidean (L2) distance.

Single encoder model. With a single encoder model, the category is combined with the text as a single sequence and fed into an encoder. The output of the encoder is a single vector that contains

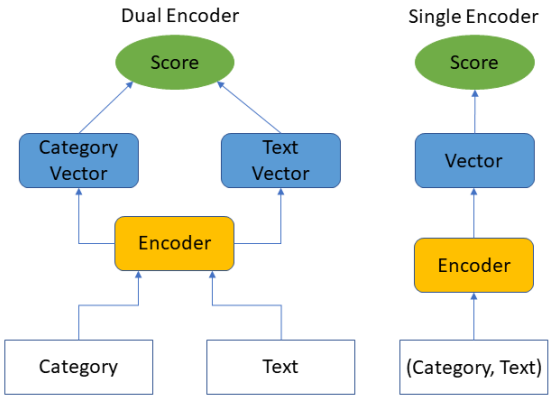

Figure 1: Dual and single encoder architectures.

the information from both the category and the text. This vector can pass through a linear layer and produce a score for this particular document-category pair. Figure 1 demonstrates the architectures of typical dual and single encoder models for text classification.

Algorithm 1: ULR for dual encoder architectures

Data: documents $\mathcal{T}$, categories $\mathcal{C}$, encoder enc, scoring function score
initialize the centroids $r_c = \text{enc}(c)\ \forall c \in \mathcal{C}$;
while *not converged* **do**
  for $t \in \mathcal{T}$ **do**
    $s_{tc} = \text{score}(\text{enc}(t), r_c)\ \forall c \in \mathcal{C}$;
    $\text{pred}_t = \text{argmin}_c s_{tc}$;
  end
  $r_c = (\frac{\sum_{t:\text{pred}_t=c} \text{enc}(t)}{\text{count}(\{t:\text{pred}_t=c\})} + \text{enc}(c))/2\ \forall c \in \mathcal{C}$;
  $\text{objective} = \sum_t s_{t\text{pred}_t}$;
end
Result: s_{tc}, objective, pred_t, and r_c of all iterations

3 Unsupervised Label Refinement (ULR)

In this section, we introduce unsupervised label refinement (ULR). ULR uses the components of a dataless classifier and refines representations of labels with a modified k-means clustering algorithm. While dataless text classifiers are designed to handle an unbounded set of categories, they are used and evaluated on a particular set of documents with a set of labels. The idea of our approach is to leverage the assumption that the documents in a text classification dataset are separable according to the accompanying set of labels. That is, given a strong document encoder, the documents should be separable by label in the encoded space. This assumption is similarly made when performing clustering for unsupervised document classification (Liang and Klein, 2009).

We use the set of unlabeled input texts to refine the predictions of our dataless classifiers via clustering. To better inform the algorithm, we initialize the clusters by using our dataless classifiers run on the provided label set for each task. The algorithm takes on different forms for the dual and single encoder models. Details are provided below.

3.1 ULR for Dual Encoder Architectures

In the setting of a dual encoder model, we propose to perform k-means clustering among text representations, i.e., of vectors produced by the text encoder. The assumption is that texts falling under the same category will be close in the semantic space. We want to adapt our dataless classifier's predictions based on the natural clustering structure in the encodings of the texts.

We show the ULR algorithm for dual encoder architectures in Algorithm 1. We use one encoder enc to encode texts and categories. We link centroids to categories and initialize the centroids using the encodings of the corresponding categories. The algorithm converges when no data point (text representation) updates its cluster assignment, i.e., the centroids stop updating. In our experiments, we run a maximum of 100 iterations. We perform model selection ("early stopping") based on the minimum value of objective among iterations.

Our k-means algorithm differs from standard k-means as our updated centroids are interpolated with the initial category embeddings. In standard k-means, the centroids are typically initialized randomly. In our case, since we link centroids to categories and use our encoder to provide initial centroids, we want to leverage the information in our category embeddings across clustering iterations. Therefore, we average the "new centroid" with the original category vector, which serves as a kind of regularization. While this update is heuristically motivated, we can reverse-engineer the clustering objective it approximately optimizes (with

Euclidean distance as score):

$$\min_{\{r_c\}_{c\in\mathcal{C}}} \min_{\{a_t\}_{t\in\mathcal{T}}} \sum_{t\in\mathcal{T}} ||\text{enc}(t) - r_{a_t}||^2 + \sum_{c\in\mathcal{C}} |c|\,||\text{enc}(c) - r_c||^2 \quad (1)$$

where $a_t \in \mathcal{C}$ is the category assignment of document $t \in \mathcal{T}$ and $|c|$ is the size of cluster/category c. See Appendix A for details. In preliminary experiments we found this modification to stabilize performance so we use it in our experiments reported below.

3.2 ULR for Single Encoder Architectures

In our single encoder architecture, a score is produced for each document-category pair indicating its relatedness. For each document, after exponentiating and normalizing the score over all categories, we obtain a distribution indicating the probability of the document belonging to each category. For text t and category c, we have a probability p_{tc}, where $\sum_c p_{tc} = 1$.

A straightforward way of classifying each document is to pick the category of which it has the highest probability score, i.e., $\text{argmax}_c\, p_{tc}$. Another way of interpreting this classification rule is to define $|\mathcal{C}|$ centroid vectors, each being an identity distribution $\text{one–hot}(c)$,[3] and pick the category having the minimum Jensen-Shannon Divergence (Lin, 1991) with the document probability vector, i.e., $\text{argmin}_c\, \text{JS}(p_t, \text{one–hot}(c))$, where JS is the Jensen-Shannon Divergence between two distributions. Clustering probability distributions based on KL divergence is a well-known special case of clustering with Bregman divergences (Banerjee et al., 2005).[4] While JS does not admit such a well-defined objective to our knowledge, we find that it works well in practice.

