Active Learning for BERT: An Empirical Study

Liat Ein-Dor[*], **Alon Halfon**[*], **Ariel Gera**[*], **Eyal Shnarch**[*], **Lena Dankin, Leshem Choshen**
Marina Danilevsky, Ranit Aharonov, Yoav Katz and Noam Slonim
IBM Research

{liate, alonhal, arielge, eyals, lenad, leshem.choshen, katz, noams}@il.ibm.com,
mdanile@us.ibm.com, ranit.aharonov2@ibm.com

Abstract

Real world scenarios present a challenge for text classification, since labels are usually expensive and the data is often characterized by class imbalance. Active Learning (AL) is a ubiquitous paradigm to cope with data scarcity. Recently, pre-trained NLP models, and BERT in particular, are receiving massive attention due to their outstanding performance in various NLP tasks. However, the use of AL with deep pre-trained models has so far received little consideration. Here, we present a large-scale empirical study on active learning techniques for BERT-based classification, addressing a diverse set of AL strategies and datasets. We focus on practical scenarios of binary text classification, where the annotation budget is very small, and the data is often skewed. Our results demonstrate that AL can boost BERT performance, especially in the most realistic scenario in which the initial set of labeled examples is created using keyword-based queries, resulting in a biased sample of the minority class. We release our research framework, aiming to facilitate future research along the lines explored here.

1 Introduction

Automatic text classification is a well studied problem in Natural Language Processing (NLP), with great practical importance and numerous real world applications (Aggarwal and Zhai, 2012). There are two major hurdles to developing effective text classifiers in practice, as well as to developing classifiers in other domains – the lack of labeled data, and class imbalance (Japkowicz and Stephen, 2002). Text classifiers often require high quantities of labeled data for model training. However, collecting such labeled data is a notoriously expensive and time-consuming process, and shortage of labeled data is exacerbated when the desired class has a relatively low prior in the data. In such a scenario, even going through the burden of labeling a random sample may yield an insufficient number of positive instances to properly train a classifier. Our focus in this work is on this challenging coupled setup, frequently encountered by real-world users – where labeled data is scarce *and* the prior of the desired class is small.

A classical approach for coping with limited annotation resources is Active Learning (AL) (Cohn et al., 1996). In this paradigm, one assumes that unlabeled data are abundant, and the goal is to focus the expensive labeling process on the most informative instances. Many AL strategies have been proposed, aiming to minimize the labeling burden, or if taken from a different perspective – maximize the value of labeling a small set of examples. Importantly, the usefulness of an AL strategy naturally depends on the classification scheme with which it is coupled. A successful AL approach for a Naive Bayes classifier may not be that effective for a modern deep-learning algorithm such as CNN, and vice versa.

A more recent relevant development is the introduction of pre-trained NLP models (cf. Qiu et al., 2020), which have been shown to substantially improve state-of-the-art results in numerous NLP tasks. A prominent example is the BERT model (Devlin et al., 2018), which has received massive attention from the NLP research community since its inception. However, the use of AL with deep pre-trained models for text classification – and BERT in particular – has so far received surprisingly little consideration. Thus, while recent papers have demonstrated the value of AL for various deep-learning text classification schemes (Shen et al., 2017; Zhang et al., 2017; Siddhant and Lipton, 2018; Prabhu et al., 2019), the potential of AL combined with BERT is yet to be explored. First, given the unique properties of pre-trained models,

[*]These authors equally contributed to this work.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 7949–7962,
November 16–20, 2020.

and the expectation that such models will yield adequate performance even with small amounts of training data, it is unclear *a priori* whether – and to what extent – established AL paradigms can further enhance their classification performance. Moreover, more recent Deep AL strategies, such as Core-Set (Sener and Savarese, 2017) and Dropout (Gal and Ghahramani, 2016), were developed in the vision domain for CNNs. The value of these strategies on top of the BERT transformer architecture remains unclear.

Our goal in this work is threefold. We study the potential of (i) various AL strategies; (ii) in conjunction with BERT, an arguably outstanding text classification scheme; (iii) within a highly challenging – yet common – real-world scenario of class imbalance and scarce labeled data. To address this goal, we conduct a systematic study, considering traditional and advanced AL strategies coupled with BERT for a wide range of datasets. We focus on three scenarios: A *balanced* setting, serving as a reference, where the prior of the class of interest is not too small; the more challenging *imbalanced* setting, where the class prior is $\leq 15\%$ but we assume a way to obtain an unbiased set of positive samples to be used for initial training; and finally, the *imbalanced-practical* setting, which is similar to the imbalanced one, but takes a step further towards a truly practical setup, in which there is no access to an unbiased positive sample. Instead, we assume the user has access to a *biased* sample, hopefully enriched with positive examples, obtained by issuing simple queries of keywords associated with the positive class.

Our results convey that AL strategies can boost BERT performance, under the challenging setting of a small annotation budget and highly skewed data, especially in the more practical real-world settings. We release our research framework[1] , including access to all datasets, an implementation of multiple AL strategies, and an associated automatic evaluation framework, aiming to facilitate further research along the lines explored here.

2 Related Work

AL has been widely used in many fields to successfully decrease the labeling effort involved in the training process. A good summary of active learning works prior to the advances in deep learning can be found in Settles (2009). Advances in deep learning have given rise to extensive research into deep active learning, which aims to adapt the classic AL framework to the special properties of DNNs. Deep AL presents some specific challenges. Since DNNs are computationally heavy, training a new model whenever a single training sample is added is highly impractical. This requires a shift to batch mode active learning, where a batch of examples is queried at every iteration. Moreover, the tendency of the softmax layer to over-confidence has led to the development of various uncertainty-based strategies tailored to the special properties of DNNs (Gal and Ghahramani, 2016).

Most of the works in deep active learning focus on image classification with convolutional neural networks (Sener and Savarese, 2017; Gal and Ghahramani, 2016; Gissin and Shalev-Shwartz, 2019). Recent papers have demonstrated the value of deep active learning for text classification (Zhang et al., 2017; Siddhant and Lipton, 2018; Prabhu et al., 2019; Lowell et al., 2018), but in general did not study AL for BERT. One exception is Zhang and Zhang (2019) who applied an ensemble of AL strategies to BERT for the task of intent classification. However, this work focuses on a single task, and does not address the effect of small and imbalanced data. Additionally, Shelmanov et al. (2019) and Liu et al. (2020) focused on particular variants of BERT (BioBERT and BERT-CRF) and studied a single or two specific tasks, with a small collection of AL strategies. To the best of our knowledge, this work is the first to systematically explore advanced strategies like Core-Set (Sener and Savarese, 2017), Dropout (Gal and Ghahramani, 2016), Expected Gradient Length (Huang et al., 2016) and Discriminative Active Learning (Gissin and Shalev-Shwartz, 2019) for BERT, in various settings and a diversity of tasks.

