Incubating Text Classifiers Following User Instruction with Nothing but LLM

Letian Peng, Zilong Wang, Jingbo Shang*
Department of Computer Science
University of California, San Diego
{lepeng, zlwang, jshang}@ucsd.edu

Abstract

In this paper, we aim to generate text classification data given arbitrary class definitions (i.e., user instruction), so one can train a text classifier without any human annotation or raw corpus. Recent advances in large language models (LLMs) lead to pioneer attempts to individually generate texts for each class via prompting. In this paper, we propose Incubator, the first framework that can handle complicated and even mutually dependent classes (e.g., "*TED Talk given by Educator*" and "*Other*"). Specifically, our Incubator is a fine-tuned LLM that takes the instruction of all class definitions as input, and in each inference, it can jointly generate one sample for every class. First, we tune Incubator on the instruction-to-data mappings that we obtained from classification datasets and descriptions on Hugging Face together with in-context augmentation by GPT-4. To emphasize the uniformity and diversity in generations, we refine Incubator by fine-tuning with the cluster centers of semantic textual embeddings of the generated samples. We compare Incubator on various classification tasks with strong baselines such as direct LLM-based inference and training data generation by prompt engineering. Experiments show Incubator is able to (1) outperform previous methods on traditional benchmarks, (2) take label interdependency and user preference into consideration, and (3) enable logical text mining by incubating multiple classifiers.

1 Introduction

Text classification is one of the most fundamental natural language processing (NLP) tasks and plays a vital role in many NLP systems (Han and Kamber, 2000). Traditional supervised text classification fine-tunes models on expensive human annotation (Zhang et al., 2015), limiting its usage for lower-source domains. Zero-shot text classification reduces manual effort by building classifiers

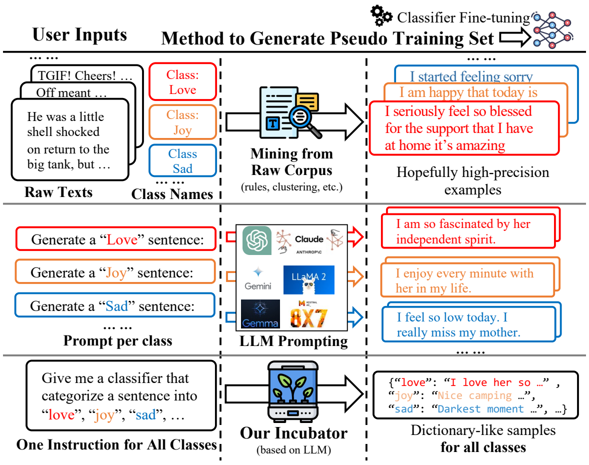

Figure 1: A comparison of Incubator with different methods for zero-shot text classification.

with minimal inputs, such as label names (Wang et al., 2021; Zhang et al., 2023b; Wang et al., 2023a). These zero-shot methods are typically based on mining pseudo-training data from massive raw texts with precise filtering algorithms, which unfortunately limits their application to simple labels. For more complex labels, their distributions are extremely scarce in raw texts and filtering algorithms struggle to recall these examples while maintaining their precision.

Large language models (LLMs) (Touvron et al., 2023a,b; OpenAI, 2023), such as GPT-3 (Brown et al., 2020), have been recently introduced to address this problem with their proficient capability to capture the nuance in complex labels. Specifically, people prompt LLMs to generate data based on each label, and then fine-tune small classifiers as the final production (Ye et al., 2022a,b).

Existing LLM-based zero-shot text classification methods, while feasible, face two major challenges, (1) class definitions can go beyond a simple label name, such as "*TED Talk given by Educator*" and (2) class definitions can depend on each other. For example, the class "*Other*" is only

* Corresponding author.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 3753–3766
November 12-16, 2024 ©2024 Association for Computational Linguistics

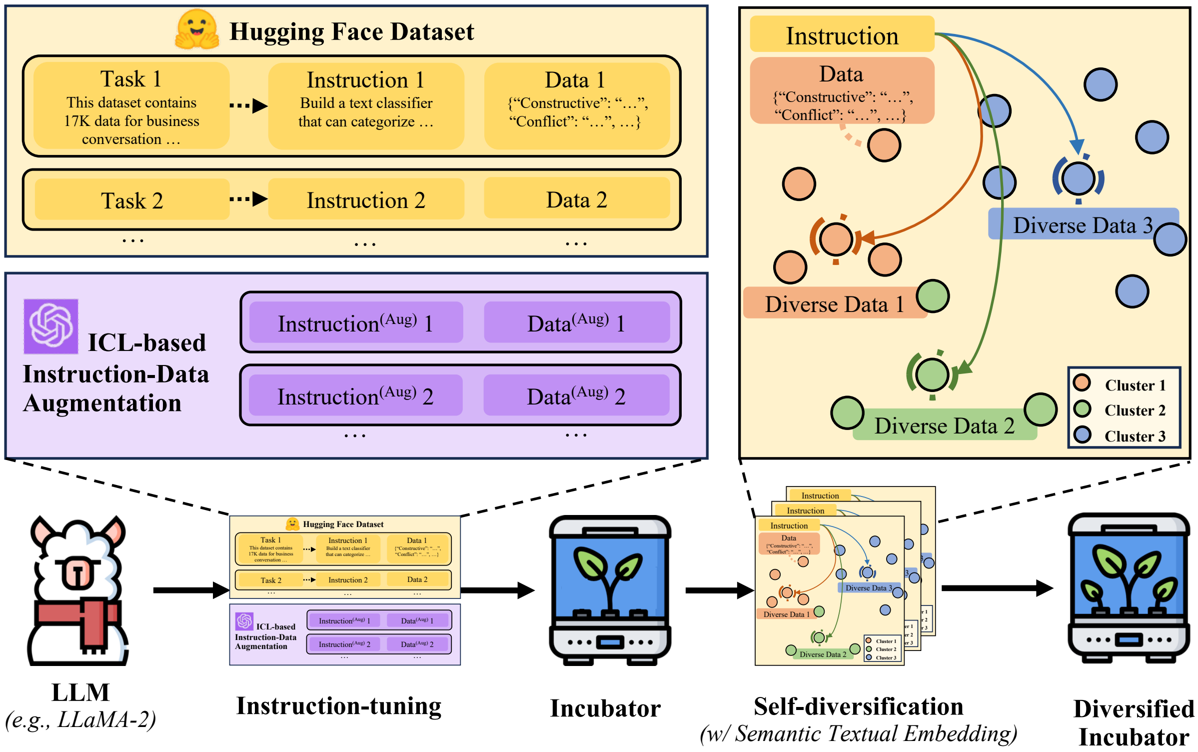

Figure 2: An overview of our Incubator framework.

meaningful when seeing other classes; As shown in Figure 1, the class "*Optimistic*" shall not contain "*Love*" when "*Love*" itself presents as a class. Therefore, the scope of the class with the same textual definition can vary as other classes change.