It is natural to represent each category as a distribution by a one-hot vector. However, in a real text classification problem, the semantics of a category is affected by how the annotators view it. For instance, a news document could relate to both "business" and "science & technology", though it will only have a single annotated category in the downstream task dataset. With this intuition, we propose to represent each category by a soft distribution over all the categories, but not necessarily a one-hot vector.

Algorithm 2 describes our k-means clustering approach applied on the single encoder model. In this algorithm our predicted categories will be the clustering assignments of the last iteration. Unlike with the dual encoder model, we do not do early stopping. We also do not use interpolated centroids as one-hot vectors may not necessarily be good category embeddings in this setting.

Algorithm 2: ULR for single encoder architectures

Data: documents $\mathcal{T}$, categories $\mathcal{C}$, scoring function score, function to compute JS divergence JS
initialize the centroids as $r_c = \text{one–hot}(c)\ \forall c \in C$;
compute each document's probability distribution over categories as $[p_t]_c \propto \exp(\text{score}(t, c))\ \forall t \in \mathcal{T}, c \in \mathcal{C}$;
while *not converged* **do**
  for $t \in \mathcal{T}$ **do**
    $s_{tc} = \text{JS}(p_t, r_c)\ \forall c \in \mathcal{C}$;
    $\text{pred}_t = \text{argmin}_c\, s_{tc}$;
  end
  $r_c = \frac{\sum_{t:\text{pred}_t=c} p_t}{\text{count}(\{t:\text{pred}_t=c\})}\ \forall c \in \mathcal{C}$;
end
Result: pred_t of the last iteration

4 Experimental Setup

We now introduce the datasets we used for evaluation and the dual encoder and single encoder dataless classifiers we used to run ULR experiments.

4.1 Evaluation

We use four text classification datasets spanning different domains for evaluation. They are: AG News[5] (AG), which uses 4 classes and covers the newswire domain; DBpedia (DBP; Lehmann et al., 2015), which has 14 classes and is from the domain of encyclopedias; Yahoo (Zhang et al., 2015), which has 10 classes and addresses categorizing

[3] Here we abuse the notation of $\text{one–hot}(c)$ to represent a one-hot vector that has a "1" at the index of category $c \in \mathcal{C}$.

[4] Specifically, if we replace $\text{JS}(p_t, r_c)$ with $D_{\text{KL}}(p_t||r_c)$ in Algorithm 2, it is an alternating minimization of the following clustering objective:

$$\min_{\{r_c\}_{c\in\mathcal{C}}} \min_{\{a_t\}_{t\in\mathcal{T}}} \sum_{t\in\mathcal{T}} D_{\text{KL}}(p_t||r_{a_t})$$

[5] https://www.di.unipi.it/~gulli/AG_corpus_of_news_articles.html

questions in online question fora; and 20 newsgroups (20NG; Lang, 1995), with 20 classes which are types of newsgroups.

We do not use any data or labels from the training set in our main experiments, but only rely on label descriptions. We use the official label names from these datasets, and only expand them if the original label name is provided as abbreviations such as "sci_tech". The exact label names we used are in the appendix.

4.2 Dataless Classifiers

We experiment with multiple dataless text classifiers that vary in terms of their complexity. Our simplest classifier uses an encoder that averages pretrained GloVe (Pennington et al., 2014) word embeddings. We also fine-tune a RoBERTa model (Liu et al., 2019) in both single and dual encoder settings, using RoBERTa-base (110M parameters). We do not run experiments with traditional dataless classifiers such as ESA (Chang et al., 2008; Gabrilovich and Markovitch, 2007) as ESA vectors are of extremely high dimension, making it computationally difficult to apply ULR. GloVe and RoBERTa produce lower dimensional vectors that are more computationally amenable to refinement. Also, we experimented with ESA and found that it does not perform as well as our RoBERTa based models (where both are tested without ULR).[6] Next we describe the details of the three dataless classifiers that we use in our experiments.

GloVe dual encoder. We use GloVe (Pennington et al., 2014) in the dual encoder setting, simply averaging word vectors to represent both the categories and the documents. We use the 300 dimensional GloVe vectors[7] trained on Common Crawl. We experiment with two distance functions when using GloVe: cosine and L2.

RoBERTa dual encoder. The category c and text t are fed separately to RoBERTa using the formatting "[CLS] c [SEP]" and "[CLS] t [SEP]". We use the average of the final-layer hidden states produced by RoBERTa as category and text vectors. For the scoring function, which computes a scalar from the category and text vectors, we consider dot product[8] and L2 distance.

RoBERTa single encoder. The category c and text t are combined in the form "[CLS] c [SEP] t [SEP]", tokenized, and encoded using RoBERTa. We truncate t to ensure the category-document pair is within 128 tokens. The vector representation of the "[CLS]" token (after a linear transformation and non-linear activation) is then passed to a linear layer to produce a score.

