Active Learning by Acquiring Contrastive Examples

Katerina Margatina[†] **Giorgos Vernikos**[‡*] **Loïc Barrault**[†] **Nikolaos Aletras**[†]
[†]University of Sheffield [‡]EPFL [*]HEIG-VD
{k.margatina, l.barrault, n.aletras}@sheffield.ac.uk
georgios.vernikos@epfl.ch

Abstract

Common acquisition functions for active learning use either uncertainty or diversity sampling, aiming to select difficult and diverse data points from the pool of unlabeled data, respectively. In this work, leveraging the best of both worlds, we propose an acquisition function that opts for selecting *contrastive examples*, i.e. data points that are similar in the model feature space and yet the model outputs maximally different predictive likelihoods. We compare our approach, CAL (Contrastive Active Learning), with a diverse set of acquisition functions in four natural language understanding tasks and seven datasets. Our experiments show that CAL performs consistently better or equal than the best performing baseline across all tasks, on both in-domain and out-of-domain data. We also conduct an extensive ablation study of our method and we further analyze all actively acquired datasets showing that CAL achieves a better trade-off between uncertainty and diversity compared to other strategies.

1 Introduction

Active learning (AL) is a machine learning paradigm for efficiently acquiring data for annotation from a (typically large) pool of unlabeled data (Lewis and Catlett, 1994; Cohn et al., 1996; Settles, 2009). Its goal is to concentrate the human labeling effort on the most informative data points that will benefit model performance the most and thus reducing data annotation cost.

The most widely used approaches to acquiring data for AL are based on uncertainty and diversity, often described as the "two faces of AL" (Dasgupta, 2011). While uncertainty-based methods leverage the model predictive confidence to select difficult examples for annotation (Lewis and Gale, 1994; Cohn et al., 1996), diversity sampling exploits heterogeneity in the feature space by typically performing clustering (Brinker, 2003; Bodó et al., 2011). Still, both approaches have core limitations that may lead to acquiring redundant data points. Algorithms based on uncertainty may end up choosing uncertain yet uninformative repetitive data, while diversity-based methods may tend to select diverse yet easy examples for the model (Roy and McCallum, 2001). The two approaches are orthogonal to each other, since uncertainty sampling is usually based on the model's output, while diversity exploits information from the input (i.e. feature) space. Hybrid data acquisition functions that combine uncertainty and diversity sampling have also been proposed (Shen et al., 2004; Zhu et al., 2008; Ducoffe and Precioso, 2018; Ash et al., 2020; Yuan et al., 2020; Ru et al., 2020).

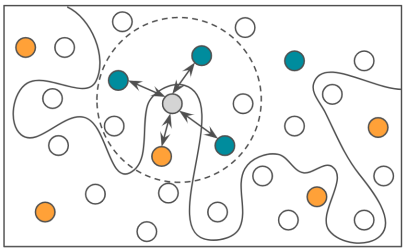

Figure 1: Illustrative example of our proposed method CAL. The solid line (model decision boundary) separates data points from two different classes (blue and orange), the coloured data points represent the labeled data and the rest are the unlabeled data of the pool.

In this work, we aim to leverage characteristics from hybrid data acquisition. We hypothesize that data points that are close in the model feature space (i.e. share similar or related vocabulary, or similar model encodings) but the model produces different predictive likelihoods, should be good candidates for data acquisition. We define such examples as *contrastive* (see example in Figure 1). For that purpose, we propose a new acquisition function that searches for contrastive examples in the pool of unlabeled data. Specifically, our method, Contrastive Active Learning (CAL) *selects unlabeled*

data points from the pool, whose predictive likelihoods diverge the most from their neighbors in the training set. This way, CAL shares similarities with diversity sampling, but instead of performing clustering it uses the feature space to create neighborhoods. CAL also leverages uncertainty, by using predictive likelihoods to rank the unlabeled data.

