# On the Fragility of Active Learners for Text Classification

**Anonymous ACL submission**

## Abstract

Active learning (AL) techniques optimally utilize a labeling budget by iteratively selecting instances that are most valuable for learning. However, they lack "prerequisite checks", i.e., there are no prescribed criteria to pick an AL algorithm best suited for a dataset. A practitioner must pick a technique they *trust* would beat random sampling, based on prior reported results, and hope that it is resilient to the many variables in their environment: dataset, labeling budget and prediction pipelines. The important questions then are: how often on average, do we expect any AL technique to reliably beat the computationally cheap and easy-to-implement strategy of random sampling? Does it at least make sense to use AL in an "Always ON" mode in a prediction pipeline, so that while it might not always help, it never under-performs random sampling? How much of a role does the prediction pipeline play in AL's success?

We examine these questions in detail for the task of text classification using pre-trained representations, which are ubiquitous today.

Our primary contribution here is a rigorous evaluation of AL techniques, old and new, across setups that vary wrt datasets, text representations and classifiers. This unlocks multiple insights around warm-up times, i.e., number of labels before gains from AL are seen, viability of an "Always ON" mode and the relative significance of different factors. Additionally, we release a framework for rigorous benchmarking of AL techniques for text classification.

## 1 Introduction

Within a supervised learning setup, Active Learning (AL) techniques use a *Query Strategy (QS)* to identify an unlabeled set of instances which is optimal in the following sense: if labelled and added to the training data, they lead to the greatest improvement in model accuracy, relative to any other same-sized set. In cases where labelling is expensive, the value proposition of AL is that it is cost-efficient compared to *random sampling*, and a model reaches greater accuracy with a smaller number of labelled instances.

In practice, an AL technique is selected based on the strength of prior reported results, i.e., there are no "prerequisite checks": tests that one might perform on an unlabeled dataset, that help to select a technique suited to a problem[1]. This trust extends to related decisions such as batch and seed sizes, as well as the hyperparameters (if any) of the AL technique since there is no way to empirically pick them: to compare with random sampling, or among techniques, labels are required. But if one had labels, they wouldn't need AL! In this sense, the AL setup is unforgiving as one needs to make the optimal choice in one shot.

This leads us to question the validity of the implicit but consequential assumption of task transfer. A related question is if it makes sense to use AL in an "Always ON" mode in a data labeling workflow; this is akin to asking if AL might perform *worse* than random sampling. We need to quantify both the frequency and magnitude of gains from AL, to be able to evaluate the cost of such pipeline; even simple AL techniques require a model to be evaluated over the unlabeled data pool, which can be expensive depending on the model complexity, size of the data pool and the latency allowed per AL iteration.

To be clear, we don't question if AL results are reproducible within the *original setups* they were reported in[2]; but whether any of those gains carry forward to *new setups*, which is how AL is used in practice.

[1] We refer to this as the practitioner's *decision model* and formalize it in §4.4.

[2] In the interest of fairness, we conducted limited **reproduciblity tests** for the AL techniques we benchmark here, and were able to replicate reported results - see §D.

We pick the area of text classification to investigate these concerns. The larger area of NLP has seen a rapid infusion of novel ideas of late. Today, a practitioner has easy access to a variety of powerful classifiers via packages such as *scikit-learn* (Pedregosa et al., 2011), *spaCy* (Honnibal et al., 2020) and *Hugging Face* (Wolf et al., 2020), and text representations, such as *Universal Sentence Encoding (USE)* (Cer et al., 2018), *MiniLM* (Wang et al., 2021) and *MPNet* (Song et al., 2020). This makes it a fertile ground for testing AL's utility.

In all this, *our motivation is to not disapprove of AL as an area for research, but to motivate the inclusion of multiple practical challenges in future studies*.

**Contributions**: Our primary contribution is a rigorous empirical analysis of the learning behavior of AL techniques over multiple text classification pipelines, that is targeted towards answering the questions asked above. Additionally, we open source an AL evaluation framework[3], to enable researchers to not only reproduce our analysis, but also to rigorously evaluate their own contributions.

## 2 Previous Work

Critique of AL is not new. Attenberg and Provost (2011) criticize AL for its unpredictable (for a task) warm-up times, i.e., a minimum number of labeled instances before which gains over random sampling are evident. Margatina and Aletras (2023) point out problems with AL simulations. Lüth et al. (2023) identify key issues leading to a lack of realistic AL evaluations and propose solutions that they apply to image classification. Lowell et al. (2019) study AL empirically but focus on the interesting notion of *successor models*, i.e., future models that would use the labeled data collected via AL using a specific model. Zhan et al. (2021) examine the empirical effectiveness of AL, but they don't evaluate on NLP tasks. Siddhant and Lipton (2018) is an empirical study of AL effectiveness similar in spirit to ours, but they focus on deep Bayesian methods.

This work differs from from existing literature wrt being a combination of: focusing on text classification, being empirical, employing a breadth of models (traditional and deep learning based) and employing recent techniques, e.g., *MPNet* (Song et al., 2020), *REAL* (Chen et al., 2023). While some conclusions we draw here might be similar to those reported earlier, we note that it is important to revise our collective mental models in a fast evolving area such as NLP, and in enabling that, even such conclusions are valuable.

## 3 Batch Active Learning - Overview

In this work, we specifically study the *batch* AL setting for text classification. Here, a QS identifies a *batch* of $b$ unlabeled points, at each iteration $t$, for $T$ iterations. A model $M_t$, that is trained on the accumulated labeled pool, is produced at the end of each iteration. The first iteration uses a seed set of $s$ randomly sampled points (although other strategies may be used).

We note that that $M_t$ should be produced using a *model selection* strategy (we use a hold-out set here), and must also be *calibrated* (we use *Platt scaling* (Platt, 2000; Niculescu-Mizil and Caruana, 2005)). The former ensures that $M_t$ doesn't overfit to the labeled data, which is likely in the initial iterations due to small quantities. The latter is required since many query strategies rely on uncertainty/confidence scores produced by $M_t$. Unfortunately, in our experience, multiple implementations/studies miss one or both of these steps.

To avoid any ambiguity, we provide pseudo-code for this AL setting in Algorithm 1 in §A.

## 4 Experiment Setup

In this section, we describe our experiment setup in detail.

### 4.1 Configuration Space of Experiments

Our experiment configurations vary wrt *datasets*, *text representations*, *classifiers*, the *batch* and *seed sizes*, and of course, the *QS*. We study the following QS here: (1) *Random* as baseline, (2) *Margin*[4] (Scheffer et al., 2001; Schröder et al., 2022), (3) *Contrastive Active Learning (CAL)* (Margatina et al., 2021), (4) *Discriminative Active Learning (DAL)* (Gissin and Shalev-Shwartz, 2019; Ein-Dor et al., 2020), and (5) *Representative Errors for Active Learning (REAL)* (Chen et al., 2023). We picked these either because they are contemporary, e.g., *REAL*, *DAL*, *CAL*, or have produced strong contemporary results, e.g., *Margin*.

Figure 1 enumerates the configuration space. For further details (including hyperparameters) see §B and §E. Note that all representations used are based

[3] Our code is available here: https://anonymous.4open.science/r/On_the_Fragility_of_Active_Learners/README.md.

[4] Also referred to as *Smallest Margin* or *Breaking Ties*, it is still considered to be competitive (Schröder et al., 2022).