4.3 Fine-tuning RoBERTa Models

We use the NatCat dataset (Chu et al., 2020) to fine-tune the RoBERTa models. NatCat comprises document-category pairs from three resources: Wikipedia, Stack Exchange, and Reddit. Each NatCat document comes with positive and negative categories. A positive category describes the document, and a negative category is randomly sampled and is irrelevant to the document. The RoBERTa models are fine-tuned as binary classifiers to indicate whether a category is positive for a document. We use the Hugging Face framework (Wolf et al., 2019) to fine-tune all RoBERTa models, using 300k instances from NatCat.[9]

Fine-tuning RoBERTa dual encoder. While many other combinations of score and loss function could be considered, we report results with two particular combinations: dot product paired with binary cross entropy and L2 distance paired with a contrastive hinge loss. When dot product is used, RoBERTa is fine-tuned to minimize binary cross entropy between $\text{score}(\text{enc}(c), \text{enc}(t))$ and a binary label $y \in \{0, 1\}$.

When using L2 distance, we fine-tune RoBERTa to minimize a contrastive hinge loss:

$$\text{loss} = \max(x_p + \gamma - x_n, 0)$$

where $x_p = \text{score}(\text{enc}(c_p(t)), \text{enc}(t))$, $x_n = \text{score}(\text{enc}(c_n(t)), \text{enc}(t))$, score is the squared L2 distance, $c_p(t)$ is a positive category for text t, $c_n(t)$ is a negative category, and γ is a parameter indicating the margin. This loss aims to make negative category-text pairs have higher squared L2 distance than negative pairs by the margin.

Fine-tuning RoBERTa single encoder. When used as a single encoder, RoBERTa is fine-tuned

[6] As a comparison to the results of the RoBERTa dual encoder in Table 2, ESA accuracies (%) are 71.2 for AG, 62.5 for DBP, 29.7 for Yahoo, and 25.1 for 20NG.

[7] http://nlp.stanford.edu/data/glove.840B.300d.zip

[8] Practically we find the model is easier to train with dot product compared to cosine similarity.

[9] The finetuning hyperparameters are in the appendix.

		AG	DBP	Yahoo	20NG	avg
GloVe						
cosine	baseline	66.1	43.5	**29.4**	29.8	42.2
	+ ULR	**78.3**	**46.4**	27.9	**30.8**	**45.9**
L2	baseline	40.5	15.7	14.5	19.5	22.6
	+ ULR	**58.7**	**35.7**	**23.3**	**25.9**	**35.9**
RoBERTa						
cosine	baseline	**74.0**	84.6	52.3	36.4	61.8
	+ ULR	73.7	**93.3**	**54.3**	**40.3**	**65.4**
L2	baseline	69.9	78.8	55.7	**37.8**	60.6
	+ ULR	**70.7**	**90.5**	**61.3**	37.4	**65.0**

Table 2: Accuracies (%) when applying ULR to the GloVe and RoBERTa dual encoder architecture ("baseline"), for two different choices of distance function (cosine and L2).

	AG	DBP	Yahoo	20NG	avg
baseline	72.6	81.8	59.3	36.0	62.4
+ ULR	**75.1**	**88.6**	**60.0**	**36.5**	**65.1**

Table 3: Accuracies (%) when applying ULR to the RoBERTa single encoder architecture ("baseline").

		AG	DBP	Yahoo
cosine	ensemble	79.1	84.6	**54.5**
	+ ULR	**81.1**	**93.3**	54.4
L2	ensemble	72.4	79.0	56.5
	+ ULR	**73.4**	**90.7**	**61.7**

Table 4: Accuracies (%) of ensemble predictions using 10 choices of label names using the RoBERTa dual encoder architecture ("ensemble"), and results when combining it with ULR.

to minimize binary cross entropy between the predicted score $\text{score}(\text{enc}(c, t))$ and a binary label y, where score is a linear function that transforms the vector into a scalar score.

5 Experimental Results of ULR

Dual encoder models. With the dual encoder models, we used two sets of distance measures. The first distance is the cosine distance of two vectors. In this case, we always normalize the vector representations before applying ULR. The second distance measure is the L2 distance.

Table 2 shows results for the GloVe and RoBERTa dual encoder model with two distance functions. With GloVe, except for the single case of cosine distance with Yahoo, all accuracies improve, with some improving by large amounts (up to 20% absolute). With the RoBERTa dual encoder model, the choice of distance function matches the scoring function we used when fine-tuning RoBERTa, i.e., cosine distance is used with dot product scoring and L2 distance is used with the contrastive hinge loss. ULR improves accuracies by 3.6% to 4.4% on average, and the improvements are consistent across distance functions and datasets, except for the cases of 20 NEWSGROUPS with L2 distance and AG with cosine distance, which show slight degradations.

Single encoder model. Table 3 summarizes the results of applying ULR to the RoBERTa single encoder architecture. ULR improves performance across all four datasets, ranging from 0.5% for 20NG up to 6.8% on DBP. Additional experiments in the appendix show the impact of early stopping and interpolation using this architecture. **[added this sentence –KPG]**

Label ensembles. Finding the best choice of label names for dataless classifiers is difficult without labeled data. Therefore, it is easier to supply multiple choices of label names for a given task.

We manually pick 10 different sets of label names for each dataset,[10] generate category and text representations with our RoBERTa dual encoder model, and perform ULR. The exact choices of label names are in the appendix. The predicted scores for the 10 sets are summed up as the final ensemble predictions. Table 4 shows results. Compared to Table 2, all accuracies on all tasks improve. Even with a stronger starting point from ensembling, ULR still yields consistent improvements in accuracy, with the single exception of Yahoo and cosine distance.

6 Robustness Experiments

Different choices of label names. Dataless text classification tasks are known to suffer from high variance due to label descriptions. Performance can vary dramatically across different choices of category names, as shown in Table 1.