We evaluate our approach in seven datasets from four tasks including sentiment analysis, topic classification, natural language inference and paraphrase detection. We compare CAL against a full suite of baseline acquisition functions that are based on uncertainty, diversity or both. We also examine robustness by evaluating on out-of-domain data, apart from in-domain held-out sets. Our contributions are the following:

1. We propose CAL, a new acquisition function for active learning that acquires contrastive examples from the pool of unlabeled data (§2);

2. We show that CAL performs consistently better or equal compared to all baselines in all tasks when evaluated on in-domain and out-of-domain settings (§4);

3. We conduct a thorough analysis of our method showing that CAL achieves a better trade-off between diversity and uncertainty compared to the baselines (§6).

We release our code online [1].

2 Contrastive Active Learning

In this section we present in detail our proposed method, CAL: Contrastive Active Learning. First, we provide a definition for contrastive examples and how they are related to finding data points that are close to the decision boundary of the model (§2.1). We next describe an active learning loop using our proposed acquisition function (§2.2).

2.1 Contrastive Examples

In the context of active learning, we aim to formulate an acquisition function that selects contrastive examples from a pool of unlabeled data for annotation. We draw inspiration from the contrastive learning framework, that leverages the similarity between data points to push those from the same class closer together and examples from different classes further apart during training (Mikolov et al., 2013; Sohn, 2016; van den Oord et al., 2019; Chen et al., 2020; Gunel et al., 2021).

In this work, we define as contrastive examples two data points if their model encodings are similar, but their model predictions are very different (maximally disagreeing predictive likelihoods).

Formally, data points x_i and x_j should first satisfy a similarity criterion:

$$d\big(\Phi(x_i), \Phi(x_j)\big) < \epsilon \tag{1}$$

where $\Phi(.) \in \mathbb{R}^{d'}$ is an encoder that maps x_i, x_j in a shared feature space, $d(.)$ is a distance metric and ϵ is a small distance value.

A second criterion, based on model uncertainty, is to evaluate that the predictive probability distributions of the model $p(y|x_i)$ and $p(y|x_j)$ for the inputs x_i and x_j should maximally diverge:

$$\mathrm{KL}\big(p(y|x_i)||p(y|x_j)\big) \rightarrow \infty \tag{2}$$

where KL is the Kullback-Leibler divergence between two probability distributions [2].

For example, in a binary classification problem, given a reference example x_1 with output probability distribution $(0.8, 0.2)$ [3] and similar candidate examples x_2 with $(0.7, 0.3)$ and x_3 with $(0.6, 0.4)$, we would consider as contrastive examples the pair (x_1, x_3). However, if another example x_4 (similar to x_1 in the model feature space) had a probability distribution $(0.4, 0.6)$, then the most contrastive pair would be (x_1, x_4).

Figure 1 provides an illustration of contrastive examples for a binary classification case. All data points inside the circle (dotted line) are similar in the model feature space, satisfying Eq. 1. Intuitively, if the divergence of the output probabilities of the model for the gray and blue shaded data points is high, then Eq. 2 should also hold and we should consider them as contrastive.

From a different perspective, data points with similar model encodings (Eq. 1) and dissimilar model outputs (Eq. 2), should be close to the model's decision boundary (Figure 1). Hence, we hypothesize that our proposed approach to select

[1] https://github.com/mourga/contrastive-active-learning

[2] KL divergence is not a symmetric metric, $\mathrm{KL}(P||Q) = \sum_x P(x)\log\big(\frac{P(x)}{Q(x)}\big)$. We use as input Q the output probability distribution of an unlabeled example from the pool and as target P the output probability distribution of an example from the train set (See §2.2 and algorithm 1).

[3] A predictive distribution $(0.8, 0.2)$ here denotes that the model is 80% confident that x_1 belongs to the first class and 20% to the second.