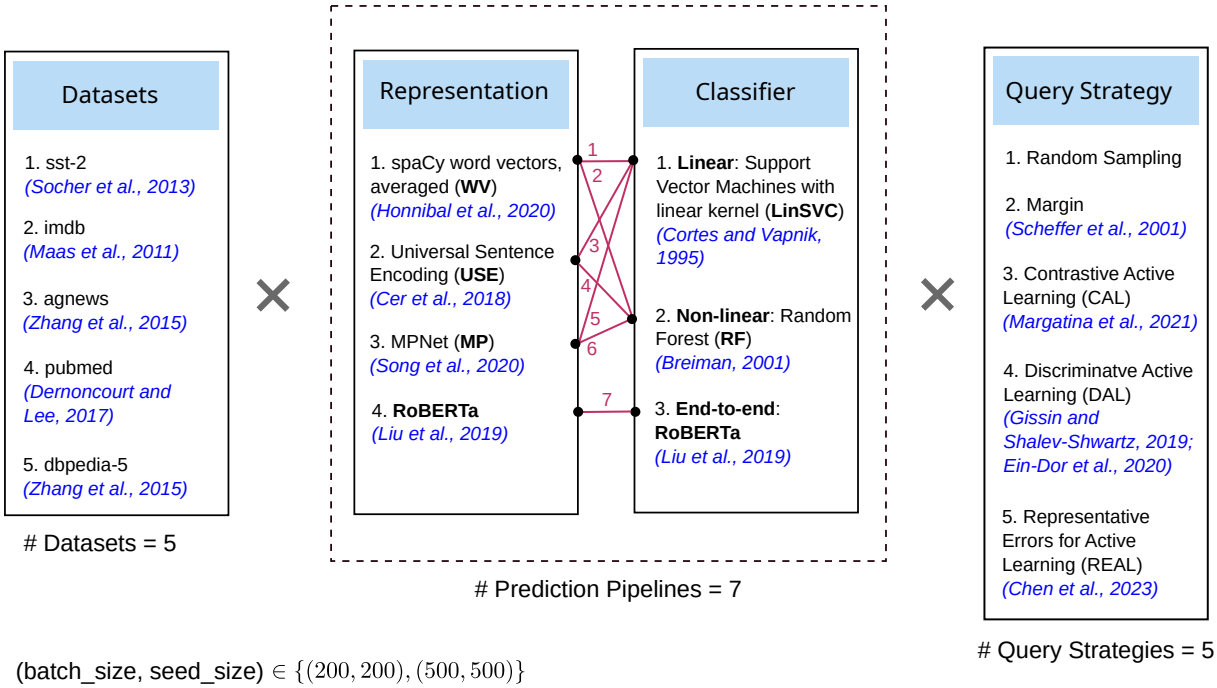



Figure 1: The space of experiments is shown. See §4.1 for description. All representations are produced by pre-trained models, which are ubiquitous in practice today. The lines between the boxes "Representation" and "Classifier" denote combinations that constitute our prediction pipelines. Note that RoBERTa is an end-to-end predictor, where there are no separate representation and classification steps.

on *pre-trained* models which have grown quite popular in the past few years. For classification, we picked one each of a linear, non-linear and Deep Learning based classifier. Since batch or seed sizes are inconsistent in AL literature, e.g., DAL, REAL and CAL respectively use batch sizes of 50, 150, 2280 - we vary these settings as well.

For an idea of the breadth of this search space, see Figure 2 which shows results for the dataset *agnews* and batch/seed size of $(200, 200)$.

### 4.2 Metrics and Other Settings

The **classifier accuracy metric** we use is the **F1 (macro)** score, since it prevents performance wrt dominant classes from overwhelming results. For measuring the **effectiveness of a QS**,we use the **relative improvement wrt the random QS** of the classifier score (see Equation 1). The **size of the unlabeled pool** is **20000** at the start of each experiment. If the original dataset has more than than 20000 instances, we extract a label-stratified sample, to retain the original class distribution. The **size of the test set** is **5000** - also a label-stratified sample from the corresponding test set of the original dataset.

We run an experiment till the size of the labeled set has grown to 5000 instances[5]. This implies $\boldsymbol{T = (5000 - 200)/200 = 24}$ **iterations** for the batch/seed size setting of $(200, 200)$, and similarly $\boldsymbol{T = 9}$ **iterations** for the $(500, 500)$ setting.

As shown in Figure 1 we have **350 unique configurations**. We also execute **each configuration three times** in the interest of robust reporting. This gives us a a total of $350 \times 3 = \mathbf{1050}$ **trials**. For *each AL iteration* of *each of these trials*, we follow the due process of model selection and calibration[6].

### 4.3 Notation and Terminology

We introduce some notation here that will help us precisely describe our analysis in later sections.

Let $f$ be a function that computes the model metric of interest, e.g., *F1-macro*. This accepts, as parameters, the *random variables* $h, q, d, b, s, n$, which are defined as follows:

- $h \in H$, the set of prediction pipelines.
- $q \in Q$, the set of query strategies. For convenience, we also define $q_R$ to be the *random*

[5]Beyond this labeled set size (unrelated to the test set size) different QSes produce similar gains - see §C.

[6]*RoBERTa* is the only exception since it is naturally well-calibrated (Desai and Durrett, 2020).