We argue that the LLMs need further instruction-tuning (Ouyang et al., 2022), particularly for classification data generation. Specifically, we leverage public classification datasets with descriptions for this tuning. This allows the user to control the LLM to generate useful training data for small models based on (1) label interdependency and (2) user preferences described in the instructions. Consequently, the LLM-based zero-shot text classification can be formalized as *model incubation* that "User requires a model by an instruction, the LLM (**Incubator**) then generates useful training data to incubate such a classifier."

In this paper, we first collect pairs of dataset descriptions and training data samples on Huggingface (Wolf et al., 2019a), each formalized as a dictionary with each label as a key and a sample as the value. These data are beneficial for Incubator to learn label interdependency as the examples from different classes are presented jointly. Then the data descriptions are manually converted to user instructions, which establishes a mapping from the user instruction to training data. These instructions are augmented by a relatively strong LLM (e.g., `GPT-4`) using in-context learning (ICL) (Dong et al., 2023b) and used to instruction-tune an open-source LLM (e.g., `LLaMA-2-7b-hf`) as our Incubator. Note that we can leverage `GPT-4` with ICL as Incubator too. We recommend open-source LLMs as Incubator because of open parameters, inference efficiency, and further fine-tuning.

To alleviate the known negative impact of data bias on text classification (Dixon et al., 2018; Li et al., 2021b; Jin et al., 2022) and bias in contents generated by LLMs (Gallegos et al., 2023; Fang et al., 2023), we propose a novel self-diversification technique to increase the data uniformity and diversity, utilizing the text representations from a text embedder (Wang et al., 2022). Specifically, we instruct the Incubator many times (e.g., 1024), and then use a clustering algorithm (e.g., K-means) to get the sample nearest to each cluster center which are semantically different from one another. These samples are incorporated in the same batch to further instruct-tune Incubator to increase the data uniformity and diversity.

We conduct experiments to test the classifier incubation ability of our Incubator on various tasks to test its basic incubation ability, label interdependency awareness, and user instruction following ability. These tasks involve incubating classifiers for (1) traditional benchmarks, (2) classification tasks with "*Other*" as a label, and (3) classification tasks with user customization for personal preference. We include strong baselines such as directly instructing the LLM to classify texts and prompting LLMs to generate data for each label separately.

Experiment results verify our Incubator to be able to (1) incubate strong text classifiers that outperform the baselines, (2) consider the label interdependency and follow the user preference in the instruction, (3) incubate multiple text classifiers and use logical conjunctions to realize advanced text mining systems.

Our contributions in this paper are three-fold.

- We propose an instruction-tuning framework for LLMs, which incubates text classifiers following user instructions for complicated and mutually dependent classes.
- We propose a novel self-diversification technique, which utilizes the cluster centers of generated samples to increase the uniformity and diversity in Incubator generation.
- We conduct extensive experiments on benchmark datasets to demonstrate the superior accuracy of the incubated text classifiers.
- We showcase how to apply Incubator to realize advanced text mining systems by incubating multiple text classifiers with logical conjunctions[1].

2 Related Works

2.1 Zero-shot Text Classification

Traditional zero-shot text classification methods are based on text mining in massive raw texts with label names for surface form matching (Wang et al., 2021; Wang and Shang, 2022; Zhang et al., 2023b; Wang et al., 2023a) or semantic matching (Hanjie et al., 2022; Aggarwal et al., 2023; Zhao et al., 2023). A related setup allows incorporating some seed words for each class to strengthen the text mining precision (Wang et al., 2023b; Dong et al., 2023a). With the emergence of LLMs, many pioneer studies on LLM-based zero-shot text classification propose to prompt LLMs with label names to synthesize texts falling in target classes. These texts are used to fine-tune small classifiers on those generated results (Ye et al., 2022a,b). However, these methods are substantially label-wise text generation, which fails to consider the whole classification task, involving label interdependency and user preference. Our work aims to fill in this blank by proposing a framework that builds customized classifiers according to user instructions.

2.2 Instruction-tuning

Following instructions (Zhang et al., 2023a) is a fundamental capability for Large Language Models (LLMs), crucial for understanding and acting upon user commands, thus enhancing their appeal to user-specific applications. InstructGPT (Ouyang et al., 2022) represents an initial exploration into LLMs tailored to follow instructions, revealing their capacity to perform tasks as directed by users. ChatGPT (OpenAI, 2023), with its superior capability to follow instructions, bolstered by reinforcement learning with human feedback (RLHF), has enjoyed considerable acclaim both within and beyond the language research community. Furthermore, publicly available LLMs designed for instruction-following, such as LLaMA (Touvron et al., 2023a,b; Meta, 2024), offer a rich foundation for investigating the ability of LLMs to execute instructions. Instruction-tuning not only contributes to the success of LLMs in text-to-text tasks (Zhang et al., 2023a), but is also able to customize image generation (Chae et al., 2023) and text embeddings (Peng et al., 2024). Our work follows this trend to instruction-tune LLMs as Incubator, which customize classifiers according to user instructions.