Algorithm 1 Single iteration of CAL

Input: labeled data $\mathcal{D}_{\textbf{lab}}$, unlabeled data $\mathcal{D}_{\textbf{pool}}$, acquisition size b, model $\mathcal{M}$, number of neighbours k, model representation (encoding) function $\Phi(.)$

1 **for** x_p *in* $\mathcal{D}_{\textbf{pool}}$ **do**

2 $\quad \{(x_l^{(i)}, y_l^{(i)})\}, i = 1, ..., k \leftarrow \text{KNN}(\Phi(x_p), \Phi(\mathcal{D}_{\textbf{lab}}), k)$ ▷ find neighbours in $\mathcal{D}_{\textbf{lab}}$

3 $\quad p(y|x_l^{(i)}) \leftarrow \mathcal{M}(x_l^{(i)}), i = 1, ..., k$ ▷ compute probabilities

4 $\quad p(y|x_p) \leftarrow \mathcal{M}(x_p)$

5 $\quad \text{KL}(p(y|x_l^{(i)})||p(y|x_p)), i = 1, ..., k$ ▷ compute divergence

6 $\quad s_{x_p} = \frac{1}{k} \sum_{i=1}^{k} \text{KL}(p(y|x_l^{(i)})||p(y|x_p))$ ▷ score

7 **end**

8 $Q = \underset{x_p \in \mathcal{D}_{\textbf{pool}}}{\text{argmax}} s_{x_p}, |Q| = b$ ▷ select batch

Output: Q

contrastive examples is related to acquiring difficult examples near the decision boundary of the model. Under this formulation, CAL does not guarantee that the contrastive examples lie near the model's decision boundary, because our definition is not strict. In order to ensure that a pair of contrastive examples lie on the boundary, the second criterion should require that the model classifies the two examples in different classes (i.e. different predictions). However, calculating the distance between an example and the model decision boundary is intractable and approximations that use adversarial examples are computationally expensive (Ducoffe and Precioso, 2018).

2.2 Active Learning Loop

Assuming a multi-class classification problem with C classes, labeled data for training $\mathcal{D}_{\textbf{lab}}$ and a pool of unlabeled data $\mathcal{D}_{\textbf{pool}}$, we perform AL for T iterations. At each iteration, we train a model on $\mathcal{D}_{\textbf{lab}}$ and then use our proposed acquisition function, CAL (Algorithm 1), to acquire a batch Q consisting of b examples from $\mathcal{D}_{\textbf{pool}}$. The acquired examples are then labeled[4], they are removed from the pool $\mathcal{D}_{\textbf{pool}}$ and added to the labeled dataset $\mathcal{D}_{\textbf{lab}}$, which will serve as the training set for training a model in the next AL iteration. In our experiments, we use a pretrained BERT model $\mathcal{M}$ (Devlin et al., 2019), which we fine-tune at each AL iteration using the current $\mathcal{D}_{\textbf{lab}}$. We begin the AL loop by training a model $\mathcal{M}$ using an initial labeled dataset $\mathcal{D}_{\textbf{lab}}$ [5].

Find Nearest Neighbors for Unlabeled Candidates The first step of our contrastive acquisition function (cf. line 2) is to find examples that are similar in the model feature space (Eq. 1). Specifically, we use the `[CLS]` token embedding of BERT as our encoder $\Phi(.)$ to represent all data points in $\mathcal{D}_{\textbf{lab}}$ and $\mathcal{D}_{\textbf{pool}}$. We use a K-Nearest-Neighbors (KNN) implementation using the labeled data $\mathcal{D}_{\textbf{lab}}$, in order to query similar examples $x_l \in \mathcal{D}_{\textbf{lab}}$ for each candidate $x_p \in \mathcal{D}_{\textbf{pool}}$. Our distance metric $d(.)$ is Euclidean distance. To find the most similar data points in $\mathcal{D}_{\textbf{lab}}$ for each x_p, we select the top k instead of selecting a predefined threshold ϵ (Eq. 1) [6]. This way, we create a neighborhood $N_{x_p} = \{x_p, x_l^{(1)}, \ldots, x_l^{(k)}\}$ that consists of the unlabeled data point x_p and its k closest examples x_l in $\mathcal{D}_{\textbf{lab}}$ (Figure 1).