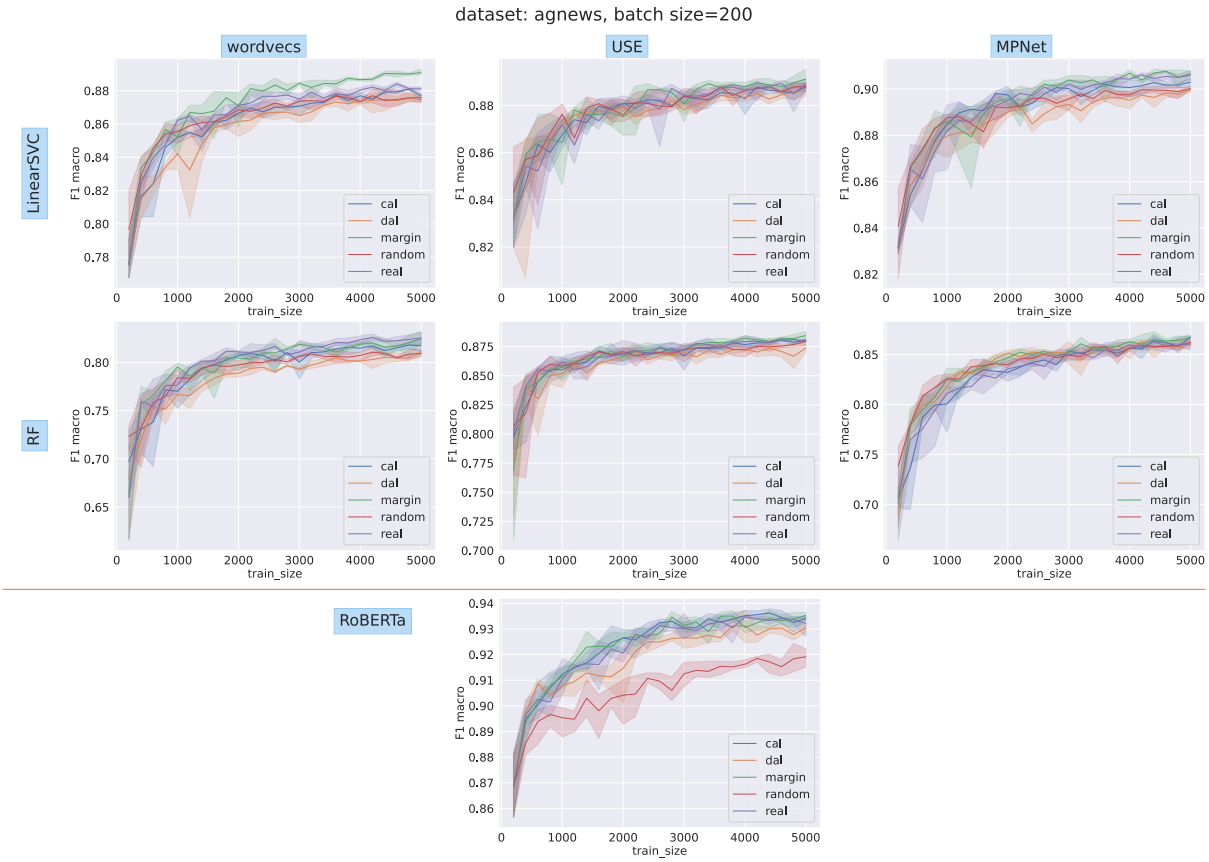



Figure 2: F1 macro scores on the test set at each iteration, for the dataset *agnews* and batch size of 200. The $x$-axes show size of the labeled data, the $y$-axes show the F1-macro scores on the test data.

QS, and $\mathcal{Q}_{NR} = \{cal, dal, real, margin\}$, i.e., the subset of non-random QS.

- $d \in D$, the set of datasets.
- $(b, s) \in V$, the set of batch and seed size combinations, i.e., $V = \{(200, 200), (500, 500)\}$
- $n$ is the size of the labeled data. In our experiments, $s \leq n \leq 5000$.

A specific value is indicated with a prime symbol on the corresponding variable, e.g., $h'$ is a specific prediction pipeline.

**QS Effectiveness**: We evaluate a non-random QS by measuring the relative improvement wrt the random QS, at a given number of labeled instances $n'$. We use the shorthand $\delta$:

$$\delta(f(h, q, d, b, s, n')) = \frac{f(h, \boldsymbol{q}, d, b, s, n') - f(h, \boldsymbol{q_R}, d, b, s, n')}{f(h, \boldsymbol{q_R}, d, b, s, n')} \quad (1)$$

### 4.4 Decision Model

Before looking at the results, we formalize the *decision model* of a practitioner using our notation. This helps us justify the aggregations we perform over results of individual experiments.

Because of lacking prerequisite checks, there is no preference for picking a factor in combination with others. We model them as *independent* variables, i.e., the probability of a configuration is $p(h)p(q)p(d)p(b, s)$. Since each of these probabilities is also *uniform*, e.g., the general practitioner is equally likely to encounter any dataset $d \in D$, each configuration has an identical probability of occurrence[7]: $1/(|H| \times |Q| \times |D| \times |V|)$. In other words, any *expectation* we wish to compute over these settings under this decision model is a simple *average*.

## 5 Results

We are now ready to look at the results of our experiments.

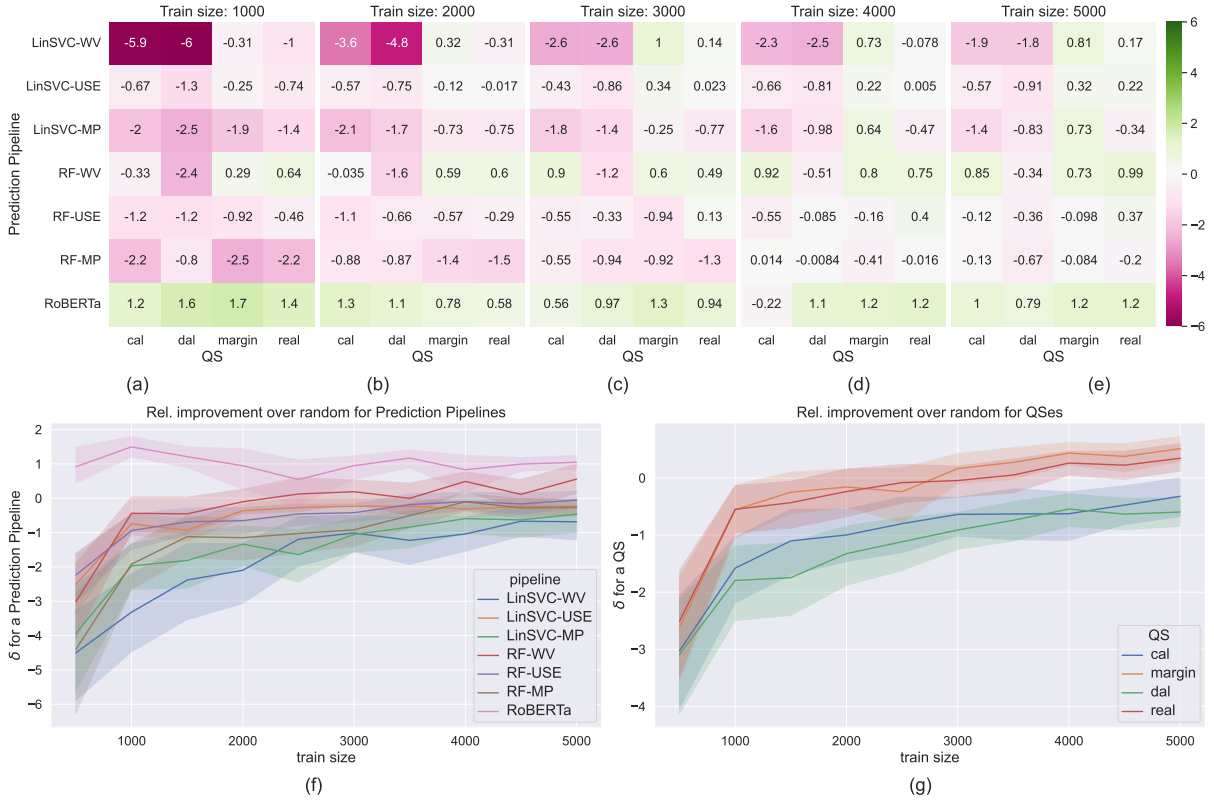



Figure 3: Expected relative improvement in *F1-macro* score over random. (a)-(e) show this for different predictors and QS, at different training sizes (see titles). These correspond to Equation 2. (f) and (g) show marginalized improvements for different predictors and QSes respectively; see equations 3 and 4.

### 5.1 Expected Gains from AL

Figure 3 shows the expected relative improvement, grouped in the following ways:

1. Figure 3(a)-(e): These heatmaps show the expected $\delta$ at a given number of instances $n' \in \{1000, 2000, 3000, 4000, 5000\}$. A cell for predictor $h'$ and a QS $q' \in Q_{NR}$ in the heatmap for $n'$ training instances shows[8]:

$$\mathbb{E}_{d,b,s}[\delta(f(h', q', d, b, s, n'))] \quad (2)$$

   The rows are arranged roughly in increasing order of classifier capacity, i.e., *LinSVC*, *RF*, *RoBERTa*, and within a group, in increasing order of approximate representation quality: word vectors (*WV*), *USE*, *MPNet*[9].