One advantage of ULR is that it is robust to label noise. That is, even given a poor choice of label names, ULR can help the classifier to partially recover some of the accuracy, as shown in the lower

[10] We exclude 20NG mainly because it is difficult to come up with synonyms for its labels for this ensemble experiment and the robustness analysis in Section 6, since some labels in 20NG include proper nouns like "windows", "ibm", "mac", etc.

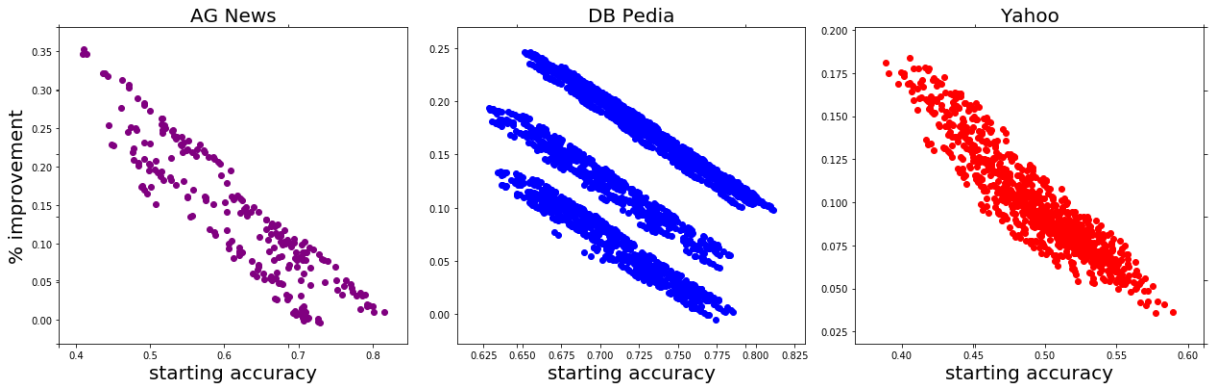

Figure 2: Accuracy (%) improvement when applying ULR compared to the RoBERTa dual encoder model (with Euclidean distance). The horizontal axis is the initial accuracy and the vertical axis is the absolute accuracy improvement after ULR.

		AG News	DBP	Yahoo
cos.	baseline	69.6 (8.7)	79.4 (3.2)	43.8 (3.9)
	+ ULR	**77.5** (5.5)	**91.6** (3.2)	**45.3** (3.9)
	# imp.	$\frac{213}{240}$ = 89%	$\frac{3867}{3867}$ = 100%	$\frac{732}{1015}$ = 72%
L2	baseline	62.2 (9.6)	71.7 (3.8)	49.5 (3.8)
	+ ULR	**75.2** (4.1)	**85.2** (5.5)	**59.3** (1.5)
	# imp.	$\frac{237}{240}$ = 99%	$\frac{2960}{2963}$ = 100%	$\frac{947}{947}$ = 100%

Table 5: Robustness analysis when varying choices of label names. The "baseline" and "+ ULR" rows show average accuracies (%) with standard deviations in parentheses of the RoBERTa dual encoder architecture among all category naming choices. The "# imp." rows show the numbers and percentages of cases where the performance improves after ULR.

portion of Table 1. We now describe similar experiments on a larger scale. In particular, we test several hundred sets of label names for each of AG, DBP, and Yahoo. For each category, we randomly assign different label names to it, all having similar meaning. The exact combinations of label names are in the appendix.

We then perform dataless text classification and ULR with our RoBERTa dual encoder model, either under cosine or L2 distance. Table 5 shows the results of the average performance gains before and after ULR. We also report the number and percentage of cases in which ULR improves accuracy. ULR improves the performance on average for all three tasks, and improves individual accuracies in the vast majority of cases. These results show that ULR is not only effective at improving the accuracy of dataless text classifiers across a wide range of label name sets, but also can help to mitigate harmful effects due to suboptimal label names.

We next study the relationship between the RoBERTa dual encoder model's initial accuracies and its accuracy gains with ULR. Figure 2 plots accuracy improvements vs. initial accuracies for the three datasets. We find that ULR gives larger gains when the initial accuracies are lower.

We also computed oracle accuracies by choosing the combination of label names that maximize accuracy (without ULR). With cosine distance, they are 81.9% for AG, 87.5% for DBP, and 53.3% for Yahoo. With L2 distance, they are 81.5% for AG, 81.1% for DBP, and 58.9% for Yahoo. Using ULR with the RoBERTa dual encoder provides better performance on DBP and Yahoo, and competitive results on AG, as shown in Tables 2 and 4.

Clustering with random initialization. We now investigate the impact of our initialization in ULR. We consider a variation in which we randomly initialize centroids. Since we can no longer link clusters and categories, this task becomes unsupervised text classification. A standard k-means algorithm is applied to update the centroids. Unlike Algorithm 1, in this experiment we do not interpolate the updated centroids with the initial centroids, as the initial centroids are random. The accuracy is calculated based on the oracle one-to-one mapping between final centroids and categories. This is often referred to as "one-to-one accuracy".

Table 6 presents results on AG with this experiment. We performed 240 trials with different random initialization of the centroids, and unsurprisingly, accuracy is improved in all cases. With L2 distance, ULR with random initialization even outperforms centroids initialized as category embeddings, which shows that unsupervised clustering is

		cosine	L2
AG News	baseline	37.3	39.9
	+ ULR	**61.4**	**75.8**
	# improved	$\frac{239}{240}$ = 99.6%	$\frac{240}{240}$ = 100%

Table 6: Average 1-to-1 accuracy (%) of ULR with random initialization of centroids. "baseline" and "ULR" are average accuracy (%) of the RoBERTa dual encoder architecture among all random initialized centroids. "# imp." are the numbers and percentages of cases where the performance improves after ULR.