3 Empirical Evaluation

3.1 Data

We consider 10 datasets (see Table 1) that cover a variety of domains, and for each we select one target class as our classification goal, thus creating a set of binary classification tasks. Three datasets are originally skewed, i.e., the target class prior is $\leq 15\%$: Wiki Attack (Wulczyn et al., 2017), which annotates Wikipedia discussions for offensive con-

[1] https://github.com/IBM/low-resource-text-classification-framework

No.	Dataset	Size	Class	Prior
1	Subjectivity-imb	5,556	subjective	10%
2	Polarity-imb	5,923	positive	10%
3	AG's News-imb	17,538	world	10%
4	Wiki attack	21,000	general	12%
5	ISEAR	7,666	fear	14%
6	TREC	5,952	location	15%
7	AG's News	21,000	world	25%
8	CoLA	9,594	unacceptable	30%
9	Subjectivity	10,000	subjective	50%
10	Polarity	10,662	positive	50%

Table 1: Dataset details: size, target (positive) class, and its prior in the dataset.

tent;[2] ISEAR (Shao et al., 2015), which annotates personal reports for emotion; and TREC (Li and Roth, 2002) which considers the answer type of questions. In four datasets the target class prior is $20\% - 50\%$: AG's News (Zhang et al., 2015), which categorizes news articles; CoLA (Warstadt et al., 2018), which annotates sentences for grammatical acceptability; Subjectivity (Pang and Lee, 2004), which classifies movie snippets into subjective or objective; and Polarity (Pang and Lee, 2005), which includes sentiment analysis on movie reviews. In addition, we enriched the imbalanced datasets by creating imbalanced versions of three balanced datasets via sub-sampling the target class instances towards a prior of 10% (Table 1, rows 1–3).

Each dataset was split into train, dev, and test sets, keeping the original split, if exists, and otherwise applying a $70\%/10\%/20\%$ split, respectively. For large datasets, we limit the sizes to $15K/3K/3K$ respectively by randomly sampling from each set. The complete details along with links to all datasets are provided in Appendix A.

3.2 Experimental setup

We apply pool-based active learning (Settles, 2009) in batch mode, using BERT as the classification model. Seven selection strategies are examined over the 10 fully labeled binary classification datasets described above. The use of fully labeled datasets enables simulating manual labeling (Yang and Loog, 2018). Per dataset, we use its train set as the initial *pool* of examples from which instances are selected for labeling.

We assume an initial annotation budget that enables labeling 100 examples, used to create an initial seed L. In some setups, L may contain additional instances without their ground truth labels, and in general the way L is selected depends on the experimental scenario, as described below. We denote by U the instances in the pool that do not belong to L.

For a given AL strategy, a single experiment starts with the seed L, used to train BERT as the initial classifier (iteration 0). Next, we conduct 5 iterations. In each, the AL strategy selects a batch of 50 unlabeled instances from U that are added to L along with their true labels, and BERT is trained over these expanded data. Note, the BERT fine-tuning in each iteration is done from scratch, to avoid overfitting data from previous rounds, as suggested in Hu et al. (2018). In each experiment, all AL strategies start with the same initial seed. The reported results are the average over 5 different experiments, i.e., 5 different initial seeds.

For each AL strategy, we consider the following three scenarios:

Balanced: Here, the positive class prior is not very low, hence a randomly selected sample is expected to have a sufficient number of positive examples. Correspondingly, the seed L is simply defined as 100 instances sampled at random from the pool. We apply this scenario to datasets with $20\% - 50\%$ of positive labels.[3]

For datasets with a positive class label $\leq 15\%$, a random seed of 100 instances led to unstable BERT runs (data not shown), presumably due to the combination of small and highly skewed training data resulting from such random selection. Hence, for these imbalanced datasets, we consider the two scenarios described below. In both cases, we expand the initial set of 100 labeled examples with another set of 100 instances, selected at random from the remaining data, which are all added to L with a negative label. In other words, the low prior of the positive class naturally implies high prior of the negative class, enabling to expand the fully labeled 100 instances with an additional set of 100 instances that are – weakly – labeled as negative examples (without the need for additional annotation budget). Hence, in both scenarios described below, L contains a total of 200 examples.

Imbalanced: Here, the 100 fully labeled examples are drawn at random from the positive examples in the dataset, hence all 100 are indeed positive, and

[2] This data set contains offensive language. IBM abhors use of such language and any form of discrimination.

[3] Although not all these datasets are strictly balanced, we chose this name for brevity of presentation.

are further an unbiased sample of the positive class. In this setting we assume high-precision heuristics that enable generating a relatively unbiased sample; but in many real-world cases such heuristics may not exist, or are expected to have limited coverage and would not enable sampling at will[4]. Thus, such heuristics cannot be assumed to yield a large training set, but may nevertheless be used for obtaining a small initial seed in an active learning setting.

Imbalanced-practical: In this scenario, we simulate a more realistic setting in which a user attempts to obtain as many positive examples as possible using the budget of 100 annotations. To this end, we design a simple keyword-based query for each dataset, which aims to retrieve a set of instances enriched with positive examples, using words assumed to be associated with the positive class. We opted for keyword-based queries as they are often used in practice in real-world scenarios. We apply the query to the pool, and randomly draw 100 instances from the query result, which are then added to the seed with their ground truth labels. Note that these 100 examples are expected to be enriched with positive examples, yet in a biased manner, since by construction, all examples match the query we started with. Specifically, this scenario was tested on four datasets for which a simple string (or sub-string) match query with enough hits could be defined: (i) for the *fear* class in ISEAR the query is [*fear* or *afraid* or *scared* or *scary*] (*fear*, for example, can also capture *fearsome*); (ii) for the TREC *location* class, [*Where* or *countr* or *cit*] (matching *cit* captures both the singular and plural of city); (iii) for the Wiki *attack* class, [*[A-Z]!*] (capturing a word ending with an upper case letter which is immediately followed by an exclamation mark, e.g., *IDIOT!*), and (iv) for the AG's Newsimb *world* class the query is a list of countries and territories separated by 'or'. It is likely that better queries could be defined. However, our goal here was to simulate a realistic setting in which a user relies on a relatively simple heuristic, and to examine the behavior of AL with BERT when initiated with a potentially biased seed.