2. Figure 3(f): This shows $\delta$ only for prediction pipelines, marginalizing over QSes. This is easy to show in a standard line-plot. The $y$-value for $x = n'$ for predictor $h'$ denotes:

$$\mathbb{E}_{d,b,s,q \in Q_{NR}}[\delta(f(h', q, d, b, s, n'))] \quad (3)$$

3. Figure 3(g): This is analogous to (f) and shows $\delta$ for QSes while marginalizing over predictors. The $y$-value for a specific $x = n'$ for QS $q' \in Q_{NR}$ denotes:

$$\mathbb{E}_{d,b,s,h}[\delta(f(h, q', d, b, s, n'))] \quad (4)$$

*Observations*: In Figure 3(a)-(e), we see that as we move towards the right, the number of cells with $\delta \gtrapprox 0$ increases. This suggests that, in general, as the pool of labeled instances grows, AL becomes more effective. This might seem promising at first, but note that (a) we cannot predict *when* this happens in practice: we lack the theoretical tools, and it varies wrt both the predictor and the QS, and (b) if you look closely, its not that AL is becoming more effective but, rather, all configurations are

[7]They may inherit an environment with a specific prediction pipeline or a query strategy - we also present these conditional results. But within these conditions, the other factors are assumed to be independent and individually uniform.

[8]This expectation is over batch and seed sizes at given values of $n'$; but note, different batch sizes *don't produce same values for* $n'$. This is explicitly reconciled - see §F.

[9]The relative ordering of USE vs MPNet was obtained from the *Massive Text Embedding Benchmark (MTEB)* rankings, where *MPNET* leads *USE* by $\sim 100$ positions today.

converging towards[10] $\delta = 0$. In other words, *in low label regimes, where we expect AL to benefit us, there can be a lot of variance - it might even under-perform random sampling - and at high label regimes, their performance, even if positive, is not very different from random sampling*.

Among predictors (Figure 3(f), but this is also apparent in (a)-(e)), for *RoBERTa* we consistently observe $\delta > 0$. Among QSes, *REAL* and *Margin*, seem to do well at larger data regimes - as visible in Figure 3(g), but also in (d) and (e). The performance of *Margin* might seem somewhat surprising, since this is an old technique (proposed in Scheffer et al. (2001)), but similar observations have been reported elsewhere (Schröder et al., 2022).

### 5.2 Always ON Mode

Another question we might ask is that even if AL doesn't always surpass random, is there a downside to making it a permanent part of a labeling workflow - multiple tools allow this today[11], e.g., Montani and Honnibal; Tkachenko et al. (2020-2022)?

Table 1 shows some relevant numbers.

| *Avg. for* | % times $\delta < 0$ | $\overline{\delta}_{\geq 0}$ | $\overline{\delta}$ |
|---|---|---|---|
| **Overall** | 51.82 | 0.89 | -0.74 |
| LinSVC-WV | 61.71 | 0.70 | -1.90 |
| LinSVC-USE | 61.57 | 0.46 | -0.64 |
| LinSVC-MP | 63.71 | 0.40 | -1.48 |
| RF-WV | 47.29 | 1.31 | -0.30 |
| RF-USE | 60.57 | 0.71 | -0.63 |
| RF-MP | 60.14 | 0.60 | -1.24 |
| RoBERTa | 7.71 | 1.29 | 1.01 |
| CAL | 55.60 | 0.81 | -1.07 |
| DAL | 70.12 | 0.82 | -1.29 |
| Margin | 38.45 | 0.97 | -0.25 |
| REAL | 43.10 | 0.89 | -0.34 |

Table 1: The %-age of times model *F1-macro* scores are worse than random are shown. Also shown are the average $\delta$s when scores are at least as good as random, and average $\delta$s in general. These are relevant to the "Always ON" mode, discussed in §5.2. See Table 6 in §G for standard deviations.

[10]This is something we observe in a separate analysis as well - see §C. In fact, this is the reason why we grow the labeled set to only 5000 instances in our experiments - mentioned in §C.

[11]**Important**: We have *not* evaluated these tools. They are cited as examples of common tools used in data labeling workflows in the industry.

*Observations*: In general, (first row, "**Overall**"), the number of incidents where the relative improvement was *strictly negative* (counted at various labeled data sizes across configurations) is $51.82\%$. This might be suggested by the heatmaps in Figure 3(a)-(e) as well, where approximately the left upper triangle of the plots combined indicates $\delta < 0$. The average improvement when AL is as good as random is low, i.e., $\overline{\delta}_{\geq 0} = 0.89$, and on the whole this quantity is *actually negative*, i.e., $\overline{\delta} = -0.74$. Again, the use of *RoBERTa* leads to favorable scores. Among QSes, *Margin* and *REAL* perform relatively well.

Under our decision model - §4.4 - the practical implication is bleak: in the "Always ON" mode, stopping labeling early risks negative improvement. The only way to ensure $\delta \geq 0$ is to accumulate quite a few labels, i.e., move out of the left upper triangular region in Figure 3(a)-(e), but then the average improvement is low. Essentially, the "Always ON" mode is viable if the small relative gains from labeling $4000-5000$ instances are useful.

### 5.3 Effect of Prediction Pipeline vs QS

Papers on AL typically contribute QSes. Here we ask if that focus is warranted, i.e., what has a greater impact? - the QS or the prediction pipeline?

We might suspect that it is the pipeline, given the performance of *RoBERTa* in both Figure 3 and Table 1. To precisely assess their relative effect, we use the *Friedman* test (Friedman, 1937) in the following way:

1. Take the example of QSes. For each non-random QS $q'$, we list the scores $\delta(f(h, q', d, b, s, n))$ for different values of $h, d, b, s, n$. Since there are four non-random QSes, this gives us four sets of matched observations.

2. We calculate the *p-value* on these observations. A low value indicates a high sensitivity to changing the QS.

We follow an analogous procedure for prediction pipelines, where we obtain seven matched observation sets. These are the *p-values* we obtain:

- QSes: $8.45e-129$.
- Prediction Pipelines: $5.39e-186$.

The lower *p-value* for prediction pipelines indicates that they have a greater influence on the relative

| *Predictor* | *p-value* | *QS* | *p-value* |
|---|---|---|---|
| LinSVC-WV | 0.18 | CAL | 0.77 |
| LinSVC-USE | 0.41 | DAL | 0.02 |
| LinSVC-MP | 0.60 | Margin | 0.32 |
| RF-WV | 0.13 | REAL | 0.07 |
| RF-USE | 0.03 | | |
| RF-MP | 0.03 | | |
| RoBERTa | $1.32e{-10}$ | | |
| **Overall:** 0.90 | | | |

Table 2: The *p-values* for a two-sided *Wilcoxon signed-rank test* over $\delta$ values, from using batch/seed size $(200, 200)$ vs $(500, 500)$. See §5.4 for details.