	AG	DBP	Yahoo	20NG	avg
baseline	74.0	84.6	52.3	36.4	61.8
+ 30 labeled ins.	81.5	97.1	66.0	58.1	75.7
+ ULR	**85.2**	**97.4**	**66.7**	**58.2**	**76.9**

Table 7: Accuracies (%) when adding 30 labeled instances for each category with the RoBERTa dual encoder model ("baseline").

powerful in text classification with good text representations. However, since we do not have the one-to-one mapping between category and final centroids in the purely unsupervised setting, the results in Table 6 are not directly comparable to those in earlier tables.

7 ULR with Few Shot Learning

In this section, we apply ULR in the few shot learning setting. In particular, we draw 30 labeled instances for each category from the training split in the original datasets. We then further fine-tune our RoBERTa dual encoder model with the labeled instances. We adopt the hyperparameters of fine-tuning text classifiers from Wolf et al. (2019) and fine-tune for 3 epochs on these 30 labeled instances for each category. Then we apply ULR on the unlabeled test set in addition to the 30 labeled instances from each category.

Unlike Algorithm 1, these 30 labeled instances from each category are fixed to be assigned to their corresponding clusters. We compute the centroids of these clusters from 30 labeled instances by $r_c^l = \frac{\sum_{t \in \mathcal{L}_c} \text{enc}(t)}{|\mathcal{L}_c|}$ $\forall c \in \mathcal{C}$ where documents $\mathcal{L}_c$ are labeled with c $\forall c \in \mathcal{C}$.

At every iteration, the update rule of the centroids as in Algorithm 1 will include the text vector representations of these 30 labeled instances, i.e., $r_c = (\frac{\sum_{t:\text{pred}_t=c} \text{enc}(t)}{\text{count}(\{t:\text{pred}_t=c\})} + r_c^l + 2 \times \text{enc}(c))/4$.

Table 7 summarizes the results of combining 30 labeled instances for each category and ULR. Unsurprisingly, adding labeled instances improves accuracy by a large margin. Even so, ULR yields an additional improvement of 3.7% for AG, and also improves on the other datasets.

8 Analysis

A key challenge of dataless text classifiers is semantic drift between the training data and downstream tasks. As ULR is based on k-means, the centroids move towards the embeddings of the majority of text instances belonging to that category. Hence after ULR the locations of the centroids combine the information in the initial dataless text classifiers and the document sets for the downstream tasks.

Using AG, Table 8 shows the 5 text instances closest to the centroids for "business" and "sci&tech" before and after ULR. These categories are hard to distinguish as technology companies often appear in both business and technology news. However, after ULR, we see that texts containing technology companies ("LeapFrog", "ScanSoft") become less central for the "business" category, replaced by texts describing transactions and earnings. Also, a text containing a company name ("NEC") has become less central for the "sci&tech" category, replaced in the top 5 by a similar text that does not emphasize a company name.

In certain text classification settings, we may be able to abstain from classifying instances we are less confident about. This notion can be instantiated by checking the texts closest to the centroids corresponding to the labels, which we can view intuitively as the ones the model is most confident about. Table 9 shows a confidence-based evaluation, evaluating the k texts that are closest to each centroid and showing that ULR consistently improves precision at k.

9 Related Work

Earlier we discussed prior work on dataless and zero-shot text classification. We briefly introduce more related work in this section. Song et al. (2015) provide a text label refinement algorithm to adjust the label set with noisy and missing labels. There is also a wealth of prior work in semi-supervised text classification: using unlabeled text to improve classification performance (Nigam et al., 2000; Mukherjee and Awadallah, 2020; Xie et al., 2019). These methods typically learn generally useful text representations from a large corpus of unlabeled text and use them for a specific target task with

category	baseline	+ ULR
business	Hicks Muse Pays for ConAgra's Swift Stake ... EDS Is Charter Member of Siebel BPO Alliance ... GM, Daimler Go Green ... *LeapFrog*'s Greener Pastures (The Motley Fool) ... *ScanSoft* to acquire 3 software firms ...	CA to buy Netegrity for 430 m ... Sara Lee 1st-Quarter Net Rises on Fee (Reuters) ... Sears and Kmart Agree to Merge in 11 Billion Deal ... LCC Int Posts 3Q Profit, Shares Tumble, ... Before-the-Bell: GenCorp Falls 5.6 Pct. (Reuters) ...
sci&tech	Robot eats flies to make power ... Particle lab celebrates 50th birthday ... Time on a Chip: The Incredible Shrinking Atomic ... *NEC* Unveils World Fastest Vector Supercomputer .. Breakthrough on hydrogen fuel ...	Particle lab celebrates 50th birthday ... Breakthrough on hydrogen fuel ... Time on a Chip: The Incredible Shrinking Atomic ... Record Breaking Supercomputer Performance ... Robot eats flies to make power ...

Table 8: Top-5 scoring texts belonging to categories "business" and "sci&tech" in AGNews.