Compute Contrastive Score between Unlabeled Candidates and Neighbors In the second step, we compute the divergence in the model predictive probabilities for the members of the neighborhood (Eq. 2). Using the current trained model $\mathcal{M}$ to obtain the output probabilities for all data points in N_{x_p} (cf. lines 3-4), we then compute the Kullback–Leibler divergence (KL) between the output probabilities of x_p and all $x_l \in N_{x_p}$ (cf. line 5). To obtain a score s_{x_p} for a candidate x_p, we take the average of all divergence scores (cf. line 6).

Rank Unlabeled Candidates and Select Batch We apply these steps to all candidate examples $x_p \in \mathcal{D}_{\textbf{pool}}$ and obtain a score s_{x_p} for each. With

[4] We simulate AL, so we already have the labels of the examples of $\mathcal{D}_{\textbf{pool}}$ (but still treat it as an unlabeled dataset).

[5] We acquire the first examples that form the initial training set $\mathcal{D}_{\textbf{lab}}$ by applying random stratified sampling (i.e. keeping the initial label distribution).

[6] We leave further modifications of our scoring function as future work. One approach would be to add the average distance from the neighbors (cf. line 6) in order to alleviate the possible problem of selecting outliers.

DATASET	TASK	DOMAIN	OOD DATASET	TRAIN	VAL	TEST	CLASSES
IMDB	Sentiment Analysis	Movie Reviews	SST-2	22.5K	2.5K	25K	2
SST-2	Sentiment Analysis	Movie Reviews	IMDB	60.6K	6.7K	871	2
AGNEWS	Topic Classification	News	-	114K	6K	7.6K	4
DBPEDIA	Topic Classification	News	-	20K	2K	70K	14
PUBMED	Topic Classification	Medical	-	180K	30.2K	30.1K	5
QNLI	Natural Language Inference	Wikipedia	-	99.5K	5.2K	5.5K	2
QQP	Paraphrase Detection	Social QA Questions	TWITTERPPDB	327K	36.4K	80.8K	2

Table 1: Dataset statistics.

our scoring function we define as contrastive examples the unlabeled data x_p that have the highest score s_{x_p}. A high s_{x_p} score indicates that the unlabeled data point x_p has a high divergence in model predicted probabilities compared to its neighbors in the training set (Eq. 1, 2), suggesting that it may lie near the model's decision boundary. To this end, our acquisition function selects the top b examples from the pool that have the highest score s_{x_p} (cf. line 8), that form the acquired batch Q.

3 Experimental Setup

3.1 Tasks & Datasets

We conduct experiments on sentiment analysis, topic classification, natural language inference and paraphrase detection tasks. We provide details for the datasets in Table 1. We follow Yuan et al. (2020) and use IMDB (Maas et al., 2011), SST-2 (Socher et al., 2013), PUBMED (Dernoncourt and Lee, 2017) and AGNEWS from Zhang et al. (2015) where we also acquired DBPEDIA. We experiment with tasks requiring pairs of input sequences, using QQP and QNLI from GLUE (Wang et al., 2019). To evaluate robustness on out-of-distribution (OOD) data, we follow Hendrycks et al. (2020) and use SST-2 as OOD dataset for IMDB and vice versa. We finally use TWITTERPPDB (Lan et al., 2017) as OOD data for QQP as in Desai and Durrett (2020).

3.2 Baselines

We compare CAL against five baseline acquisition functions. The first method, ENTROPY is the most commonly used uncertainty-based baseline that acquires data points for which the model has the highest predictive entropy. As a diversity-based baseline, following Yuan et al. (2020), we use BERTKM that applies k-means clustering using the l_2 normalized BERT output embeddings of the fine-tuned model to select b data points. We compare against BADGE (Ash et al., 2020), an acquisition function that aims to combine diversity and uncertainty sampling, by computing *gradient embeddings* g_x for every candidate data point x in $\mathcal{D}_{\textbf{pool}}$ and then using clustering to select a batch. Each g_x is computed as the gradient of the cross-entropy loss with respect to the parameters of the model's last layer, aiming to be the component that incorporates uncertainty in the acquisition function [7]. We also evaluate a recently introduced cold-start acquisition function called ALPS (Yuan et al., 2020) that uses the masked language model (MLM) loss of BERT as a proxy for model uncertainty in the downstream classification task. Specifically, aiming to leverage both uncertainty and diversity, ALPS forms a *surprisal embedding* s_x for each x, by passing the unmasked input x through the BERT MLM head to compute the cross-entropy loss for a random 15% subsample of tokens against the target labels. ALPS clusters these embeddings to sample b sentences for each AL iteration. Lastly, we include RANDOM, that samples data from the pool from a uniform distribution.