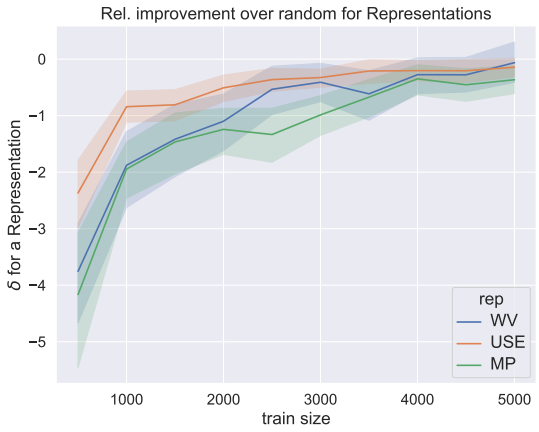



Figure 4: Effect of text representations on the relative improvement.

improvement. This complements our other observations that relative improvements are not consistent for QSes alone.

### 5.4 Effect of Batch/Seed Size

We perform a *Wilcoxon signed-rank test* (Wilcoxon, 1945) to assess the effect of batch/seed sizes on $\delta$. This is a paired test and ideally we should match observations $\delta(f(h, q, d, 200, 200, n))$ and $\delta(f(h, q, d, 500, 500, n))$. However, recall that since different batch/seed sizes don't lead to the same values of $n$ - we explicitly align the sizes for such comparison (detailed in §F).

The overall *p-value* of 0.90 indicates that our batch/seed settings don't influence $\delta$ in general. The exception is *RoBERTa*, with *p-value*$= 1.32e{-10}$. A further one-sided test tells us that the batch/seed size setting of $(200, 200)$ leads to greater $\delta$ values (*p-value*$= 6.57e{-11}$).

### 5.5 Effect of Representation

Finally, we assess the effect of text representation on relative improvements. Since we want to evaluate representations alone (the prediction pipeline as a whole was already evaluated in §5.1), we ignore *RoBERTa* for this exercise, since its an end-to-end classifier.

Figure 4 shows how the relative improvement $\delta$ varies with the embedding used, marginalized over other configuration variables.

We note that *USE* outperforms *MPNet*. This is surprising to us because on the MTEB (Muennighoff et al., 2022) benchmarks *MPNet* scores much higher. A hypothesis that might explain both results is that *USE* doesn't capture fine-grained contexts as much as *MPNet* does; while this might be problematic for MTEB (esp. tasks that rely on precise similarity measurement, such as retrieval), the fuzzier embedding space of *USE* is better in terms of covering the concept space in the dataset earlier in the AL process. This enables better assessment of informativeness, and therefore, sampling, by a non-random QS.

## 6 Summary and Conclusion

After extensive evaluation of different AL algorithms, we are forced to conclude that it is difficult to practically benefit from AL. Gains from QSes are inconsistent across datasets, prediction pipelines and text representations. In fact, between QSes and prediction pipelines, the latter seems to have a greater influence on the relative improvement over random (§5.3). The only general pattern we see is that positive relative improvements become likely as labeled instances accumulate; but these improvements are too small to be broadly useful (§5.1). Another reason as to why it is hard to derive any practical advice is that we lack the tools, theoretical or empirical, to identify a settings-specific warm-start size; when do we stop labeling to realize gains, however small? Further, we noted in §5.2 that using AL in an "Always ON" mode can actually perform worse than random sampling.

The use of *RoBERTa* as the prediction pipeline is the only (isolated) case where we see consistent positive relative improvements. Our hypothesis as to why is that an end-to-end classifier has a more coherent view of the overall distribution, and therefore informativeness of a sample. But, obviously, we can't discount the role that *RoBERTa's* specific pre-training might play here, and further experimentation is required to disentangle their respective influences.

Although extensive, this study may be considered "limited" relative to real-world variances, e.g., many more choices of classifiers, datasets, which leads us to suspect that the true picture is probably more dismal.

What might we do to make the field of AL more useful? We feel the biggest problem in AL use is that practitioners have to *blindly guess* what specific AL technique will work best for their problem. As a field we need to embrace a broader discourse where the success of a technique needs to be tied to fundamental properties of datasets, e.g., *topological* features (Chazal and Michel, 2021), and predictors, e.g., *VC dimension* (Vapnik, 1995), that are identifiable in an *unsupervised* manner in novel settings.

## 7 Limitations

Being an empirical work, our conclusions are tied to the algorithms and settings analyzed.

## References


Josh Attenberg and Foster Provost. 2011. Inactive learning? difficulties employing active learning in practice. *SIGKDD Explor. Newsl.*, 12(2):36–41.

Leo Breiman. 2001. Random forests. *Machine Learning*, 45(1):5–32.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, Brussels, Belgium. Association for Computational Linguistics.

Frédéric Chazal and Bertrand Michel. 2021. An introduction to topological data analysis: Fundamental and practical aspects for data scientists. volume 4.

Cheng Chen, Yong Wang, Lizi Liao, Yueguo Chen, and Xiaoyong Du. 2023. Real: A representative error-driven approach for active learning. In *Machine Learning and Knowledge Discovery in Databases: Research Track: European Conference, ECML PKDD 2023, Turin, Italy, September 18–22, 2023, Proceedings, Part I*, page 20–37, Berlin, Heidelberg. Springer-Verlag.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning*, 20(3):273–297.

Franck Dernoncourt and Ji Young Lee. 2017. PubMed 200k RCT: a dataset for sequential sentence classification in medical abstracts. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 308–313, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Shrey Desai and Greg Durrett. 2020. Calibration of pre-trained transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 295–302, Online. Association for Computational Linguistics.

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active Learning for BERT: An Empirical Study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7949–7962, Online. Association for Computational Linguistics.

Milton Friedman. 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200):675–701.

Daniel Gissin and Shai Shalev-Shwartz. 2019. Discriminative active learning. *CoRR*, abs/1907.06347.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. "spacy: Industrial-strength natural language processing in python".

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.

David Lowell, Zachary C. Lipton, and Byron C. Wallace. 2019. Practical obstacles to deploying active learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 21–30, Hong Kong, China. Association for Computational Linguistics.

Carsten Tim Lüth, Till J. Bungert, Lukas Klein, and Paul F Jaeger. 2023. Navigating the pitfalls of active learning evaluation: A systematic framework for meaningful performance assessment. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Katerina Margatina and Nikolaos Aletras. 2023. On the limitations of simulating active learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4402–4419, Toronto, Canada. Association for Computational Linguistics.

Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. 2021. Active learning by acquiring contrastive examples. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 650–663, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ines Montani and Matthew Honnibal. Prodigy: A modern and scriptable annotation tool for creating training data for machine learning models.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

Emma Thuong Nguyen and Abhishek Ghose. 2023. Are good explainers secretly human-in-the-loop active learners? In *"AI&HCI workshop at the 40th International Conference on Machine Learning", ICML*.

Alexandru Niculescu-Mizil and Rich Caruana. 2005. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, page 625–632, New York, NY, USA. Association for Computing Machinery.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

John Platt. 2000. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Adv. Large Margin Classif.*, 10.

Tobias Scheffer, Christian Decomain, and Stefan Wrobel. 2001. Active hidden markov models for information extraction. In *Advances in Intelligent Data Analysis*, pages 309–318, Berlin, Heidelberg. Springer Berlin Heidelberg.

Christopher Schröder, Andreas Niekler, and Martin Potthast. 2022. Revisiting uncertainty-based query strategies for active learning with transformers. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2194–2203, Dublin, Ireland. Association for Computational Linguistics.

Aditya Siddhant and Zachary C. Lipton. 2018. Deep Bayesian active learning for natural language processing: Results of a large-scale empirical study. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2904–2909, Brussels, Belgium. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew 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*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. In *Advances in Neural Information Processing Systems*, volume 33, pages 16857–16867. Curran Associates, Inc.