		P@10	P@50	P@100	P@500
AG	baseline	85.0	87.5	89.3	85.4
	+ ULR	85.0	**89.5**	**91.5**	**89.9**
DBP	baseline	83.6	84.1	84.1	84.6
	+ ULR	**92.1**	**92.1**	**92.6**	**91.5**
Yahoo	baseline	79.0	84.8	83.6	83.3
	+ ULR	**91.0**	**90.2**	**89.4**	**87.2**
20NG	baseline	64.5	62.0	57.6	35.7
	+ ULR	**71.5**	**65.0**	**62.0**	**38.8**

Table 9: Precision (%) at k (P@k) with dual encoder RoBERTa and cosine scoring.

limited supervision (Howard and Ruder, 2018; Devlin et al., 2019; Liu et al., 2019; Lan et al., 2020; Peters et al., 2018). Metric learning (Wohlwend et al., 2019) is related to our work, but they focus on few-shot learning and we work on improving unsupervised text classifiers. Related contemporaneous work has proposed methods to generate more relevant label names from a given set (Meng et al., 2020; Schick et al., 2020). ULR is orthogonal to such methods of choosing label names, as these label names can be set as extra initial centroids. Finally, label descriptions have also been exploited in supervised learning to improve text classifiers (Chai et al., 2020; Wang et al., 2019; Sun et al., 2019).

10 Conclusion

Our proposed ULR algorithms provide a simple but effective framework to improve the performance of dataless text classifiers. Extensive experiments show its flexibility and robustness, offering promise for making dataless text classification more useful for practitioners.

Acknowledgments

We would like to thank the anonymous reviewers for helpful comments. This research was supported in part by a Bloomberg data science research grant to KS and KG.

References

Arindam Banerjee, Srujana Merugu, Inderjit S Dhillon, and Joydeep Ghosh. 2005. Clustering with Bregman divergences. *Journal of machine learning research*, 6(Oct):1705–1749.

Duo Chai, Wei Wu, Qinghong Han, Fei Wu, and Jiwei Li. 2020. Description based text classification with reinforcement learning. *ICML*.

Ming-Wei Chang, Lev Ratinov, Dan Roth, and Vivek Srikumar. 2008. Importance of semantic representation: Dataless classification. In *AAAI*.

Xingyuan Chen, Yunqing Xia, Peng Jin, and John Carroll. 2015. Dataless text classification with descriptive LDA. In *AAAI*.

Zewei Chu, Karl Stratos, and Kevin Gimpel. 2020. Natcat: Weakly supervised text classification with naturally annotated datasets. *arXiv preprint arXiv:2009.14335*.

Yann N Dauphin, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. 2013. Zero-shot learning for semantic utterance classification. *arXiv preprint arXiv:1401.0509*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Geli Fei and Bing Liu. 2016. Breaking the closed world assumption in text classification. In *NAACL*.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJCAI*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *ACL*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Yuezhang Li, Ronghuo Zheng, Tian Tian, Zhiting Hu, Rahul Iyer, and Katia Sycara. 2016. Joint embedding of hierarchical categories and entities for concept categorization and dataless classification. In *COLING*.

Percy Liang and Dan Klein. 2009. Online EM for unsupervised models. In *NAACL-HLT*.

Jianhua Lin. 1991. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151.

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. *CoRR*, abs/1907.11692.

Yukun Ma, Erik Cambria, and Sa Gao. 2016. Label embedding for zero-shot fine-grained named entity typing. In *COLING*.

Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. 2019. Weakly-supervised hierarchical text classification. In *AAAI*.

Yu Meng, Yunyi Zhang, Jiaxin Huang, Chenyan Xiong, Heng Ji, Chao Zhang, and Jiawei Han. 2020. Text classification using label names only: A language model self-training approach. In *EMNLP*.

Subhabrata Mukherjee and Ahmed Hassan Awadallah. 2020. Uncertainty-aware self-training for text classification with few labels. *arXiv preprint arXiv:2006.15315*.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable prediction of medical codes from clinical text. *arXiv preprint arXiv:1802.05695*.

Jinseok Nam, Eneldo Loza Mencía, and Johannes Fürnkranz. 2016. All-in text: Learning document, label, and word representations jointly. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Kamal Nigam, Andrew Mccallum, Sebastian Thrun, and Tom Mitchell. 2000. Text classification from labeled and unlabeled documents using EM. In *Machine Learning*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Anthony Rios and Ramakanth Kavuluru. 2018. Few-shot and zero-shot multi-label learning for structured label spaces. In *EMNLP*.

Timo Schick, Helmut Schmid, and Hinrich Schütze. 2020. Automatically identifying words that can serve as labels for few-shot text classification. In *COLING*.

Lei Shu, Hu Xu, and Bing Liu. 2017. Doc: Deep open classification of text documents. In *EMNLP*.

Yangqiu Song and Dan Roth. 2014. On dataless hierarchical text classification. In *AAAI*.

Yangqiu Song, Chenguang Wang, Ming Zhang, Hailong Sun, and Qiang Yang. 2015. Spectral label refinement for noisy and missing text labels. In *AAAI*.

Chi Sun, Luyao Huang, and Xipeng Qiu. 2019. Utilizing bert for aspect-based sentiment analysis via constructing auxiliary sentence. *arXiv preprint arXiv:1903.09588*.

Jingjing Wang, Changlong Sun, Shoushan Li, Xiaozhong Liu, Luo Si, Min Zhang, and Guodong Zhou. 2019. Aspect sentiment classification towards question-answering with reinforced bidirectional attention network. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3548–3557.

Pu Wang, Charlotta Domeniconi, and Jian Hu. 2009. Towards a universal text classifier: Transfer learning using encyclopedic knowledge. In *Data Mining Workshops*.

Jeremy Wohlwend, Ethan R. Elenberg, Sam Altschul, Shawn Henry, and Tao Lei. 2019. Metric learning for dynamic text classification. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 143–152, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. 2019. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*.

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 3905–3914.

Dani Yogatama, Chris Dyer, Wang Ling, and Phil Blunsom. 2017. Generative and discriminative text classification with recurrent neural networks. In *arXiv*.