3.3 Implementation Details

We use BERT-BASE (Devlin et al., 2019) adding a task-specific classification layer using the implementation from the HuggingFace library (Wolf et al., 2020). We evaluate the model 5 times per epoch on the development set following Dodge et al. (2020) and keep the one with the lowest validation loss. We use the standard splits provided for all datasets, if available, otherwise we randomly sample a validation set from the training set. We test all models on a held-out test set. We repeat all experiments with five different random seeds resulting into different initializations of the parameters of the model's extra task-specific output feedfor-

[7] We note that BERTKM and BADGE are computationally heavy approaches that require clustering of vectors with high dimensionality, while their complexity grows exponentially with the acquisition size. We thus do not apply them to the datasets that have a large $\mathcal{D}_{\textbf{pool}}$. More details can be found in the Appendix A.2

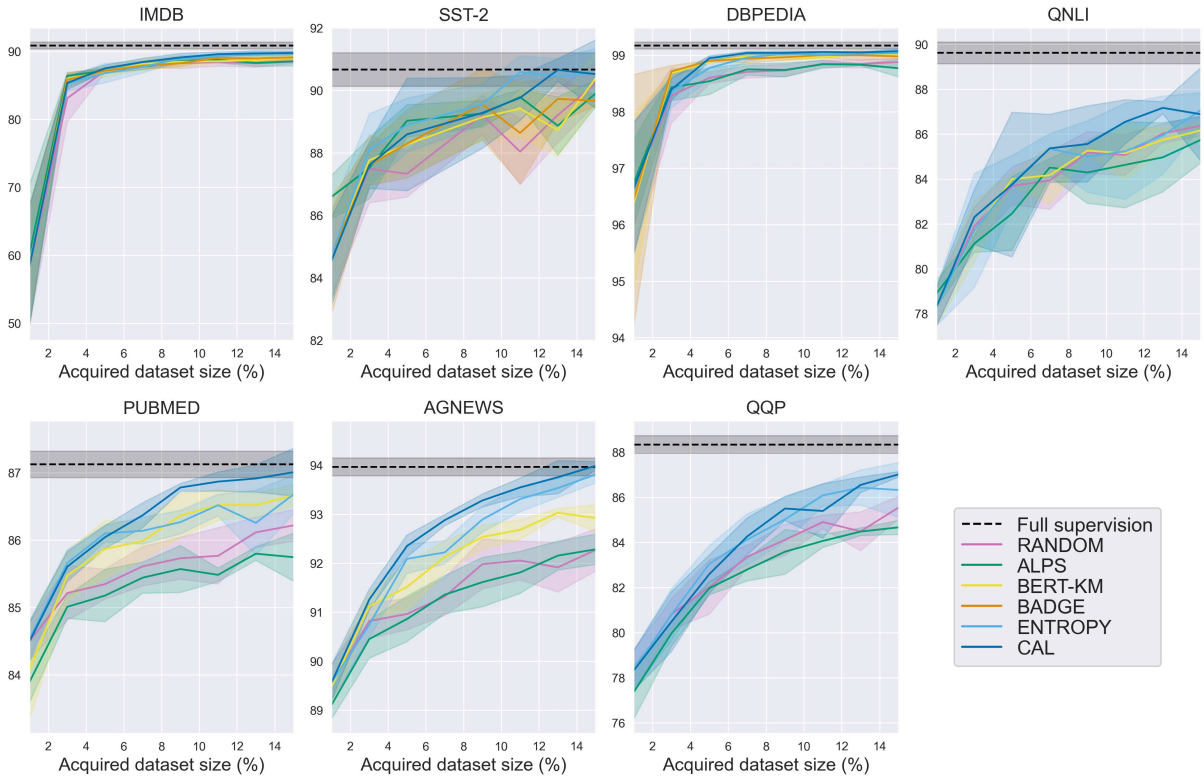

Figure 2: In-domain (ID) test accuracy during AL iterations for different acquisition functions.

ward layer and the initial $\mathcal{D}_{\textbf{lab}}$. For all datasets we use as budget the 15% of $\mathcal{D}_{\textbf{pool}}$, initial training set 1% and acquisition size $b = 2\%$. Each experiment is run on a single Nvidia Tesla V100 GPU. More details are provided in the Appendix A.1.