Maxim Tkachenko, Mikhail Malyuk, Andrey Holmanyuk, and Nikolai Liubimov. 2020-2022. Label Studio: Data labeling software. Open source software available from https://github.com/heartexlabs/label-studio.

Vladimir N. Vapnik. 1995. *Constructing Learning Algorithms*. Springer New York.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. 2021. MiniLMv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2140–2151, Online. Association for Computational Linguistics.

Frank Wilcoxon. 1945. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Xueying Zhan, Huan Liu, Qing Li, and Antoni B. Chan. 2021. A comparative survey: Benchmarking for pool-based active learning. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4679–4686. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

## A Pseudo-code for Batch Active Learning

**Algorithm 1:** Batch Active Learning.

**Input:** Unlabeled data $X_U$, test data $(X_{test}, Y_{test})$, query strategy $\mathcal{Q}$, seed set selection strategy $\mathcal{A}$, search space $\Theta$ for model $\mathcal{M}$, seed size $s$, batch size $b$, number of iterations $T$, metric $\mathcal{V}$

**Result:** Scores on test set at various iterations $\{(\mathcal{V}_0, 0), (\mathcal{V}_1, 1), ..., (\mathcal{V}_T, T)\}$

1 $result \leftarrow \{\}$ `// to be returned`
2 $X_{L,0}, X_{U,0} \leftarrow \mathcal{A}(X_U, s)$
3 $(X_{L,0}, Y_{L,0}) \leftarrow$ obtain labels for $X_{L,0}$
4 $M_0 \leftarrow \arg\max_{\theta \in \Theta} M_\theta((X_{L,0}, Y_{L,0}))$
`// both model selection and calibration are performed`
5 $\mathcal{V}_0 \leftarrow \mathcal{V}(M_0(X_{test}), Y_{test})$
6 $result \leftarrow result \cup \{(\mathcal{V}_0, 0)\}$
7 **for** $t \leftarrow 1$ **to** $T$ **do**
8 $X^{new}_{L,t}, X_{U,t} \leftarrow \mathcal{Q}(M_{t-1}, X_{U,t-1}, (X_{L,t-1}, Y_{L,t-1}), b)$
9 $(X^{new}_{L,t}, Y^{new}_{L,t}) \leftarrow$ obtain labels for $X^{new}_{L,t}$
10 $(X_{L,t}, Y_{L,t}) \leftarrow$ add $(X^{new}_{L,t}, Y^{new}_{L,t})$ to $(X_{L,t-1}, y_{L,t-1})$
11 $M_t \leftarrow \arg\max_{\theta \in \Theta} M_\theta((X_{L,t}, Y_{L,t}))$
$\mathcal{V}_t \leftarrow \mathcal{V}(M_t(X_{test}), Y_{test})$
12 $result \leftarrow result \cup \{(\mathcal{V}_t, t)\}$
13 **end**
14 **return** $result$

At a high-level, at every AL iteration $1 \leq t \leq T$, we use a query strategy $\mathcal{Q}$ to select a $b$-sized batch of instances from the unlabeled pool of data (line 8). We obtain labels for this set (line 9) and add it to the existing pool of labeled data (line 10). We then train a model $M_t$ over this data (line 11). We emphasize that:

1. The model $M_t$ is obtained after performing *model selection* over its hyperparameter space $\Theta$, using *grid-search* against a *validation set*. The validation set is a label-stratified subset (a 20% split) of the current labeled set; the rest is used for training.

2. The model is also *calibrated*[12]. This is critical since query strategies $\mathcal{Q}$ often use the predicted class probabilities from $M_t$. We use *Platt scaling* (Platt, 2000; Niculescu-Mizil and Caruana, 2005).

The process is initialized by selecting a seed set of size $s$ from the unlabeled data pool, using a strategy $\mathcal{A}$ (line 2). We use random selection for this step.

We also note that a "model" here might mean a combination of a text representation, e.g., *word vectors*, and a classifier, e.g., *Random Forest*; further detailed in Section 4.1.

## B Experiment Configurations

In our experiments, we vary *classifiers*, *text representations* (we often jointly refer to them as a *prediction pipeline*), *batch size*, *seed size* and, of course, *query strategies*. These combinations are visualized in Figure 1, and are detailed in Section.

These combinations are listed below:

1. **Prediction pipeline**: There are two categories of pipelines we use:

   (a) Separate representation and classifier: The representations used are *USE* (Cer et al., 2018), *MPNet* (Song et al., 2020) and *word vectors*[13] (we use the models provided by the *spaCy* library (Honnibal et al., 2020)). For classification, we use *Random Forests (RF)* (Breiman, 2001) and *Support Vector Machines* (Cortes and Vapnik, 1995) with a *linear kernel* - we'll term the latter as "*LinearSVC*". We use off-the-shelf representations and they are *not* fine-tuned on our data. Only the classifiers are trained on our data.

   (b) End-to-end classifier: This does not require a separate representation model. We use *RoBERTa* (Liu et al., 2019) (a variant of *BERT*). This is fine-tuned on the labeled data at each AL iteration.

   Hyperparameter search spaces are detailed in Section E.2 of the Appendix. As noted in Section 3, model selection and calibration are performed during training of a prediction pipeline. The only exception is *RoBERTa*, which has been shown to be well-calibrated out of the box (Desai and Durrett, 2020).

[12] A notable exception is in our use of the *RoBERTa* model, which already is well calibrated (Desai and Durrett, 2020).

[13] The vectors of all words in a sentence are averaged to obtain its representation.

The first category gives us $2 \times 3 = 6$ combinations. Counting *RoBERTa*, we have **7 prediction pipelines** in our study.

2. **Query Strategy**: we list these below, with the year of publication mentioned, to show our focus on contemporary techniques:

   (a) *Random*: the batch is selected uniformly at random. This forms our baseline.

   (b) *Margin*[14] (Scheffer et al., 2001) (2001): this selects instances with the smallest differences between the confidence of the most likely and the second-most likely predicted (by the current classifier[15]) classes. Despite being a relatively old technique, it continues to be competitive (Schröder et al., 2022).

   (c) *Contrastive Active Learning (CAL)* (Margatina et al., 2021) (2021): chooses instances whose predicted class-probability distribution is the most different (based on *KL divergence*) from those of their $k$-nearest neighbors. This is similar to another work (Nguyen and Ghose, 2023), where such conflicts are detected using the *explanation space* produced by XAI techniques.

   (d) *Discriminative Active Learning (DAL)* (Gissin and Shalev-Shwartz, 2019; Ein-Dor et al., 2020) (2019): a binary classifier (a feedforward neural network) is constructed to discriminate between labeled and unlabeled data, and then selects unlabeled instances with the greatest predicted probability of being unlabeled. This picks examples that are most different from the labeled instances in this classifier's representation space. While the original work (Gissin and Shalev-Shwartz, 2019) only considers image datasets, a separate study shows its efficacy on text (Ein-Dor et al., 2020).

   (e) *Representative Errors for Active Learning (REAL)* (Chen et al., 2023) (2023): identifies clusters in the unlabeled pool and assigns the majority predicted label as a "pseudo-label" to all points in it. Instances are then sampled whose predictions differ from the pseudo-label. The extent of disagreement and cluster size are factored into the sampling step

   We use a total of **5 query strategies**.