3.3 Active Learning Strategies

We consider several AL strategies for choosing the batch of 50 instances to label in each iteration. In addition, as a baseline, we consider a **Random** strategy, where batch instances are chosen at random from the unlabeled set.

- **Least Confidence** (**LC**, Lewis and Gale, 1994): selects instances for which the model is least certain according to the max-entropy decision rule.
- **Monte Carlo Dropout** (**Dropout**, Gal and Ghahramani, 2016): Similar to LC, but instance uncertainty is calculated using Monte Carlo Dropout on 10 inference cycles, with the max-entropy acquisition function[5].
- **Perceptron Ensemble** (**PE**): Selects instances with highest uncertainty – similarly to LC – but averaging over an ensemble of models. Here, we use a light-weight ensemble strategy to overcome the unrealistic computational cost required for training an ensemble of BERT models. PE is composed of 10 perceptrons which are trained to solve the original task using L, where the perceptron inputs are the CLS vectors of the fine-tuned BERT model.
- **Expected Gradient Length** (**EGL**, Huang et al., 2016): selects instances with the largest expected gradient norm, as they are expected to wield a large influence on the model. The expectation is computed over the posterior distribution of labels for the example according to the trained model.
- **Core-Set** (Sener and Savarese, 2017): selects instances that best cover the dataset in the learned representation space (CLS), using the greedy method described in Sener and Savarese (2017).
- **Discriminative Active Learning** (**DAL**, Gissin and Shalev-Shwartz, 2019): This approach aims to select instances that make L most representative of the entire pool. We follow the exact method used in Gissin and Shalev-Shwartz (2019).

We chose these strategies as spanning the leading state-of-the-art approaches in the AL domain: uncertainty-sampling (LC and Dropout), uncertainty-sampling using ensemble methods (PE), expected model change (EGL), and diversity sampling (DAL and Core-Set).

3.4 Implementation Details

Overall, the results presented here consist of 2,520 fine-tuning experiments (14 dataset-scenario combinations $\times$ 5 initial seeds $\times$ (1 base model + (7 selection strategies $\times$ 5 iterations)). In order to run multiple experiments in parallel, experiments were performed on Intel® Xeon CPU E5-2699 v4

[4]For instance, for the task of classifying emotional situations, the prefix "This is a situation where I felt afraid:" indicates that the following sentence belongs to the *fear* class, but is expected to be rare within the corpus.

[5]other functions were shown to yield similar results (Gissin and Shalev-Shwartz, 2019)

Random	LC	Dropout	EGL	Core-Set	DAL	PE
< 1	84	840	1106	98	167	370

Table 2: Runtimes (in seconds) for a single iteration for different AL strategies, assuming 7,000 unlabeled examples.

@ 2.20GHz, with 88 CPUs and 748 GB of RAM. BERT training and inference were performed on Nvidia® Tesla K80 GPUs (single GPU per run).

Table 2 lists AL batch selection runtimes for different AL strategies. Runtimes for all strategies except Random are dominated by BERT inference, as BERT model outputs are used in selecting batch instances. Notably, two strategies demand longer inference times: EGL due to the gradient calculation, and Dropout due to the larger number of inference cycles ($\times 10$).

3.5 BERT Training Details

In each fine-tuning run, $\text{BERT}_{\text{BASE}}$ (110M parameters) was trained for 5 epochs, using a learning rate of 5×10^{-5}, and keeping the best model based on its performance on the dev set. In practice, dev sets may be unavailable, particularly under a limited annotation budget. Using a dev set to reduce variance and noise between runs helps stabilize the results, but importantly, we verified that ignoring the dev data and setting a constant number of epochs yields qualitatively similar, albeit noisier, results. Our experiments showed that increasing the batch size had a substantial effect on improving the stability of BERT results. However, due to memory limitations of the GPU, increasing the batch size comes at the expense of the maximal sequence length. We empirically determined that setting the batch size to 50, and the maximal sequence length to 100 tokens (after WordPiece tokenization), yielded the best results. We otherwise used the default settings in the TensorFlow implementation of BERT.

3.6 AL Research Framework

Our open-source framework allows a user to experiment with the active learning strategies in (§3.3) and evaluate their performance over the datasets in (§3.1). The framework also supports adding new AL strategies, making it easy to evaluate their potential.

Strategy	Balanced	Imbalanced	Imbalanced practical
Core-Set	10^{-2}	$< 10^{-5}$	$< 10^{-8}$
Dropout	$< 10^{-3}$	$< 10^{-8}$	$< 10^{-8}$
EGL	$---$	$< 10^{-4}$	$< 10^{-8}$
LC	$< 10^{-5}$	$< 10^{-9}$	$< 10^{-7}$
DAL	$< 10^{-2}$	$< 10^{-5}$	$< 10^{-6}$
PE	$---$	$< 10^{-2}$	$< 10^{-6}$

Table 3: Wilcoxon test p-values (after Bonferroni correction) for different AL strategies compared to Random. $---$ denotes insignificant results ($p \geq 0.05$).

4 Results

We report results for the AL strategies (§3.3) in three experimental scenarios (§3.2). Following the standard in the field, we use accuracy as the classification metric for the balanced scenario, and F1[6] for the imbalanced and imbalanced-practical scenarios, where the prior for positives is relatively low.

Figure 1 depicts the classification quality (accuracy or F1) per iteration for each dataset, for the relevant scenarios. For clarity of presentation, we only plot the Random baseline and three strategies that represent the different approaches. As can be seen in the full figure in the Appendix (Figure 3), the other strategies behave similarly.

In most datasets, all AL strategies performed better than the Random baseline, even in cases where the baseline results were already very good, e.g., AG's News and Subjectivity. Interestingly, the largest improvements were observed in the *imbalanced-practical* scenario. Here, the AL strategies improve the F1 of the Random baseline by a large margin of $4 - 8\%$ on average. These results demonstrate that AL can indeed enhance BERT results when the annotation budget is small, especially for datasets having a low prior for positive examples, as is the case in many real-world settings.

To check the significance of the differences, we calculate the Wilcoxon p-value[7] for every AL strategy compared to the Random baseline, per scenario, and perform a Bonferroni correction to adjust for the multiple strategies examined. To calculate the p-value for a strategy S per scenario, we compare the classification metric for all pairs (S_{dik}, R_{dik}) such that R is the Random baseline results, $d \in D$, where D is the set of datasets in-

[6] computed at the default threshold of 0.5

[7] We chose Wilcoxon p-value because of its non-parametric nature.