2.3 Model Incubation

The area closest to model incubation is symbolic distillation (West et al., 2022; Li et al., 2023), which distills a teacher model into a different type of student model. Those student models can function very differently from the initial language modeling teacher, such as commonsense reasoning (West et al., 2022) and information extraction (Zhou et al., 2023). Another relevant domain is training data generation including augmentation. Besides classification data generation (Ye et al., 2022a,b; Peng et al., 2023), there also exists generation pipelines for question answering (Do et al., 2023; Gou et al., 2023) and natural language generation (Xu et al., 2021). Model incubation differs from previous works as it takes user instruction as the input, which allows a more user-oriented model customization for personal usage.

3 Our Incubator Framework

Figure 2 offers an overview of our Incubator framework, including two stages, (1) **Instruction-tuning**

[1]The datasets and models used in the experiments will be released for reproducibility.

and (2) **Self-diversification**. The instruction-tuning stage utilizes the existing resources on the Huggingface platform to learn an LLM as Incubator to generate training data based on user instructions. The self-diversification stage further improves the uniformity and diversity in Incubator generation with an auxiliary text embedder and clustering. We now elaborate on the details of these two stages.

3.1 Instruction-tuning for Incubator

Instruction-to-data Dataset We select 25 text classification datasets on the Huggingface platform[2] to build the initial instruction-to-data dataset for instruction-tuning, such as financial news, counterfactual reviews, and toxic conversations. For each dataset, we extract its description and sample a few (we select 10) samples per class from it, which are formalized as Python dictionaries. The keys in the dictionary are labels and each label corresponds to one text data as the value. Consequently, we get 250 instruction-to-data samples as the initial dataset. We present a specific case inside the dataset in the Appendix B.

ICL-based Augmentation Directly instruction-tuning the LLM on the initial dataset will likely introduce overfitting and bias to the Incubator due to the limited number of instructions (Song et al., 2023). Thus, we address these issues by data augmentation (Ye et al., 2024) and use ICL (Dong et al., 2023b) by `GPT-4` (OpenAI, 2023) as the implementation (Ho et al., 2023). We show the specific prompt for in-context learning in Table 8 of Appendix C. We have two in-context examples that map instructions to training data as Python dictionaries, which are randomly sampled in each query. Finally, we augment the instruction-to-data dataset to 12K samples. This dataset is then used to fine-tune the LLM as the Incubator.

3.2 Self-diversification for Incubator

Dataset uniformity and diversity are essential to text classification (Dixon et al., 2018) while the contents from LLMs are generally biased, especially when sampling from a single instruction (Gallegos et al., 2023; Fang et al., 2023). Thus, we propose a novel self-diversification technique to improve the generation quality from our Incubator. The main idea is to instruction-tune the LLM on the same instruction with several semantically different data samples. We refer to a pre-trained text embedder, specifically E5 (Wang et al., 2022), to calculate the semantic similarity (Chandrasekaran and Mago, 2022). In our implementation, we reuse the instructions in the instruction-tuning dataset. For each instruction, we generate many (We select 1024) training data[3] and encode the data into the latent embedding space. As the data are formalized as Python dictionaries, we concatenate the embeddings of the values (texts) corresponding to a fixed order of keys.

$$E(d) = \bigoplus_{i=1}^{n} E(d[l_i])$$

where $E(\cdot)$, d, l_i refer to the encoder, the data (dictionary) and the i-th label. $\oplus$ represents the concatenation operation and n represents the total label number. After all data are encoded, we run a K-means (We select $K = 8$) clustering algorithm on the embeddings and find the K samples with embeddings that are closest to the cluster centers. These samples, together with the instruction, establish a one-to-many mapping that maps instruction to very semantically diverse data samples. We incorporate these data in a batch of K and further instruction-tune the LLM on it. Intuitively, this procedure will increase the appearance probability of data with unique semantics to benefit the incubated classifier.

4 Experiments

We conduct several experiments to evaluate the performance of our Incubator. We include experiments on traditional datasets, and revised datasets with the label "Other". We also evaluate the ability of Incubator to handle complex personal labels and even ones with conjunctive relationships.

4.1 Evaluations and Datasets

Towards a comprehensive evaluation of our Incubator, we organize the evaluation into three scenarios.

(1) Traditional Benchmarks We include 8 traditional text classification benchmarks, such as sentiment analysis classification (1) **SST-2** (Socher et al., 2013), (2) **SST-5** (Socher et al., 2013), and (3) **Emotion** (Saravia et al., 2018), topic classification (4) **AG News** (Zhang et al., 2015), news location classification (5) **NYT-LOC** (Mozzherina, 2013), question type classification (6) **TREC** (Li and Roth,

[2] The source datasets are shown in Appendix E.

[3] Generally, the data share the same label set.

Method	SST-2	SST-5	Emotion	AG News	NYT-LOC	TREC	SNIPS	Hillary	Average
Prompting	91.43	39.95	46.65	77.65	71.07	60.80	42.29	63.46	61.66
Debiased Seed†	84.38	25.48	18.67	**81.31**	76.79	34.86	87.96	48.29	57.22
SemSup-XC++†	85.67	37.87	39.45	72.26	81.46	19.80	66.86	45.32	56.09
ZeroGen++	82.04	39.37	45.40	65.57	78.62	59.10	89.78	57.97	64.73
ProGen++	84.07	41.49	46.00	67.72	79.64	59.80	90.21	57.42	65.79
Incubator (Ours)	**90.01**	**46.06**	46.55	69.46	81.86	**71.40**	**93.57**	**67.31**	**70.78**
-Diversification	85.45	45.29	**46.80**	69.91	**83.58**	63.60	91.07	64.01	68.71
Incubator w/ GPT-4	86.99	44.43	47.80	80.79	86.87	77.80	94.14	64.01	72.85

Table 1: Text Classification Benchmark Results. All methods are based on LLaMA except for *Incubator w/ GPT-4*. †: Methods require more than label names. **Debiased Seed** requires a raw corpus for text mining. **SemSup-XC++** requires a pre-trained text embedded for semantic similarity calculation

2002), intent classification (7) **SNIPS** (Coucke et al., 2018), and (8) sentiment classification towards a particular public figure **Hillary** (Barbieri et al., 2020).