3. **Datasets**: we use **5 standard datasets**: *ag-news*, *sst-2*, *imdb*, *pubmed* and *dbpedia-5* (a 5-label version of the standard *dbpedia* dataset that we created). These are detailed in Table 3. The extent of class imbalance is represented by the *label entropy* column, which is calculated as $\sum_{i \in C} -p_i \log_{|C|} p_i$, with $C$ being the set of classes.

4. **Batch and Seed sizes**: We use batch and seed size combinations of $(200, 200)$ and $(500, 500)$. This is a total of **2 combinations**.

5. **Trials**: For statistical significance, we run **3 trials** for each combination of the above settings.

## C In what data regimes do query strategies most differ?

We would intuitively expect that *F1-macro* scores from different QSes (for a given pipeline and dataset) should converge as we see more data due to at least two reasons:

- The concept space in the data would be eventually covered after a certain number of instances. Adding more data isn't likely to add more information, i.e., there are *diminishing returns* from adding more data.

- At later iterations, there is less of the unlabeled pool to choose from.

Indeed, Figure 5 confirms this. We first compute variances in *F1-macro* scores for each different pipeline/dataset combination[16] across QSes at a given labeled set size. And then we average these variances across datasets and pipelines - this is the $y$-axis. We see that the expected variance shrinks after a while, and at 5000 labeled points it is close to zero, i.e., the differences from using different QSes, pipelines etc isn't much. This is why we restrict the labeled set size to 5000 instances in our experiments (as mentioned in §4.2).

[14] Also referred to as *Smallest Margin* or *Breaking Ties*.

[15] Note that in reference to Algorithm 1, at iteration $t$, the query strategy $\mathcal{Q}$ uses model $M_{t-1}$.

[16] This step comes first since the accuracies obtained by a *LinearSVC* would be very different from those by *RoBERTa*, and we don't want to mix them.

| Dataset | # classes | Label entropy | Description |
|---|---|---|---|
| sst-2 | 2 | 1.0 | Single sentences extracted from movie reviews with their sentiment label (Socher et al., 2013). |
| imdb | 2 | 1.0 | Movie reviews with corresponding sentiment label (Maas et al., 2011). |
| agnews | 4 | 1.0 | News articles with their topic category (Zhang et al., 2015). |
| pubmed | 4 | 0.9 | Sentences in medical articles' abstracts which are labeled with their role on the abstract (Dernoncourt and Lee, 2017). |
| dbpedia-5 | 5 | 1.0 | A subset of *dbpedia* (Zhang et al., 2015) which contains Wikipedia articles accompanied by a topic label. The original dataset's instances are evenly distributed across 14 classes. To form *dbpedia-5*, we use only the first 5 classes: *Company, EducationalInstitution, Artist, Athlete, OfficeHolder*. This was done to reduce the training time of one-vs-all classifiers, e.g., *LinearSVC*. |

Table 3: Datasets used. Label entropy represents class imbalance - see §B for description.

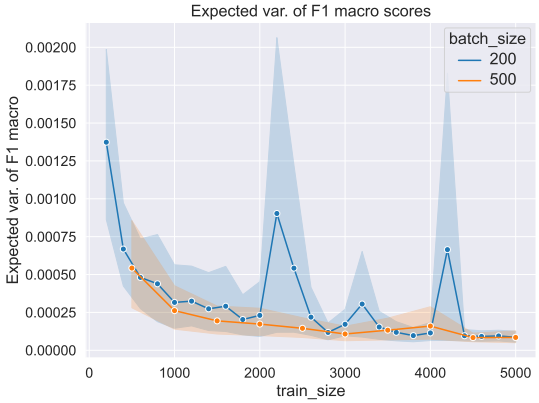



Figure 5: Expectation over variance of *F1-macro* given a pipeline and dataset, plotted against size of labeled data. Note that the batch/side sizes don't strongly influence trends.

## D Reproducibility Experiments

As mentioned earlier, our intention is *not* to suggest that the techniques we evaluate, e.g., REAL, CAL, DAL, don't work. In the specific settings discussed in their respective papers, they most likely perform as reported. In the interest of fairness, we have conducted limited independent tests that confirm this.

In all cases, we have attempted to replicate the original settings, e.g. same train/development/test data split, model type, seed/batch sizes, number of AL iterations as shown in Table 4. For CAL, REAL, we report the F1-macro scores on *agnews*, in which classes are evenly distributed, instead of the accuracy provided in the original papers. For DAL, we use the dataset *cola*[17] and utilise the *Hugging Face* library to finetune BERT (while the original work employs *TensorFlow*[18], but we use equivalent settings). Figure 6 shows a comparison between our results and the reported ones in these papers (Margatina et al., 2021; Chen et al., 2023; Ein-Dor et al., 2020) for CAL, REAL, DAL, respectively. Despite some minor differences in the setups, we observe that these AL methods work as described in their respective papers in these settings.

One significant difference between these settings compared to our methodology is the use of a predetermined *labeled* development set for all BERT/RoBERTa model finetuning. This set is relatively larger than the AL batch or seed size and is not part the labeled data available at each AL iteration. This is impractical in scenarios where AL is typically used: labeling is expensive. Moreover, in some cases, there is no model selection performed, which we remedy in our experiments (Section 3).

## E Hyperparameters

### E.1 Query Strategy (QS) hyperparameters

For each QS's hypeparameters, we use the values recommended by the authors in corresponding papers. This means setting number of nearest neighbors in CAL to 10, number of clusters in DAL to 25, and keeping the same discriminative model in REAL.

[17] https://nyu-mll.github.io/CoLA/
[18] https://www.tensorflow.org/

| AL | Dataset | AL loop | Classifier & text representation | QS parameters | Metric |
|---|---|---|---|---|---|
| CAL | *agnews* | b=2280<br>s=1140<br>T = 7 | BERT (bert-base-cased)<br>[CLS] at the last hidden layer<br>learning rate = 2e-5<br>train batch size = 16<br># epochs = 3<br>sequence length = 128<br>warmup ratio = 0.1<br># evaluations per epoch = 5 | # neighbors=10 | F1-macro |
| DAL | *cola* | b=50<br>s=100<br>T = 5 | BERT (bert-base-uncased)<br>[CLS] at the pooled layer<br>learning rate = 5e-5<br>train batch size = 50<br># epochs = 5<br>sequence length = 50<br>warmup ratio = 0<br># evaluations per epoch = 1 | - | Accuracy |
| REAL | *agnews* | b=150<br>s=100<br>T = 8 | RoBERTa (roberta-base)<br>[CLS] at the last hidden layer<br>learning rate = 2e-5<br>train batch size = 8<br># epochs = 4<br>sequence length = 96<br>warmup ratio = 0.1<br># evaluations per epoch = 4 | # clusters=25 | F1-macro |

Table 4: Settings for reproducibility experiments.

### E.2 Hyperparameters search for prediction pipelines

Table 5 shows the search space for hyperparameters we use for each classifier.