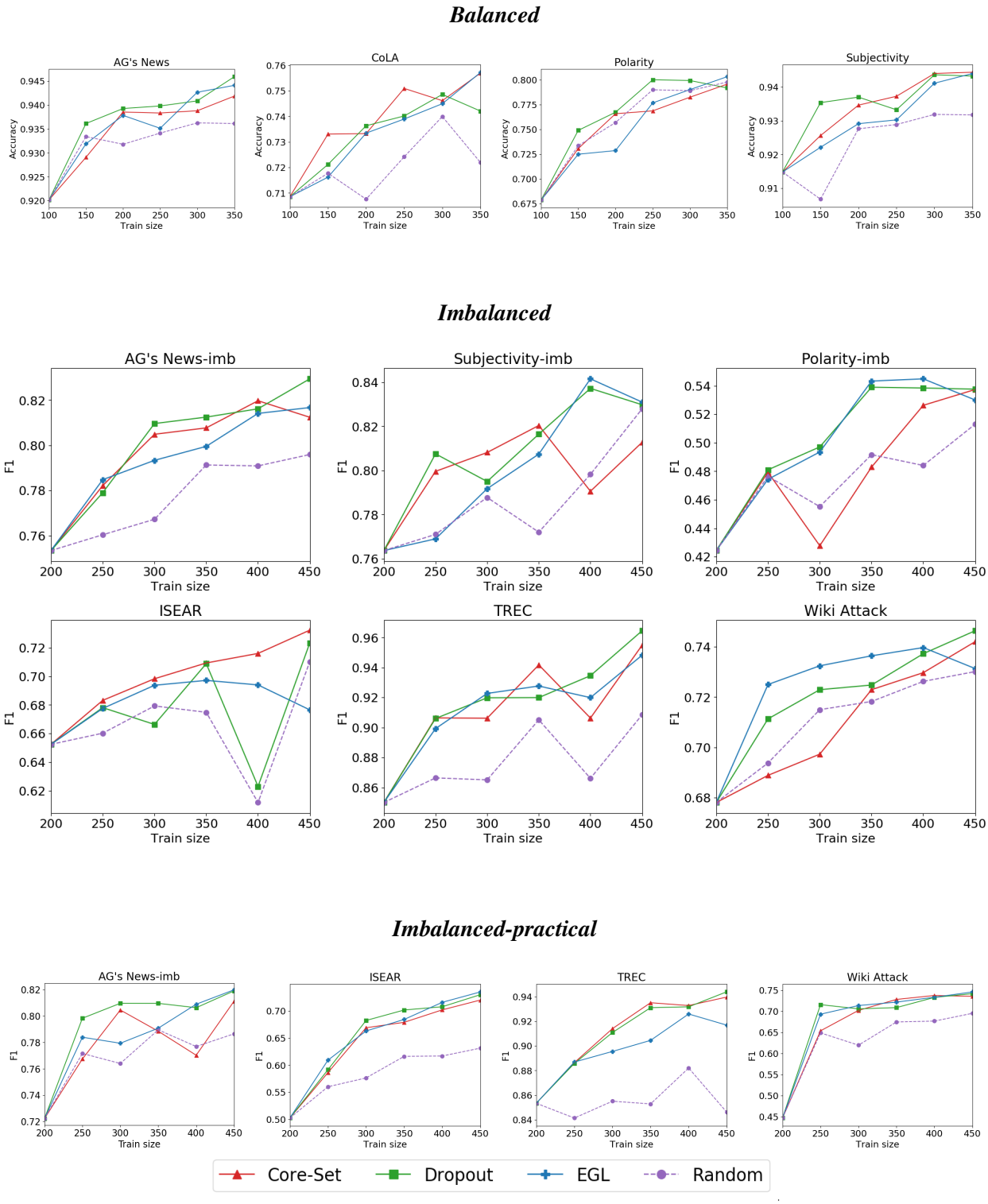

Figure 1: AL strategies compared to the Random baseline in the *balanced* (top row), *imbalanced* (two middle rows) and *imbalanced-practical* (bottom row) scenarios. Train size indicates the size of L, where each iteration adds 50 samples.

cluded in the scenario, $i = (1...5)$ is the iteration index, and $k = (1...5)$ is the experiment number. As can be seen in Table 3, all the examined AL strategies significantly and consistently outperform the Random baseline when the dataset is highly skewed (*imbalanced* and *imbalanced-practical* scenarios). All strategies except PE and EGL also outperform the baseline for the *balanced* scenario.

While AL strategies improve over the Random baseline, apparently no single strategy consistently outperforms all its counterparts. This finding echoes Lowell et al. (2018), who studied AL for text classification and sequence tagging in non-BERT models, and demonstrated the brittleness and inconsistency of AL results. For significance analysis, we calculate the p-value for every pair of strategies per scenario in a similar manner to the one described above per strategy versus Random, correcting for the multiple pairs examined, and indeed find no overall significant performance difference between any pair of AL strategies. Note, however, that some AL strategies are more efficient than others with respect to runtime - see Table 2.

As may be expected, using a seed with positive labels obtained by a query, which is naturally biased towards instances that satisfy the query, typically results in an initial model with lower F1, compared to starting with an unbiased set of positive examples (compare iteration 0 per dataset in Figure 1 between the *imbalanced-practical* and *imbalanced* scenarios). Interestingly, though, after several iterations, the AL strategies seem to bridge the gap and end up with similar classification performance in both scenarios. We further examined whether the increase in F1 for the imbalanced scenarios is driven by an increase in precision or recall. We find that in the *imbalanced-practical* scenario, the improvement in F1 is completely dominated by an increase in recall, supporting the notion that the AL strategies enable the model to extrapolate and generalize beyond the biased sample obtained by the query[8]. In contrast, in the *imbalanced* scenario the increase in F1 is mostly driven by an increase in precision. For the precision and recall curves, see Figures 4 and 5 in the Appendix.

To conclude, our analyses suggest two results that were not trivial to begin with. Applying AL to BERT can further boost the performance of this top performing model. Furthermore, even when initiated with a biased seed of positive examples – as may often occur in practice – AL strategies can swiftly generalize from this seed and significantly improve the model recall, ending up with overall strong F1 performance.

[8]The low recall of the queries can be seen in Table 5 in the Appendix.

5 Analysis

We perform a comparative analysis of the different AL strategies, aiming to better understand their relative advantages and disadvantages, and provide some insights that may lead to improved AL strategies in future work.