(2) Label "Other" We also test the ability of Incubator to handle stronger label interdependency by datasets with "*Other*". We convert several datasets by grouping minor categories based on the proportion as a single "Other" label, with details mentioned in the Appendix D. These datasets include unbalanced datasets: Emotion, NYT-LOC, and **Massive** (FitzGerald et al., 2022). These revised datasets will be also released for reproducibility

(3) Complicated Class Definitions To further showcase the usefulness of Incubator, we come with several complicated instructions for Incubator to incubate text classifiers that will be later used to mine the desired texts from massive raw documents, such as **TED Talk Summary**[4], **Steam Game Description**[5], and **Text Message**[6].

Note that all the datasets in our evaluations are **excluded** from the instruction-tuning data of Incubator.

4.2 Implementation Details

We implement Incubator by fine-tuning the parameters of LLaMA-2 (`LLaMA-2-7b-hf`) (Touvron et al., 2023b) on our constructed instruction-tuning dataset with AdamW optimizer (Loshchilov and Hutter, 2019) and cosine annealing learning rate scheduler (Loshchilov and Hutter, 2017). The specific hyperparameters for the optimization are shown in Table 7 in Appendix A.

For all experiments, our Incubator is queried to generate 1024 data dictionaries, each with one sample per class, to incubate a small classifier, which is selected as `RoBERTa-Large` (Liu et al., 2019). The RoBERTa is fine-tuned with the same optimizer and scheduler as for instruction-tuning and the hyperparameters for the incubation are also presented in Table 7.

[4]Huggingface: chirunder/gigant/ted_descriptions
[5]Huggingface: FronkonGames/steam-games-dataset
[6]Huggingface: chirunder/text_messages

4.3 Compared Methods

One can directly prompt the LLM, LLaMA-2 (`LLaMA-2-7b-hf`), which is the same as the LLM used in Incubator, with all the labels and the input text in the prompt and ask it to categorize the text into one of the labels (Sun et al., 2023). We name this method as **Prompting**.

We first include some traditional zero-shot text classifications for reference:

- **Debiased Seed** (Dong et al., 2023a): This is a state-of-the-art text mining method for zero-shot text classification. The method precisely assigns pseudo-labels to texts by seed word (the same as label name (Wang et al., 2023c)) matching with label cleaning (Mekala et al., 2022). These mined texts are then used to fine-tune a classifier.
- **SemSup-XC++** (Aggarwal et al., 2023): This method uses semantic similarity (Chandrasekaran and Mago, 2022) between texts and label descriptions to assign labels with the highest description similarity to texts. The original SemSup-XC mines class descriptions and trains a text embedding by contrastive learning (Gao et al., 2021). We upgrade SemSup-XC to a stronger SemSup-XC++ for LLMs and the advancement in text embedding. We generate the class descriptions by a state-of-the-art LLM, GPT-4 (OpenAI, 2023), and produce the embeddings by a strong text embedder (Wang et al., 2022).

For the main comparison, we include strong baselines that generate training data without requiring massive raw texts as follows.

- **ZeroGen++** (Ye et al., 2022a): This method prompts LLMs (`LLaMA-2-7b-hf`) to generate texts based on label descriptions in generation instructions. Different from our Incubator, ZeroGen handles each label separately, such as "Generate a negative movie review". Towards a fair comparison with our method, we formalize our instruction-tuning dataset as the template used in ZeroGen to further fine-tune the model. The baseline upgraded by further fine-tuning is named ZeroGen++.
- **ProGen++** (Ye et al., 2022b): This method further develops ZeroGen++ by an iterative ICL-based augmentation. With the classifier obtained from ZeroGen++, ProGen++ selects the most helpful data with an influence function (Koh and Liang, 2017) that measures the change in the model's loss on the test data point. The most influential data points are selected as in-context examples to prompt the LLM to generate more helpful data to strengthen the classifier.

Incubator w/ GPT-4: This is a variant of our Incubator that prompts `GPT-4` with in-context examples from the Huggingface platform and the instruction to sample the training data. We present this not as a baseline but to showcase that the Incubator idea also applies to propriety LLMs.

All data generation baselines generate the same amount of data (1024 per class) towards a fair comparison. The reported results are the average of 5 runs, except for SemSup-XC++, which does not have randomness in the method.

4.4 Traditional Benchmark Results

The experiment results on traditional benchmarks are shown in Table 1. The comparison between ZeroGen and ProGen baselines verifies our Incubator has a significant advantage over those label interdependency-agnostic methods, which indicates the advantage of Incubator to consider the full label set in the instruction.

Moreover, the self-diversification procedure is shown to highly contribute to the performance of Incubator, which boosts the performances on 5 out of 8 datasets and achieves comparable performances on others. Thus, self-diversification is verified to be a reliable and beneficial technique to strengthen the Incubator.

In comparison with data generation methods, the text mining and semantic similarity-based baselines show significant limitations on some datasets. For instance, Debiased Seed shows a significantly weak performance on Emotion and TREC (question classification) as the seed words are hard to propose for these classes. SemSup-XC++ also shows a limitation when texts are in a special domain for semantic similarity calculation (e.g., questions in TREC).

We also present the performances of direct inference based on `LLaMA-2-7b-hf`, which is generally outperformed by the small LMs fine-tuned on datasets generated by LLMs. This result is consistent with the discovery that LLMs are better generators than discriminators (Dai et al., 2023). However, this requires the LLM generator to be aware of all labels to avoid the ignorance of label interdependency. Otherwise, the generator might underperform direct prompting LLM as shown in the comparison between ZeroGen++ (ProGen++) with direct prompting.

Finally, we evaluate the ICL-based Incubator with GPT-4 as the backbone model. With a significantly larger amount of parameters, Incubator with GPT-4 outperforms the one based on LLaMA-2. This indicates larger backbone models can further scale up the performance of our Incubator. Also, tunable models can benefit from self-diversification to narrow the gap between the close-source GPT-4, which can also be improved once it becomes open-source for fine-tuning.