## F Averaging over Different Batch-Sizes

When computing expectations over different batch/seed sizes (like in Equation 2) a challenge is that different settings don't lead to same number of instances. For ex., for $b = 200, s = 200$, the size of the trained pool assumes the values $200, 400, .., 5000$, and for $b = 500, s = 500$, the sizes are $500, 1000, .., 5000$. To compute an expectation of the form $\mathbb{E}_{b,s}[., n']$, we use the sizes from the larger batch, i.e., $n' \in \{500, 1000, .., 5000\}$, and map the *closest sizes* from the smaller batch to them. For ex., here are some size mappings from the small batch case to the larger one: $800 \rightarrow 1000, 1000 \rightarrow 1000, 1200 \rightarrow 1000, 1400 \rightarrow 1500, 1600 \rightarrow 1500$.

## G Always ON Mode

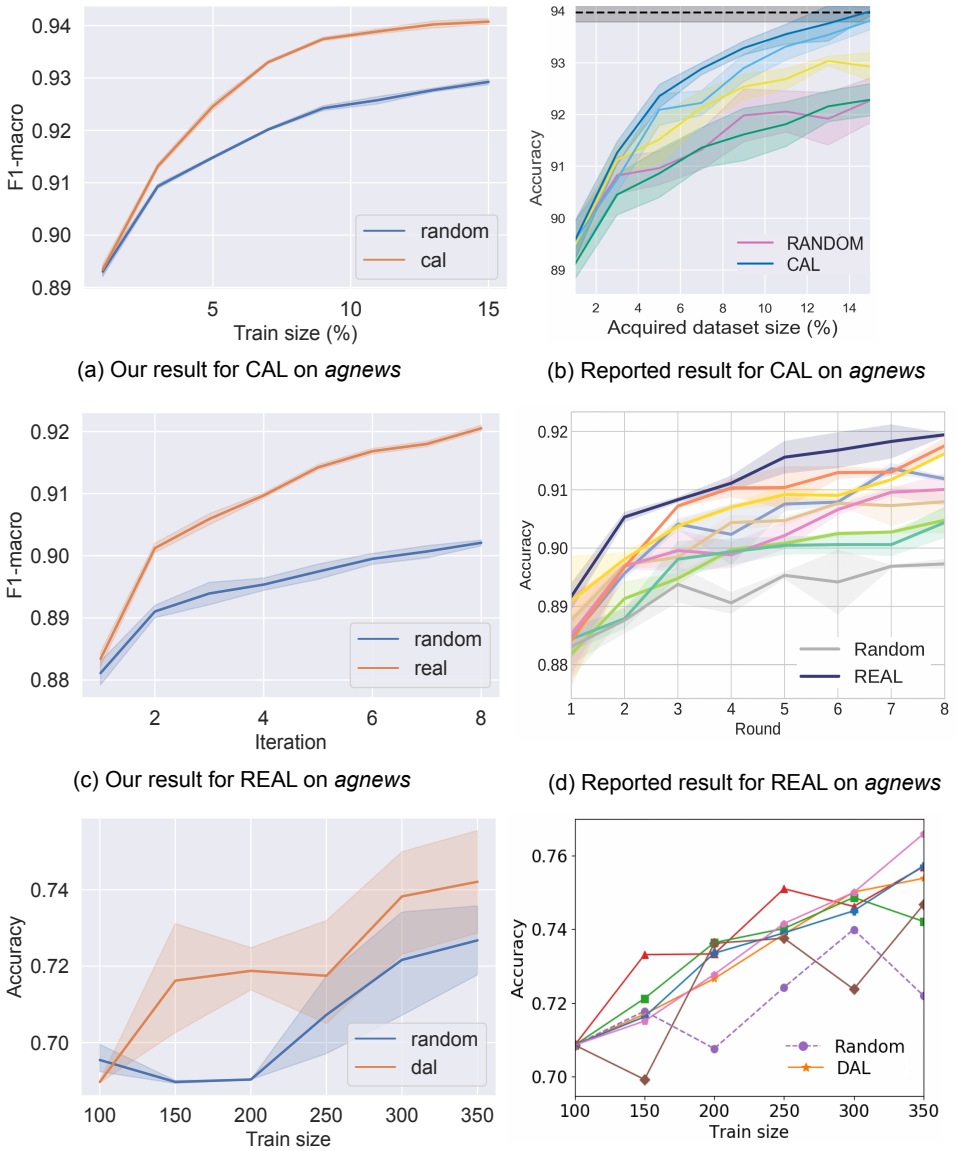
Table 6 presents standard deviations for the "Always ON" case, and is a companion to Table 1 in §5.2. Note the extremely high variances in moving across combinations of the configurations and size of the labeled set.

| Classifier | Hyperparameters |
|---|---|
| RoBERTa | roberta-base<br>[CLS] at the last hidden layer<br>learning rate = {3e-5, 5e-5}<br>train batch size = 16<br># epochs = {5, 10}<br>sequence length = 128<br>warmup ratio = 0.1<br># evaluations per epoch = 5 |
| LinearSVC | C = {0.001, 0.01, 0.1, 1, 10, 100, 1000}<br>class weight = balanced |
| RF | min samples leaf = {1, 5, 9}<br># estimators = {5, 10, 20, 30, 40, 50}<br>max depth = {5, 10, 15, 20, 25, 30}<br>class weight = balanced<br>max features = sqrt |

Table 5: Hyperparameters for each classifier in the prediction pipelines.

| *Avg. for* | % times $\delta < 0$ | $\overline{\delta}_{\geq 0}$ | $\overline{\delta}$ |
|---|---|---|---|
| **Overall** | 51.82 | 0.89 $\pm$ 0.92 | -0.74 $\pm$ 3.02 |
| LinSVC-WV | 61.71 | 0.70 $\pm$ 0.60 | -1.90 $\pm$ 3.94 |
| LinSVC-USE | 61.57 | 0.46 $\pm$ 0.49 | -0.64 $\pm$ 1.85 |
| LinSVC-MP | 63.71 | 0.40 $\pm$ 0.44 | -1.48 $\pm$ 3.53 |
| RF-WV | 47.29 | 1.31 $\pm$ 1.01 | -0.30 $\pm$ 2.63 |
| RF-USE | 60.57 | 0.71 $\pm$ 0.69 | -0.63 $\pm$ 1.85 |
| RF-MP | 60.14 | 0.60 $\pm$ 0.55 | -1.24 $\pm$ 3.59 |
| RoBERTa | 7.71 | 1.29 $\pm$ 1.17 | 1.01 $\pm$ 1.94 |
| cal | 55.60 | 0.81 $\pm$ 0.86 | -1.07 $\pm$ 3.23 |
| dal | 70.12 | 0.82 $\pm$ 0.94 | -1.29 $\pm$ 3.22 |
| margin | 38.45 | 0.97 $\pm$ 0.88 | -0.25 $\pm$ 2.78 |
| real | 43.10 | 0.89 $\pm$ 0.99 | -0.34 $\pm$ 2.67 |

Table 6: The $\%$-age of times model *F1-macro* scores are worse than random, the average $\delta$s when scores are at least as good as random and average $\delta$s in general. These are identical to the values in Table 1 in §5.2, but the standard deviations are additionally shown here.




(a) Our result for CAL on *agnews*

(b) Reported result for CAL on *agnews*

(c) Our result for REAL on *agnews*

(d) Reported result for REAL on *agnews*

(e) Our result for DAL on *cola*

(f) Reported result for DAL on *cola*

Figure 6: Comparison between published results in (Margatina et al., 2021; Chen et al., 2023; Ein-Dor et al., 2020) and ours with the same settings for CAL, REAL, DAL.