To enable an appropriate comparison, this analysis is performed after the initial BERT model is trained and each AL strategy has selected 50 examples for labeling. Correspondingly, all strategies select examples from the same unlabeled set U while using outputs from the same BERT model. We measure two batch properties which are known in the literature to impact AL effectiveness:

Diversity: Choosing a batch of diverse examples is often better than choosing one containing very similar and perhaps redundant examples. Following Zhdanov (2019), we define the Diversity of a set B as:

$$D(B) = \left(\frac{1}{|U|} \sum_{x_i \in U} \min_{x_j \in B} d(x_i, x_j) \right)^{-1} \quad (1)$$

where x_i denotes the representation of the [CLS] token of example i obtained by the model which was trained using L, and $d(x_i, x_j)$ denotes the Euclidean distance between x_i and x_j.

Representativeness: A known issue with AL strategies, especially the uncertainty-based ones, is their tendency to select outlier examples that do not properly represent the overall data distribution. We thus examine the representativeness of the selected batches. We rely on the KNN-density measure proposed by Zhu et al. (2008), in which the density of an example is quantified by the average distance between the example in question and its K most similar examples (i.e., K nearest neighbors) within U, based on the [CLS] representations as above. An example with high density degree is less likely to be an outlier. We define the representativeness of a batch as one over the average KNN-density of its instances using the Euclidean distance with $K = 10$.

The diversity and representativeness of the different strategies are depicted in Figure 2, where for

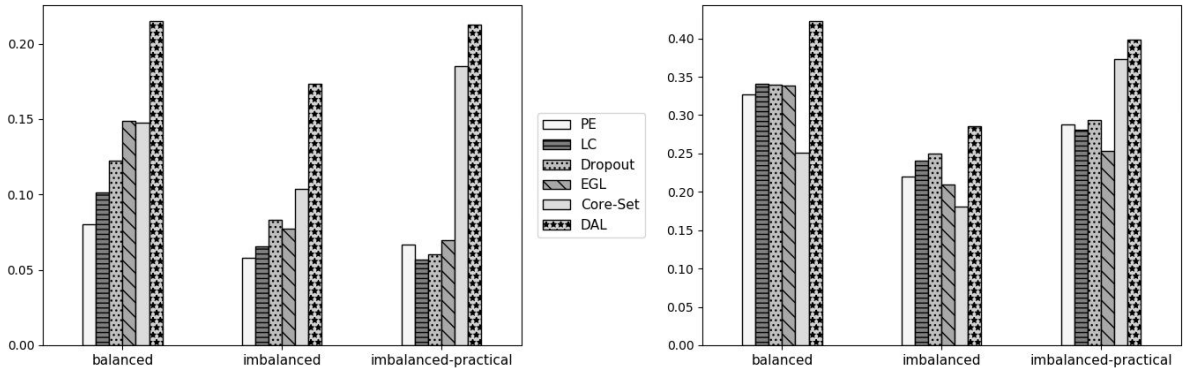

Figure 2: Diversity (left) and Representativeness (right) of the batches selected by the different AL strategies in each of the three scenarios.

each scenario we average results over all datasets and seed selections. As expected, the batch-aware strategies, DAL and Core-Set, which were designed to increase diversity, are characterized by the most diverse batches, with DAL achieving the highest diversity values, demonstrating the success of using mini-queries (Gissin and Shalev-Shwartz, 2019) to reduce redundancy of the selected examples. In contrast, the other strategies tend to select less diverse batches, i.e., they are prone to choose redundant examples, especially in the *imbalanced-practical* scenario. Thus, combining these approaches with methods that encourage diversity (e.g., He et al., 2014; Zhdanov, 2019; Ash et al., 2019) can potentially lead to further improvement in their resultant prediction performance. In terms of representativeness, DAL, which is a representativeness-driven method, again consistently leads across the scenarios. In contrast, the tendency of the greedy core-set version to select outliers (Sener and Savarese, 2017), is indeed reflected in its relatively low representativeness scores. Interestingly, this is not the case for the *imbalanced-practical* scenario. This result can be attributed to the high bias of L towards query matches, which results in poor representativeness of L, which in turn leaves the main "responsibility" for representing the dataset on the batch examples selected from U. A deeper investigation of this result is left for further investigation. Other strategies have low representativeness scores compared to DAL, implying that they can be improved by combining them with techniques for encouraging representativeness and avoiding outliers.

A popular approach for improving classification quality is combining several, preferably complementary, AL strategies. In order to find pairs of strategies with high synergistic potential, we measured the overlap between the batches selected by each pair of strategies. Our analysis shows that for all pairs of strategies, the expected batch overlap is relatively low, and does not exceed 15%. In general, overlap was higher in the imbalanced scenarios, probably due to the general incentive to select positive examples, which are rare in the data. Also, the overlap within the uncertainty-based strategies was generally quite high. Nevertheless, the highest overlap between batches was between EGL (which is not an uncertainty-based approach) and LC. We leave for future work to try a combination of strategies with low overlap as a way to improve classification even further.

6 Conclusions

The recent emergence of pre-trained models, with BERT as a prominent example, is reshaping the NLP arena (Qiu et al., 2020). The promise embodied in these models is their ability to exploit massive unlabeled textual data to learn versatile, arguably universal language representations. These representations, in turn, are proven to be effective for a multitude of downstream NLP tasks.

A parallel line of research, dating back nearly three decades, is the notion of AL, aiming to minimize labeling burden within the supervised learning paradigm. The pairing of these two influential threads raises non-trivial questions. For example, BERT arguably attains excellent performance with relatively little labeled data, used to fine-tune this pre-trained model for a concrete task. It is not obvious to begin with, to what extent AL strategies can be used to outperform this already high bar.

To the best of our knowledge, the present work provides the first systematic study in this context,

while focusing on the prevalent problem of text classification. Moreover, we further focus our attention on a scenario well known to many practitioners – and notoriously difficult from a learning perspective – that of building a classifier when the class of interest is scarce in the data at hand. Aiming to further bridge the gap between research and practice, in our *imbalanced-practical* mode we simulate a user within this challenging scenario, armed only with simple queries to define the labeled seed that will bootstrap the AL process. Our results demonstrate the potential of AL on top of BERT, especially in this latter scenario. Notably, a training data seed resulting from a simple query is expected to capture only limited, and perhaps somewhat obvious, aspects of the class under consideration. Our study shows that the initial BERT model indeed suffers from poor prediction performance, mainly due to low recall values. However, while the random AL baseline is limited in its ability to help BERT emerge from this poor initial model, AL strategies turn out to be very helpful. Using the AL pipeline, BERT improves its recall by a large margin, generalizing beyond the narrow data it was initially exposed to.