Method	**Emotion**	**NYT-LOC**	**Massive**
Prompting	43.15	62.11	57.67
ZeroGen++	52.65	69.27	56.46
ProGen++	52.80	69.64	57.16
Incubator (Ours)	**56.00**	**84.19**	**68.36**
- Diversification	55.00	76.39	61.53
Incubator w/ GPT-4	53.40	78.36	73.84

Table 2: Results on datasets with the "Other" class.

4.5 Label "Other" Results

We present the experiment results on datasets with miscellaneous (label "Other") in Table 2. The awareness of the miscellaneous category is important for classification (Li et al., 2021a), especially when limited labels are known in a large corpus. For ZeroGen or ProGen, we use the label name "*Other than ... (other labels)*" to prompt for generation. We can observe a significantly larger gap between the Incubator and the label interdependency-agnostic methods, which shows the advantage of Incubator on datasets with miscellaneous. Furthermore, the self-diversification shows a more prominent improvement in perfor-

Target	TED Summary	Target	Steam Game	Target	Text Message
"About AI"	100%/100%	"Action"	90%/90%	"Positive"	98%/98%
"About Climate"	100%/100%	"RTS"	74%/77%	"Request"	97%/98%
"By Educator"	94%/94%	"Card"	100%/100%	"About Food"	98%/98%
"Funny"	75%/80%	"Relaxing"	100%/100%	"Work-related"	83%/86%

Table 3: Precision@100 (GPT-4 Evaluation/Human Evaluation) of incubated retrievers on unannotated corpora.

Target	ZeroGen	ProGen	Incubator
"About AI"	96%/96%	97%/97%	**100%/100%**
"About Climate"	98%/98%	98%/98%	**100%/100%**
"By Educator"	82%/85%	87%/88%	**94%/94%**
"Funny"	63%/66%	68%/72%	**75%/80%**

Table 4: Performance comparison of incubated retrievers on unannotated corpora.

mance. This phenomenon can be attributed to the requirement for a more diverse generation by the miscellaneous category.

4.6 Complicated Class Definition Results

We further showcase how Incubator can be applied to satisfy personal demands, such as mining items preferred by an individual. For each raw corpus, we propose four attributes a user might be interested in, such as "About AI" for TED Talks. For each attribute, we create an instruction to build a text classifier with two labels: the target attribute and the miscellaneous label "Other". We use the incubated classifier to score each raw text and select the texts with the top scores. For evaluation, we ask GPT-4 and humans whether the mined texts satisfy the demand with Precision@100 as the metric. The human evaluation for each result is done by 3 professional human annotators and keeps the majority decision.

The text mining performance is presented in Table 3. Incubator incubates strong text miners with generally high precision on all setups. Remarkably, we achieve nearly or exactly 100% precision on several targets. Moreover, our miners are validated to be able to handle different text domains, enabling a broad application of our Incubator.

In Table 4, we further compare the incubation performance on complicated classes between Incubator and baselines. The presented result is consistent with previous ones, which further verifies the benefit of Incubator for personalized classifier incubation.

4.7 Incubation with Logical Conjunction

We further showcase how to utilize Incubator to satisfy more complicated user demands. We increase the label complexity by adding logical conjunctions into labels, that are "and" ($\wedge$), "or" ($\vee$), and "not" ($\neg$). The logical conjunctions represent a finer-grained demand from the user. For instance, one may want to search for texts that are "Positive and about food", as "Positive" $\wedge$ "About food".

To realize such finer-grained text mining, we utilize the maneuverability of Incubator to incubate multiple text miners and combine their scores with logical probabilistic calculations as follows,

- $P(L_A \wedge L_B) = P(L_A)P(L_B)$
- $P(L_A \vee L_B) = P(L_A) + P(L_B) - P(L_A \wedge L_B)$
- $P(\neg L_A) = 1 - P(L_A)$

where L_A, L_B are two labels used as the targets for the incubation. Here we view the labels as independent for simplification. We use the Text Message corpus for text mining. For evaluation, we keep the previous scenario unchanged. We compare two types of incubation scenarios,

- **Direct Incubation** Incubator only incubates one text miner with the full label name, such as "Positive and about food".
- **Conjunctive Incubation** first decomposes the label name into multiple ones with corresponding conjunctions, like decomposing "Positive and about food" into "Positive" $\wedge$ "About food". Then the score is calculated based on logical probabilistic calculations.

The experiment results are presented in Table 5. Conjunctive incubation generally outperforms direct incubation, which shows the benefit of this strategy. As conjunctive incubation also shows strong capability on three logical variables, this shows how Incubator can be customized to more complex settings.

4.8 Case Studies on Generated Training Data

To more concretely demonstrate the intermediate processes in the incubation, we launch a study on the generated texts from the Incubator for classifier incubation. We demonstrate the generated training data for data mining in the text message corpus in Table 6. For each column, there is a piece of text generated with the target value and the other one in the same Python dictionary with the miscellaneous

Logic	Target	Direct Incubation	Conjuctive Incubation
$L_1 \wedge L_2$	"Positive and about food"	85%/85%	**88%/88%**
$L_1 \vee L_2$	"Positive or negative"	99%/99%	**100%/100%**
$L_1 \wedge \neg L_2$	"Positive but not excited"	74%/72%	**89%/86%**
$L_1 \wedge L_2 \wedge L_3$	"Positive, about food, and with dish name"	40%/43%	**84%/85%**

Table 5: The performance of incubated retrievers with logical conjunctions.

Target	Generated data with target label	Generated data with misc label
"Positive"	Hey! I love the new update. It's awesome! Wow, you got the tickets for our dream holiday! I absolutely love the new design of the app.	Just checking in on the progress of the project. I've booked the flights for next week. I'm having trouble logging into my account.
"Request"	Can you send me the report by end of today? Could you please bring me a coffee? Can you pass me the salt?	What did you do during the weekend? How was your day? Hey, did you catch the game last night?
"About food"	The pizza at Mario's is the best in town! I'm craving for a burger and fries! I just tried that new sushi place. Totally worth it!	I have an important meeting at 10am tomorrow. I might go for a run later. Hey, what time does the movie start?
"Work-related"	We need to finalize the report by tomorrow. The meeting is scheduled at 3 PM tomorrow. The project deadline has been extended.	Hey, do you want to catch a movie tonight? Do you want to catch up for dinner tonight? Hey! What are you up to this weekend?