4 Results

4.1 In-domain Performance

We present results for in-domain test accuracy across all datasets and acquisition functions in Figure 2. We observe that CAL is consistently the top performing method especially in DBPEDIA, PUBMED and AGNEWS datasets.

CAL performs slightly better than ENTROPY in IMDB, QNLI and QQP, while in SST-2 most methods yield similar results. ENTROPY is the second best acquisition function overall, consistently performing better than diversity-based or hybrid baselines. This corroborates recent findings from Desai and Durrett (2020) that BERT is sufficiently calibrated (i.e. produces good uncertainty estimates), making it a tough baseline to beat in AL.

BERTKM is a competitive baseline (e.g. SST-2, QNLI) but always underperforms compared to CAL and ENTROPY, suggesting that uncertainty is the most important signal in the data selection process. An interesting future direction would be to investigate in depth whether and which (i.e. which layer) representations of the current (pretrained language models) works best with similarity search algorithms and clustering.

Similarly, we can see that BADGE, despite using both uncertainty and diversity, also achieves low performance, indicating that clustering the constructed gradient embeddings does not benefit data acquisition. Finally, we observe that ALPS generally underperforms and is close to RANDOM. We can conclude that this heterogeneous approach to uncertainty, i.e. using the pretrained language model as proxy for the downstream task, is beneficial only in the first few iterations, as shown in Yuan et al. (2020).

Surprisingly, we observe that for the SST-2 dataset ALPS performs similarly with the highest performing acquisition functions, CAL and ENTROPY. We hypothesize that due to the informal textual style of the reviews of SST-2 (noisy social media data), the pretrained BERT model can be used as a signal to query linguistically hard examples, that benefit the downstream sentiment analysis task. This is an interesting finding and a future research direction would be to investigate the correlation between the difficulty of an example in a

TRAIN (ID)	SST-2	IMDB	QQP
TEST (OOD)	IMDB	SST-2	TWITTERPPDB
RANDOM	76.28 ± 0.72	82.50 ± 3.61	85.86 ± 0.48
BERTKM	75.99 ± 1.01	84.98 ± 1.22	-
ENTROPY	75.38 ± 2.04	**85.54** ± 2.52	85.06 ± 1.96
ALPS	77.06 ± 0.78	83.65 ± 3.17	84.79 ± 0.49
BADGE	76.41 ± 0.92	85.19 ± 3.01	-
CAL	**79.00** ± 1.39	84.96 ± 2.36	**86.20** ± 0.22

Table 2: Out-of-domain (OOD) accuracy of models trained with the actively acquired datasets created with different AL acquisition strategies.

downstream task with its perplexity (loss) of the pretrained language model.

4.2 Out-of-domain Performance

We also evaluate the out-of-domain (OOD) robustness of the models trained with the actively acquired datasets of the last iteration (i.e. 15% of $\mathcal{D}_{\textbf{pool}}$ or 100% of the AL budget) using different acquisition strategies. We present the OOD results for SST-2, IMDB and QQP in Table 2. When we test the models trained with SST-2 on IMDB (first column) we observe that CAL achieves the highest performance compared to the other methods by a large margin, indicating that acquiring contrastive examples can improve OOD generalization. In the opposite scenario (second column), we find that the highest accuracy is obtained with ENTROPY. However, similarly to the ID results for SST-2 (Figure 2), all models trained on different subsets of the IMDB dataset result in comparable performance when tested on the small SST-2 test set (the mean accuracies lie inside the standard deviations across models). We hypothesize that this is because SST-2 is not a challenging OOD dataset for the different IMDB models. This is also evident by the high OOD accuracy, 85% on average, which is close to the 91% SST-2 ID accuracy of the full model (i.e. trained on 100% of the ID data). Finally, we observe that CAL obtains the highest OOD accuracy for QQP compared to RANDOM, ENTROPY and ALPS. Overall, our empirical results show that the models trained on the actively acquired dataset with CAL obtain consistently similar or better performance than all other approaches when tested on OOD data.