This work focused on various binary classification tasks. A natural future direction is to conduct a similar empirical investigation of AL over BERT in the context of multi-class classification and regression tasks. It would also be interesting to investigate the realm of larger annotation budgets, and more recent BERT variants (Liu et al., 2019; Lan et al., 2019). Finally, the present work focused on existing AL strategies, which were mostly developed in the vision domain for CNNs. The development of novel AL methods, that are tailored for pre-trained models such as BERT, seems like an important direction for future work. We hope that the experimental results and analyses reported here, as well as the release of the research framework we developed, would be instrumental for these and other future studies.

References

Charu C Aggarwal and ChengXiang Zhai. 2012. A survey of text classification algorithms. In *Mining text data*, pages 163–222. Springer.

Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. 2019. Deep batch active learning by diverse, uncertain gradient lower bounds. *arXiv preprint arXiv:1906.03671*.

David A Cohn, Zoubin Ghahramani, and Michael I Jordan. 1996. Active learning with statistical models. *Journal of artificial intelligence research*, 4:129–145.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059.

Daniel Gissin and Shai Shalev-Shwartz. 2019. Discriminative active learning. *arXiv preprint arXiv:1907.06347*.

Tianxu He, Shukui Zhang, Jie Xin, Pengpeng Zhao, Jian Wu, Xuefeng Xian, Chunhua Li, and Zhiming Cui. 2014. An active learning approach with uncertainty, representativeness, and diversity. *The Scientific World Journal*, 2014.

Peiyun Hu, Zachary C Lipton, Anima Anandkumar, and Deva Ramanan. 2018. Active learning with partial feedback. *arXiv preprint arXiv:1802.07427*.

Jiaji Huang, Rewon Child, Vinay Rao, Hairong Liu, Sanjeev Satheesh, and Adam Coates. 2016. Active learning for speech recognition: the power of gradients. *arXiv preprint arXiv:1612.03226*.

Nathalie Japkowicz and Shaju Stephen. 2002. The class imbalance problem: A systematic study. *Intelligent Data Analysis*, pages 429–449.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations.

David D Lewis and William A Gale. 1994. A sequential algorithm for training text classifiers. In *SIGIR'94*, pages 3–12. Springer.

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

Mingyi Liu, Zhiying Tu, Zhongjie Wang, and Xiaofei Xu. 2020. Ltp: A new active learning strategy for bert-crf based named entity recognition. *arXiv preprint arXiv:2001.02524*.

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.

David Lowell, Zachary C Lipton, and Byron C Wallace. 2018. Practical obstacles to deploying active learning. *arXiv preprint arXiv:1807.04801*.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278, Barcelona, Spain.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124, Ann Arbor, Michigan. Association for Computational Linguistics.

Ameya Prabhu, Charles Dognin, and Maneesh Singh. 2019. Sampling bias in deep active classification: An empirical study. *arXiv preprint arXiv:1909.09389*.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *arXiv preprint arXiv:2003.08271*.

Ozan Sener and Silvio Savarese. 2017. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*.

Burr Settles. 2009. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Bo Shao, Lorna Doucet, and David R. Caruso. 2015. Universality versus cultural specificity of three emotion domains: Some evidence based on the cascading model of emotional intelligence. *Journal of Cross-Cultural Psychology*, 46(2):229–251.

A. Shelmanov, V. Liventsev, D. Kireev, N. Khromov, A. Panchenko, I. Fedulova, and D. V. Dylov. 2019. Active learning with deep pre-trained models for sequence tagging of clinical and biomedical texts. In *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 482–489.

Yanyao Shen, Hyokun Yun, Zachary C Lipton, Yakov Kronrod, and Animashree Anandkumar. 2017. Deep active learning for named entity recognition. *arXiv preprint arXiv:1707.05928*.

Aditya Siddhant and Zachary C. Lipton. 2018. Deep bayesian active learning for natural language processing: Results of a large-scale empirical study.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2018. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Wikipedia Talk Labels: Personal Attacks. .

Yazhou Yang and Marco Loog. 2018. A benchmark and comparison of active learning for logistic regression. *Pattern Recognit.*, 83:401–415.

Leihan Zhang and Le Zhang. 2019. An ensemble deep active learning method for intent classification. In *Proceedings of the 2019 3rd International Conference on Computer Science and Artificial Intelligence*, pages 107–111.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.

Ye Zhang, Matthew Lease, and Byron C Wallace. 2017. Active discriminative text representation learning. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Fedor Zhdanov. 2019. Diverse mini-batch active learning. *arXiv preprint arXiv:1901.05954*.

Jingbo Zhu, Huizhen Wang, Tianshun Yao, and Benjamin K Tsou. 2008. Active learning with sampling by uncertainty and density for word sense disambiguation and text classification. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 1137–1144.

A Datasets

In this paper we used the following datasets:

Subjectivity: http://www.cs.cornell.edu/people/pabo/movie-review-data/.

Polarity: http://www.cs.cornell.edu/people/pabo/movie-review-data/.

AG's News: http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html.
We used the version from: https://pathmind.com/wiki/open-datasets (look for the link *Text Classification Datasets*).

Wiki attack: https://figshare.com/articles/Wikipedia_Talk_Labels_Personal_Attacks/4054689.

ISEAR: https://www.unige.ch/cisa/research/materials-and-online-research/research-material/.

TREC: https://cogcomp.seas.upenn.edu/Data/QA/QC/

CoLA: https://nyu-mll.github.io/CoLA/

no.	dataset	class	# train	prior	# dev	prior	# test	prior	imb.
1	AG's News-imb	world	12,569	10%	2,456	10%	2,513	10%	Y
2	Subjectivity-imb	subjective	3,919	10%	560	10%	1,077	10%	Y
3	Polarity-imb	positive	4,142	10%	588	10%	1,193	10%	Y
4	Wiki attack	general	15,000	12%	3,000	11%	3,000	12%	Y
5	ISEAR	fear	5,366	14%	766	15%	1,534	15%	Y
6	TREC	location	4,674	15%	778	15%	500	16%	Y
7	AG's News	world	15,000	25%	3,000	26%	3,000	25%	N
8	CoLA	unacceptable	7,592	30%	959	30%	1,043	31%	N
9	Subjectivity	subjective	7,000	50%	1,000	50%	2,000	52%	N
10	Polarity	positive	7,463	50%	1,066	50%	2,133	50%	N

Table 4: Datasets, target classes and the split for train/dev/test sets with the class prior in each set (imb.= imbalanced).

dataset-category	query	precision	recall	F1
ISEAR-fear	fear/afraid/scared/scary	0.92	0.24	0.38
TREC-location	Where/countr/cit	1.00	0.48	0.65
Wiki attack-general	[A-Z]!	0.48	0.08	0.14
AG's news-imb	or over a list of countries and territories	0.32	0.42	0.36

Table 5: Queries performance on the test set

Table 4 provides details about their split into train, dev, and test sets. For each set its size and the prior of the target class is presented. Information about the performance on the test set of the queries used in the Imbalanced-practical scenario is given in Table 5.