Table 6: The performance of incubated retrievers with logical conjunctions.

label "Other".

The most straightforward observation is the generated data correctly follows the label, which guarantees the foundational precision of the incubated classifiers. Also, the generated texts incorporate a wide range of syntactic structures and semantic contents for the training data diversity. For the miscellaneous label, we can observe the Incubator to cover various potential negative labels. For instance, the miscellaneous category for "About food" includes labels such as "About meeting", "About sports", "About movie", which broadens the negative set understood by the incubated classifier.

Finally, we can view some attribute correlations between the data in the same generated Python dictionary. In the "Positive" example, the three samples have the same topic "Project", "Travel", and "App". With these data different in the target attribute but same in other attributes, the incubated classifier can better focus on the target attribute and eliminate spurious correlations.

5 Analyses of Incubator

5.1 Incubation Dataset Size

We first adjust the number of data generated from Incubator to investigate how the incubated classifier will be affected. We conduct experiments on TREC and SNIPS datasets with incubation data size from 4 to 1024. The results are illustrated in Figure 3. From the shown scaling-up trend, there is a clear threshold (64) on the dataset size, after which the performance gained from generating more training data becomes limited. Thus, we recommend Incubator users generate at least 64 data samples for the classifier incubation.

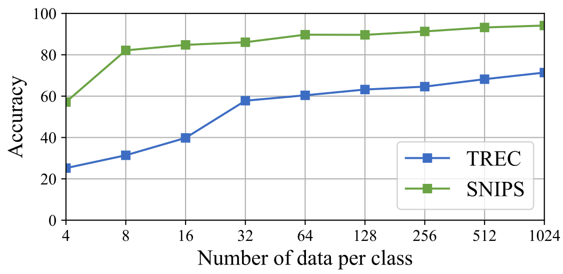

Figure 3: Incubation dataset size analysis.

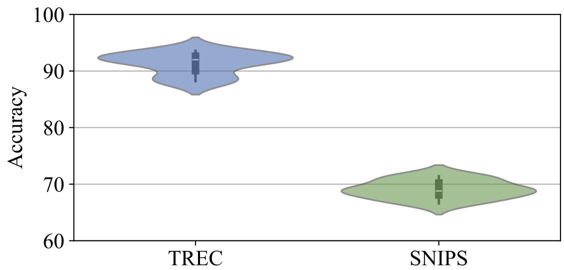

Figure 4: Analysis of Incubator instruction robustness.

5.2 Instruction Robustness

We then check the robustness of Incubator to instructions by testing with different but semantically equal instructions. We rephrase each instruction for TREC and SNIPS into 10 different versions and then run the incubation pipeline for evaluation. The robustness evaluation is presented in Figure 4. We can observe the lexical and syntactical attributes, which are changed in the rephrasing, have limited

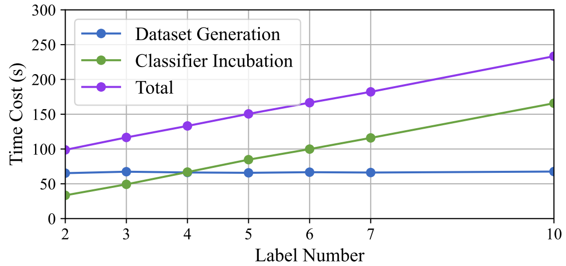

Figure 5: Efficiency Analysis of Incubator instruction.

impact on the incubated result. Thus, we conclude our Incubator is robust against the variations of the same instruction.

5.3 Efficiency Analysis

We analyze the time efficiency of the Incubator to explore its efficiency in deployment. For dataset generation, we run the LLaMA model with the acceleration by the `vllm` package (Kwon et al., 2023). For the small classifier incubation, we fine-tune the model with the trainer in the `transformers` package (Wolf et al., 2019b). We evaluate the time for dataset generation and classifier incubation (fine-tuning). The time is obtained by averaging the results in experiments on the 8 traditional benchmarks, which is illustrated in Figure 5. All experiments are run on a single A100 device.

For dataset generation, the average time is 67.53s. The generation times for all benchmarks are distributed around this average since `vllm` has a fixed max length limitation for decoding. For classifier incubation, the time is almost linearly dependent on the number of labels, which shows an average of 15.16s time cost per class.

Thus, the time efficiency of our Incubator is feasible to incubate personal classifiers. Also, the main time cost happens in classifier incubation rather than calling the LLM for dataset generation, especially when the label number is large.

6 Conclusion and Future Work

In summary, this paper proposes a new framework for model incubation by querying an instruction-tuned LLM. Our model, Incubator, is pre-trained on Huggingface resources and ICL-based augmentation. The Incubator is further strengthened by a novel self-diversification technique. We show that Incubator can incubate strong classifiers for traditional benchmarks and customized text mining, following instructions. We also include comprehensive analysis to explore the properties of the Incubator for deeper insight and better application.

Limitation and Future Work

While Incubator shows strong performance in producing reliable and customized classifiers, it has some limitations that can be further improved in future works.

Instruction Effort: Current Incubator requires the user to include all label names in the instruction, which adds effort for the user to create instructions, especially when the label number is large or the user is unclear about the label names. A combination with existing work (Wang et al., 2023a) might be a direction to reduce user efforts further.

LLM Knowledge Dependence: As an LLM-only methods, the Incubator is only able to generate text within its knowledge scope. For emerging labels, the

Future work will concentrate on two tracks. 1) **Improve the incubation quality:** We can incorporate existing or new methods to improve data generation quality like higher diversity and harder negative samples. 2) **Broaden the scope of incubated models:** The incubated model can be more than classifiers, such as question responder and text summarizer. These models might require more complicated instruction understanding and other techniques for model enhancement. Incubator still has to rely on delicate explanations or in-context examples to handle them.