Jingqing Zhang, Piyawat Lertvittayakumjorn, and Yike Guo. 2019. Integrating semantic knowledge to tackle zero-shot text classification. In *NAACL*.

Xiang Zhang, Junbo Zhao, and Yann Lecun. 2015. Character-level convolutional networks for text classification. In *NIPS*.

A Clustering Objective

For convenience, we first rewrite Eq. (1) below:

$$\min_{\{r_c\}_{c\in\mathcal{C}}} \min_{\{a_t\}_{t\in\mathcal{T}}} \sum_{t\in\mathcal{T}} ||\text{enc}(t) - r_{a_t}||^2 + \sum_{c\in\mathcal{C}} |c|\,||\text{enc}(c) - r_c||^2 \quad (2)$$

We also rewrite Algorithm 1 here as Algorithm 3:

Algorithm 3: ULR for dual encoder architectures

Data: documents $\mathcal{T}$, categories $\mathcal{C}$, encoder enc, scoring function score
initialize the centroids $r_c = \text{enc}(c)\ \forall c \in \mathcal{C}$;
while *not converged* **do**
  for $t \in \mathcal{T}$ **do**
    $s_{tc} = \text{score}(\text{enc}(t), r_c)\ \forall c \in \mathcal{C}$;
    $\text{pred}_t = \text{argmin}_c s_{tc}$;
  end
  $r_c = (\frac{\sum_{t:\text{pred}_t=c} \text{enc}(t)}{\text{count}(\{t:\text{pred}_t=c\})} + \text{enc}(c))/2\ \forall c \in \mathcal{C}$;
  $\text{objective} = \sum_t s_{t\text{pred}_t}$;
end
Result: s_{tc}, objective, pred_t, and r_c of all iterations

The objective (2) as a function of a single centroid corresponding to category $c \in \mathcal{C}$ can be written as

$$\min_{v\in\mathbb{R}^d} \sum_{t\in c} ||\text{enc}(t) - v||^2 + |c|\,||\text{enc}(c) - v||^2$$

where $c \in \mathcal{C}$ is constant given assignments a_t. It is easy to verify that the unique solution is given by

$$r_c = \frac{\frac{1}{c}\sum_{t\in c} \text{enc}(t) + \text{enc}(c)}{2}$$

which is the update in Algorithm 3. Now consider (2) as a function of category assignments

$$\min_{\{a_t\}_{t\in\mathcal{T}}} \sum_{t\in\mathcal{T}} ||\text{enc}(t) - r_{a_t}||^2 + \sum_{c\in\mathcal{C}} |c|\,||\text{enc}(c) - r_c||^2$$

where centroids r_c are held constant. Unfortunately this problem remains intractable because the regularization term is weighted by cluster sizes. We can approximate it by ignoring the regularization term: in this case the minimizer is given by $a_t = \arg\min_{c\in\mathcal{C}} ||\text{enc}(t) - r_c||$ for each $t \in \mathcal{T}$ as usual. Thus Algorithm 3 can be viewed as optimizing (2) by alternating minimization, with the simplifying (incorrect) assumption that cluster sizes are not significantly affected by category assignments.

B Experimental Setup

We list the hyperparameters we used to fine-tune our RoBERTa based dataless classification models in this section, so readers can reproduce our experimental results. The code with the scripts are also included in the supplementary material. We choose RoBERTa-base. In all fine-tuning tasks, we set the batch size to be 32. The max sequence length for RoBERTa is 128. The peak learning rate is 0.00002. We use a linear scheduler with warmup steps to be 10% of the total fine-tuning steps. The random seeds of all experiments are set to be 1.

We fine-tune RoBERTa models on random choices of single GPUs, including NVIDIA TITAN X, 1080Ti, or 2080 Ti. Most of the fine-tuning tasks can be finished within 8 hours.

C Label Names in the Evaluation Tasks

In the main experiments (Section 5), we use the following label descriptions (separated by ";") for the downstream tasks.

- AG: world; sports; business; science technology
- DBP: company; educational institution; artist; athlete; politician; transportation; building; nature; village; animal; plant; album; film; written work
- Yahoo: society culture; science mathematics; health; education reference; computers internet; sports; business finance; entertainment music; family relationships; politics government
- 20 NEWSGROUPS: atheist christian atheism god islamic; graphics image gif animation tiff; windows dos microsoft ms driver drivers card printer; bus pc motherboard bios board computer dos; mac apple powerbook; window motif xterm sun windows; sale offer shipping forsale sell price brand obo; car ford auto toyota honda nissan bmw; bike motorcycle yamaha; baseball ball hitter; hockey wings espn; encryption key crypto algorithm security; circuit electronics radio signal battery; doctor medical disease medicine patient; space orbit moon earth sky solar; christian god christ church bible jesus; gun fbi guns weapon compound; israel arab jews jewish muslim; gay homosexual sexual; christian morality jesus god religion horus

Label ensembles. In the experiment of ensembling labels, we use the following 10 sets of label name choices.