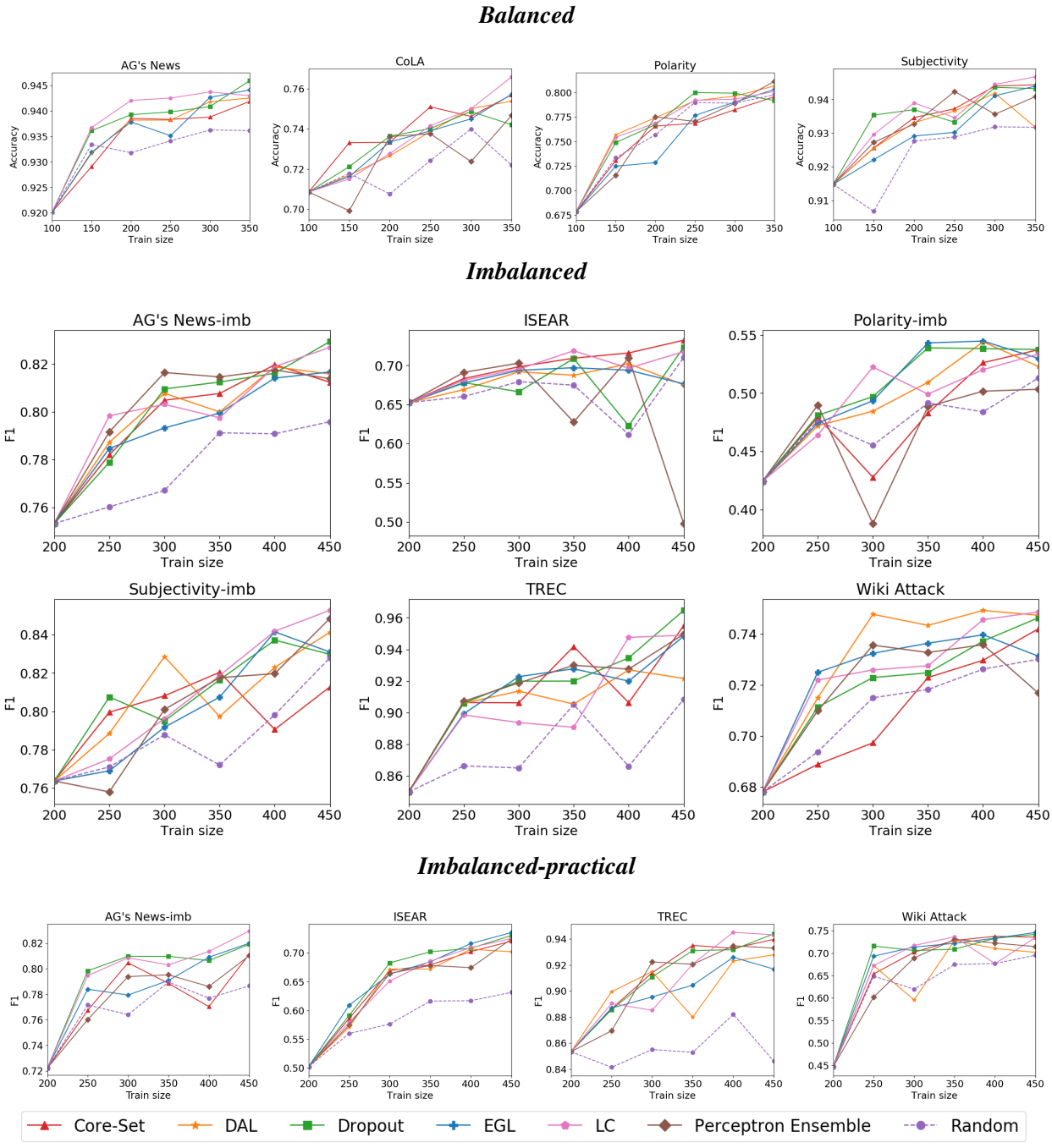

Figure 3: AL strategies compared to the Random baseline in the *balanced* (top row), *imbalanced* (two middle rows) and *imbalanced-practical* (bottom row) scenarios. Train size indicates the size of L.