References

Pranjal Aggarwal, Ameet Deshpande, and Karthik R. Narasimhan. 2023. Semsup-xc: Semantic supervision for zero and few-shot extreme classification. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 228–247. PMLR.

Francesco Barbieri, José Camacho-Collados, Luis Espinosa Anke, and Leonardo Neves. 2020. Tweeteval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 1644–1650. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric

Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Daewon Chae, Nokyung Park, Jinkyu Kim, and Kimin Lee. 2023. Instructbooth: Instruction-following personalized text-to-image generation. *CoRR*, abs/2312.03011.

Dhivya Chandrasekaran and Vijay Mago. 2022. Evolution of semantic similarity - A survey. *ACM Comput. Surv.*, 54(2):41:1–41:37.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *CoRR*, abs/1805.10190.

Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Yihan Cao, Zihao Wu, Lin Zhao, Shaochen Xu, Wei Liu, Ninghao Liu, Sheng Li, Dajiang Zhu, Hongmin Cai, Lichao Sun, Quanzheng Li, Dinggang Shen, Tianming Liu, and Xiang Li. 2023. Auggpt: Leveraging chatgpt for text data augmentation. *Preprint*, arXiv:2302.13007.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society, AIES 2018, New Orleans, LA, USA, February 02-03, 2018*, pages 67–73. ACM.

Xuan Long Do, Bowei Zou, Shafiq R. Joty, Anh Tran Tai, Liangming Pan, Nancy F. Chen, and Ai Ti Aw. 2023. Modeling what-to-ask and how-to-ask for answer-unaware conversational question generation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 10785–10803. Association for Computational Linguistics.

Chengyu Dong, Zihan Wang, and Jingbo Shang. 2023a. Debiasing made state-of-the-art: Revisiting the simple seed-based weak supervision for text classification. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 483–493. Association for Computational Linguistics.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. 2023b. A survey on in-context learning. *Preprint*, arXiv:2301.00234.

Xiao Fang, Shangkun Che, Minjia Mao, Hongzhe Zhang, Ming Zhao, and Xiaohang Zhao. 2023. Bias of ai-generated content: An examination of news produced by large language models. *CoRR*, abs/2309.09825.

Jack FitzGerald, Christopher Hench, Charith Peris, Scott Mackie, Kay Rottmann, Ana Sanchez, Aaron Nash, Liam Urbach, Vishesh Kakarala, Richa Singh, Swetha Ranganath, Laurie Crist, Misha Britan, Wouter Leeuwis, Gokhan Tur, and Prem Natarajan. 2022. Massive: A 1m-example multilingual natural language understanding dataset with 51 typologically-diverse languages. *Preprint*, arXiv:2204.08582.

Isabel O. Gallegos, Ryan A. Rossi, Joe Barrow, Md. Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K. Ahmed. 2023. Bias and fairness in large language models: A survey. *CoRR*, abs/2309.00770.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6894–6910. Association for Computational Linguistics.

Qi Gou, Zehua Xia, Bowen Yu, Haiyang Yu, Fei Huang, Yongbin Li, and Cam-Tu Nguyen. 2023. Diversify question generation with retrieval-augmented style transfer. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 1677–1690. Association for Computational Linguistics.

Jiawei Han and Micheline Kamber. 2000. *Data Mining: Concepts and Techniques*. Morgan Kaufmann.

Austin W Hanjie, Ameet Deshpande, and Karthik Narasimhan. 2022. Semsup: Semantic supervision for simple and scalable zero-shot generalization. *arXiv preprint arXiv:2202.13100*.

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2023. Large language models are reasoning teachers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 14852–14882. Association for Computational Linguistics.

Lesheng Jin, Zihan Wang, and Jingbo Shang. 2022. Wedef: Weakly supervised backdoor defense for text classification. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 11614–11626. Association for Computational Linguistics.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on*

Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894. PMLR.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP 2023, Koblenz, Germany, October 23-26, 2023*, pages 611–626. ACM.

Liunian Harold Li, Jack Hessel, Youngjae Yu, Xiang Ren, Kai-Wei Chang, and Yejin Choi. 2023. Symbolic chain-of-thought distillation: Small models can also "think" step-by-step. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 2665–2679. Association for Computational Linguistics.

Peiran Li, Fang Guo, and Jingbo Shang. 2021a. "misc"-aware weakly supervised aspect classification. In *Proceedings of the 2021 SIAM International Conference on Data Mining, SDM 2021, Virtual Event, April 29 - May 1, 2021*, pages 468–476. SIAM.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Zichao Li, Dheeraj Mekala, Chengyu Dong, and Jingbo Shang. 2021b. Bfclass: A backdoor-free text classification framework. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 444–453. Association for Computational Linguistics.

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.

Ilya Loshchilov and Frank Hutter. 2017. SGDR: stochastic gradient descent with warm restarts. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Dheeraj Mekala, Chengyu Dong, and Jingbo Shang. 2022. LOPS: learning order inspired pseudo-label selection for weakly supervised text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 4894–4908. Association for Computational Linguistics.

Meta. 2024. Meta llama-3.

Elena Mozzherina. 2013. An approach to improving the classification of the new york times annotated corpus. In *Knowledge Engineering and the Semantic Web - 4th International Conference, KESW 2013, St. Petersburg, Russia, October 7-9, 2013. Proceedings*, volume 394 of *Communications in Computer and Information Science*, pages 83–91. Springer.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Letian Peng, Yuwei Zhang, and Jingbo Shang. 2023. Generating efficient training data via llm-based attribute manipulation. *CoRR*, abs/2307.07099.

Letian Peng, Yuwei Zhang, Zilong Wang, Jayanth Srinivasa, Gaowen Liu, Zihan Wang, and Jingbo Shang. 2024. Answer is all you need: Instruction-following text embedding via answering the question. *CoRR*, abs/2402.09642.

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, Brussels, Belgium. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL.

Yisheng Song, Ting Wang, Puyu Cai, Subrota K. Mondal, and Jyoti Prakash Sahoo. 2023. A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities. *ACM Comput. Surv.*, 55(13s):271:1–271:40.