For AG:

- world; sports; business; science technology
- international; sports; business; science technology
- world; sports; business; science and technology
- international; sports; business; science and technology
- world; sports; business and finance; science technology
- international; sports; business and finance; science technology
- world; sports; business and finance; science and technology
- international; sports; business and finance; science and technology
- world politics; sports; business; science technology
- world politics; sports; business; science technology

For DBP:

- company; educational institution; artist; athlete; politician; transportation; building; nature; village; animal; plant; album; film; written work
- company; school; artist; athlete; politician; transportation; building; nature; village; animal; plant; album; film; written work
- company; educational institution; artist; athlete; politician; transportation; architecture; nature; village; animal; plant; album; movie; written work
- company; educational institution; artist; athlete; politician; transportation; building; nature; village; animal; plant; album; film; novel
- company; educational institution; artist; athlete; politician; transportation; building; nature; village; animal; plant; album; film; article
- company; educational institution; artist; athlete; politician; transportation; building; nature; village; animal; plant; collection; movie; written work
- company; school; artist; athlete; politician; transportation; architecture; nature; village; animal; plant; album; movie; written work
- company; educational institution; artist; athlete; politician; transportation; architecture; nature; village; animal; plant; album; movie; novel
- company; school; artist; athlete; politician; transportation; architecture; nature; village; animal; plant; album; film; novel

- company; educational institution; artist; athlete; official; transportation; building; nature; village; animal; plant; album; movie; article

For Yahoo:

- society culture; science mathematics; health; education reference; computers internet; sports; business finance; entertainment music; family relationships; politics government
- society culture; scientific discipline; health; education reference; computers internet; sports; business finance; entertainment music; family relationships; politics government
- society culture; science mathematics; health; learning teaching resource; computers internet; sports; business finance; entertainment music; family relationships; politics government
- society culture; science mathematics; health; education reference; computers internet; sports; business finance; entertainment music; love home; politics government
- society culture; science mathematics; health; education reference; information technology; sports; business finance; entertainment music; family relationships; politics government
- society culture; scientific discipline; health; education reference; information technology; sports; business finance; entertainment music; family relationships; politics government
- society culture; science mathematics; health; learning teaching resource; information technology; sports; business finance; entertainment music; family relationships; politics government
- society culture; science mathematics; health; education reference; computers internet; sports; commerce; entertainment music; family relationships; politics government
- society culture; science mathematics; health; education reference; information technology; sports; commerce; entertainment music; family relationships; politics government
- society culture; science mathematics; health; learning teaching resource; computers internet; sports; commerce; entertainment music; family relationships; politics government

D Robustness to Label Noise

In the robustness experiments (Section 6), we manually picked different choices of label names for each category of the downstream tasks. We list the choices of label names for each category below, separated by ";".

AG:

- world: world; world politics; world news; international; international news
- sports: sports; health; health and sports
- business: business; commerce; finance; business and finance
- science technology: science; technology; science and technology; science technology

DBP:

- company: company; corporation
- educational institution: educational institution; school
- artist: artist; creator
- athlete: athlete; sportsman
- politician: politician; official
- transportation: transportation
- building: building; architecture
- nature: nature
- village: village; suburb
- animal: animal; living thing
- plant: plant
- album; collection
- file: film; movie
- written work: written work; writing; novel; article

Yahoo:

- society culture: society culture; community
- science mathematics: science mathematics; scientific discipline
- health: health; fitness
- education reference: education reference; learning teaching resource
- computers internet: computers internet; information technology
- sports: sports; athletics
- business finance: business finance; commerce
- entertainment music: entertainment music; fun songs
- family relationships: family relationships; love home
- politics government: politics government; policy regime regulation

Different combinations of label name choices are used to generate category embeddings and perform ULR.

	AG	DBP	Yahoo	20NG	avg
baseline	72.6	64.7	57.1	28.7	55.8
+ULR	**75.6**	**75.1**	**59.9**	**35.9**	**61.6**

Table 10: Improvements of ULR with the RoBERTa dual encoder model on unbalanced datasets. **[I'm confused by the extra "ULR" in both rows – is that a typo? –KPG]** **[Yes, I removed it. –Zewei]**

	AG	DBP	YAHOO	20NG	avg
baseline	72.6	81.8	59.3	36.0	62.4
+ ULR					
	75.1	**88.6**	**60.0**	36.5	**65.0**
+ early	72.6	88.6	59.3	36.5	64.3
+ ULR + interpolate					
	74.0	87.2	59.6	**36.9**	64.4
+ early	74.0	87.2	59.6	**36.9**	64.4

Table 11: Accuracies (%) when applying ULR to the RoBERTa single encoder architecture ("baseline"). "+interpolate" are the results when we update centroids with interpolation. "+ early" means early stopping. **[unfinished sentence here? –KPG]** **[Finished. More details in the following text. –Zewei]**

E Augmented Categories

In the experiment of augmenting categories with ULR (Section 8), we use the following augmented categories (separated by ";"): math; gis; physics; codereview; stats; unix; english; tex; gaming; apple; scifi; drupal; ell; meta; electronics; travel; rpg; dba; magento; webapps; diy; wordpress; android; security; chemistry; webmasters; blender; softwareengineering; gamedev; academia.

F Unbalanced Dataset

We manually create unbalanced datasets, where each category only contains 1-100% of the original instances. **[Zewei, can you say more about how extreme the imbalance is? The range of 1-100% is a pretty big range. :) –KPG]** For a dataset with n categories, each instance belonging to category i has a probability of $\frac{i}{n}, i \in \{0, 1, \cdots, n-1\}$ of being dropped, where the orders of categories are random. **[Added this sentence to explain. –Zewei]** Table 10 shows that ULR improves performance with unbalanced datasets.

G Single Encoder

Table 11 summarizes the results of applying ULR on the single encoder model with two variations:

1. early: early stopping on the average Jenson-Shannon Divergence score of all probability distribution of text over all categories to their corresponding nearest one-hot centers;

2. interpolate: when updating centroids in Algorithm 2, we take the average of the new centroids with the centroids from the previous iteration.

Overall, these two variations do not provide better results in the setting of single encoder models.