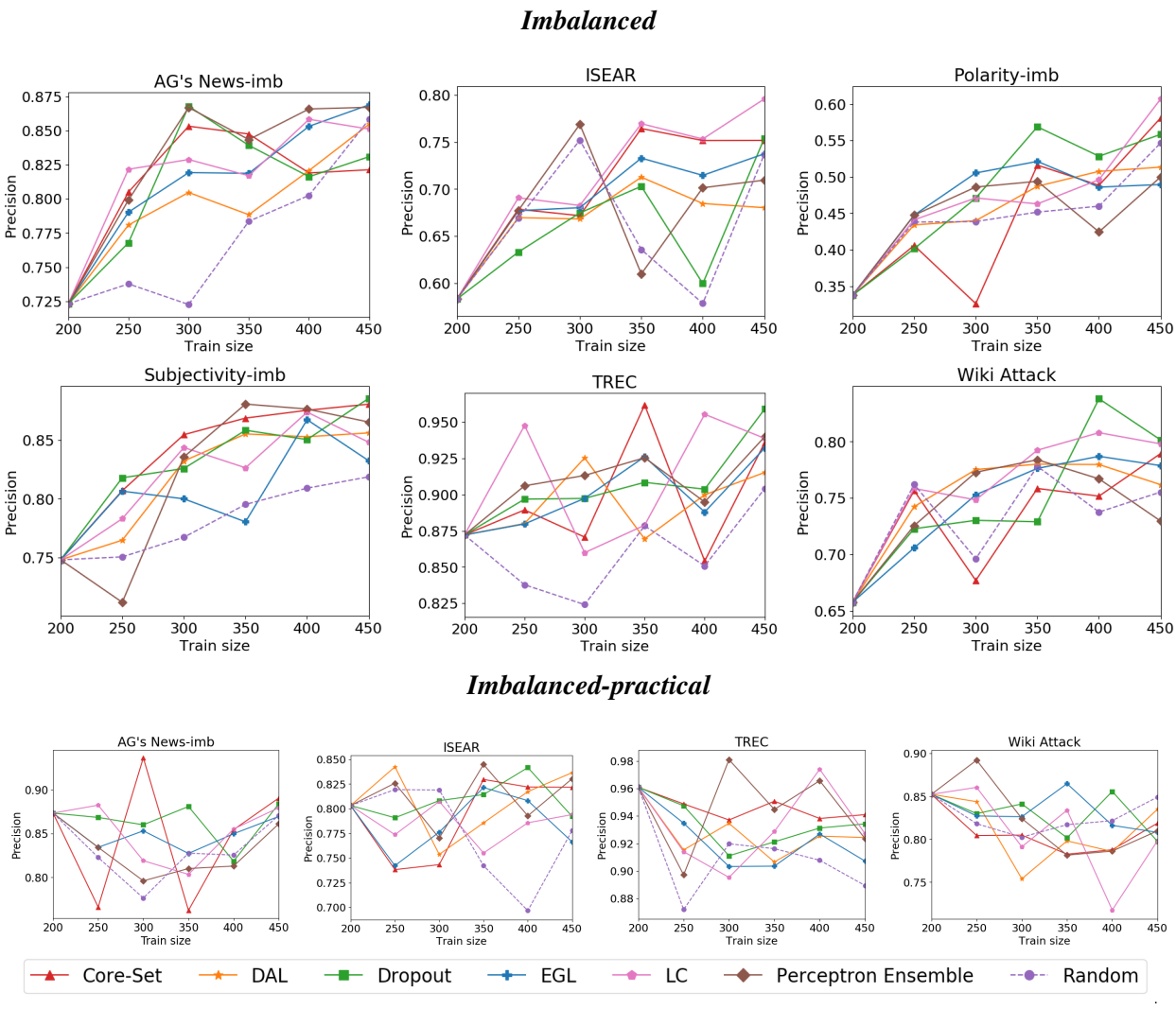

Figure 4: Precision of AL strategies and the Random baseline in the *imbalanced* (two top rows) and *imbalanced-practical* (bottom row) scenarios. Train size indicates the size of L.

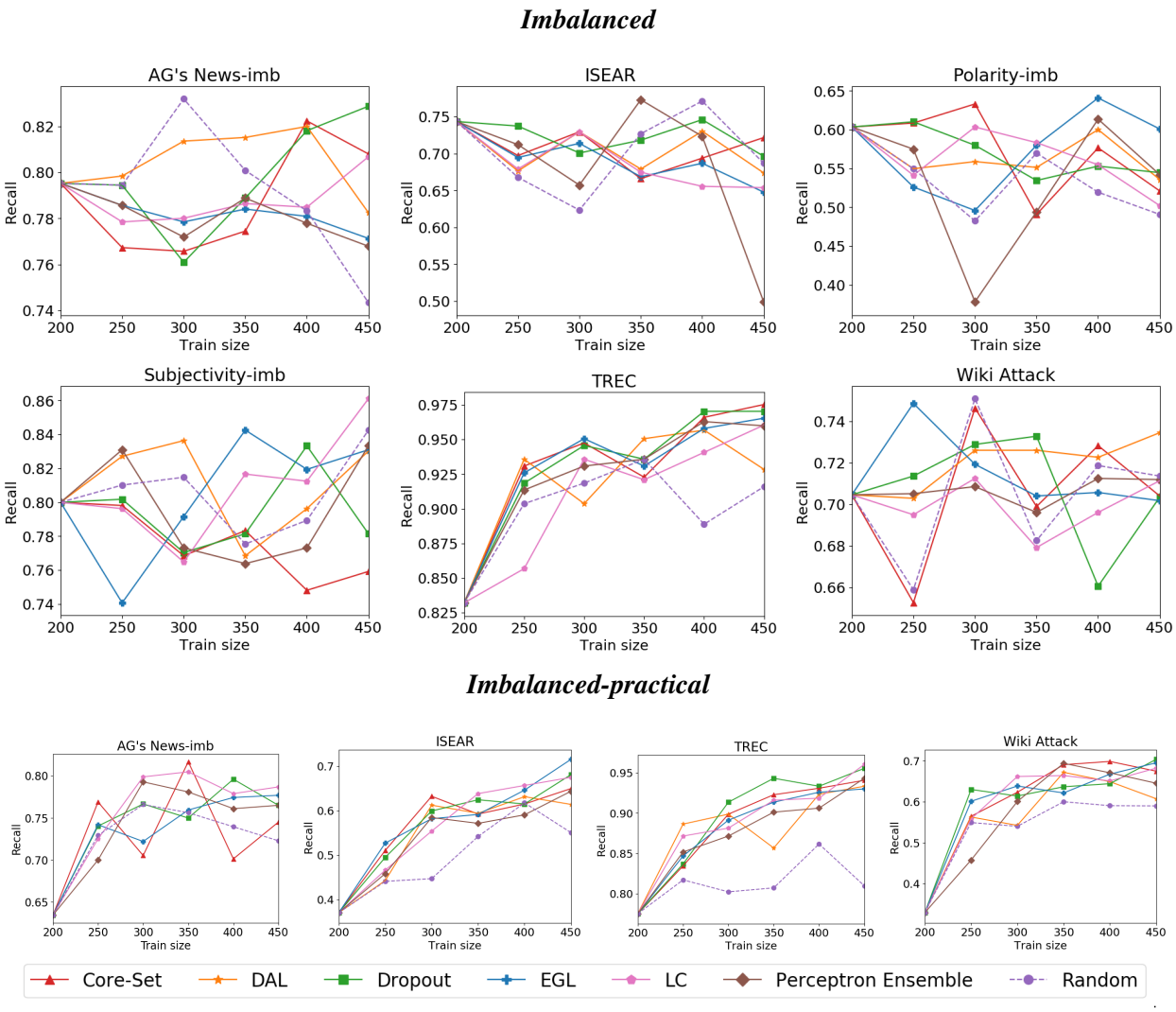

Figure 5: Recall of AL strategies and the Random baseline in the *imbalanced* (two top rows) and *imbalanced-practical* (bottom row) scenarios. Train size indicates the size of L.