Xiaofei Sun, Xiaoya Li, Jiwei Li, Fei Wu, Shangwei Guo, Tianwei Zhang, and Guoyin Wang. 2023. Text classification via large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 8990–9005. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *CoRR*, abs/2212.03533.

Tianle Wang, Zihan Wang, Weitang Liu, and Jingbo Shang. 2023a. Wot-class: Weakly supervised open-world text classification. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM 2023, Birmingham, United Kingdom, October 21-25, 2023*, pages 2666–2675. ACM.

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, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 3043–3053. 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. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 3944–3962. Association for Computational Linguistics.

Zihan Wang, Tianle Wang, Dheeraj Mekala, and Jingbo Shang. 2023c. A benchmark on extremely weakly supervised text classification: Reconcile seed matching and prompting approaches. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 3944–3962. Association for Computational Linguistics.

Zilong Wang and Jingbo Shang. 2022. Towards few-shot entity recognition in document images: A label-aware sequence-to-sequence framework. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 4174–4186.

Peter West, Chandra Bhagavatula, Jack Hessel, Jena D. Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2022. Symbolic knowledge distillation: from general language models to commonsense models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 4602–4625. Association for Computational Linguistics.

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

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

Xinnuo Xu, Guoyin Wang, Young-Bum Kim, and Sungjin Lee. 2021. Augnlg: Few-shot natural language generation using self-trained data augmentation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 1183–1195. Association for Computational Linguistics.

Jiacheng Ye, Jiahui Gao, Qintong Li, Hang Xu, Jiangtao Feng, Zhiyong Wu, Tao Yu, and Lingpeng Kong. 2022a. Zerogen: Efficient zero-shot learning via dataset generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 11653–11669. Association for Computational Linguistics.

Jiacheng Ye, Jiahui Gao, Zhiyong Wu, Jiangtao Feng, Tao Yu, and Lingpeng Kong. 2022b. Progen: Progressive zero-shot dataset generation via in-context feedback. In *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages

3671–3683. Association for Computational Linguistics.

Wenqian Ye, Guangtao Zheng, Xu Cao, Yunsheng Ma, Xia Hu, and Aidong Zhang. 2024. Spurious correlations in machine learning: A survey. *CoRR*, abs/2402.12715.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2023a. Instruction tuning for large language models: A survey. *CoRR*, abs/2308.10792.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

Yu Zhang, Bowen Jin, Xiusi Chen, Yanzhen Shen, Yunyi Zhang, Yu Meng, and Jiawei Han. 2023b. Weakly supervised multi-label classification of full-text scientific papers. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2023, Long Beach, CA, USA, August 6-10, 2023*, pages 3458–3469. ACM.

Xuandong Zhao, Siqi Ouyang, Zhiguo Yu, Ming Wu, and Lei Li. 2023. 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), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 15590–15606. Association for Computational Linguistics.

Wenxuan Zhou, Sheng Zhang, Yu Gu, Muhao Chen, and Hoifung Poon. 2023. Universalner: Targeted distillation from large language models for open named entity recognition. *CoRR*, abs/2308.03279.

A Hyperparameter

Hyperparameter	Instruction-tuning	Incubation
Initial LR	2×10^{-5}	1×10^{-5}
Batch Size	16	32
Epoch	3	8

Table 7: The hyperparameter setups in our experiments.

B Instruction-tuning Dataset Processing

Dataset: app_reviews

Description: It is a large dataset of Android applications belonging to 23 different apps categories, which provides an overview of the types of feedback users report on the apps and documents the evolution of the related code metrics. The dataset contains about 395 applications of the F-Droid repository, including around 600 versions, 280,000 user reviews (extracted with specific text mining approaches)

Instruction: Please create a model to anticipate the star rating to Android application reviews.

Data: {"1 star": …, "2 star": …, "3 star": …, "4 star": …, "5 star": …, }

Figure 6: A case in our instruction-tuning dataset for Incubator.

C Dataset Generation Prompt

Role	**Message**
User	Generate an imaginative instruction to build a text classifier and its corresponding samples.
GPT-4	"Input": "Instruction 1" "Output": {"Label 1,1": "Data 1,1", "Label 1,2": "Data 1,2", ...}
User	Generate an imaginative instruction to build a text classifier and its corresponding samples.
GPT-4	"Input": "Instruction 2" "Output": {"Label 2,1": "Data 2,1", "Label 2,2": "Data 2,2", ...}
User	Generate an imaginative instruction to build a text classifier and its corresponding samples.

Table 8: The prompt used in ICL-based augmentation.

D Revised Dataset with Miscellaneous

Dataset	Label	Other
Emotion	Joy, Sadness	Love, Anger, Fear, Surprise
NYT-LOC	America, Iraq, Japan, China	Britain, German, Canada, France, Russia, Italy
Massive	Calendar, Play, QA, Email, IoT, Weather, Transport	Lists, News, Recommendation, Datetime, Social, Alarm, Music, Audio, Takeaway, Cooking

Table 9: The revision on datasets for the label "Other".

As shown in Table 9, the minor categories with low proportion are merged together to an "Other" class.

E Source of Metadata

The datasets used to create instruction-tuning metadata are listed in Table 10.

Dataset	Label
YELP (P)	Review Sentiment
YELP (S)	Review Star
IMDB	Review Sentiment
Rotten Tomatoes	Review Sentiment
Twitter Financial News (S)	News Sentiment
Twitter Financial News (T)	News Topic
Yahoo	Question Category
Subj	Subjectiveness
Student Question	Question Category
Financial Benchmark	News Sentiment
Amazon (C)	Counterfactual
Amazon (S)	Review Sentiment
APP Review	Review Sentiment
Toxic Conversation	Toxicity
ETHOS	Toxicity
HATE	Toxicity
MASSIVE (T)	Request Topic
MASSIVE (I)	Request Intent
SNLI	Natural Language Inference
MNLI	Natural Language Inference
QNLI	Natural Language Inference
WNLI	Natural Language Inference
RTE	Natural Language Inference
QQP	Semantic Similarity
MRPC	Semantic Similarity

Table 10: The revision on datasets for the label "Other".