5 Ablation Study

We conduct an extensive ablation study in order to provide insights for the behavior of every component of CAL. We present all AL experiments on the AGNEWS dataset in Figure 3.

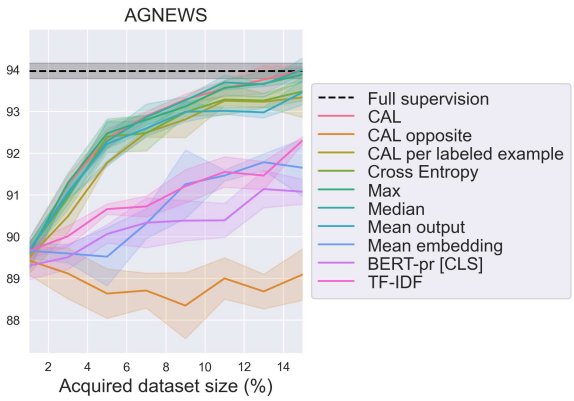

Figure 3: In-domain (ID) test accuracy with different variants of CAL (ablation).

Decision Boundary We first aim to evaluate our hypothesis that CAL acquires difficult examples that lie close to the model's decision boundary. Specifically, to validate that the ranking of the constructed neighborhoods is meaningful, we run an experiment where we acquire candidate examples that have the *minimum* divergence from their neighbors opposite to CAL (i.e. we replace $\text{argmax}(.)$ with $\text{argmin}(.)$ in line 8 of Algorithm 1). We observe (Fig. 3 - `CAL opposite`) that even after acquiring 15% of unlabeled data, the performance remains unchanged compared to the initial model (of the first iteration), even degrades. In effect, this finding denotes that CAL does select informative data points.

Neighborhood Next, we experiment with changing the way we construct the neighborhoods, aiming to improve computational efficiency. We thus modify our algorithm to create a neighborhood for each *labeled* example (instead of unlabeled).[8]. This way we compute a divergence score only for the neighbors of the training data points. However, we find this approach to slightly underperform (Fig. 3 - `CAL per labeled example`), possibly because only a small fraction of the pool is considered and thus the uncertainty of all the unlabeled data points is not taken into account.

[8] In this experiment, we essentially change the *for-loop* of Algorithm 1 (cf. line 1-7) to iterate for each x_l in $\mathcal{D}_{\textbf{lab}}$ (instead of each x_p in $\mathcal{D}_{\textbf{pool}}$) and similarly find the k nearest neighbors of each labeled example in the pool ($\text{KNN}(x_l, \mathcal{D}_{\textbf{pool}}, k)$) As for the scoring (cf. line 6), if an unlabeled example was not picked (i.e. was not a neighbor to a labeled example), its score is zero. If it was picked multiple times we average its scores. We finally acquire the top b unlabeled data with the highest scores. This formulation is more computationally efficient since usually $|\mathcal{D}_{\textbf{lab}}| << |\mathcal{D}_{\textbf{pool}}|$.

Scoring function We also experiment with several approaches for constructing our scoring function (cf. line 6 in Algorithm 1). Instead of computing the KL divergence between the predicted probabilities of each candidate example and its labeled neighbors (cf. line 5), we used cross entropy between the output probability distribution and the gold labels of the labeled data. The intuition is to evaluate whether information of the actual label is more useful than the model's predictive probability distribution. We observe this scoring function to result in a slight drop in performance (Fig. 3 - `Cross Entropy`). We also experimented with various pooling operations to aggregate the KL divergence scores for each candidate data point. We found maximum and median (Fig. 3 - `Max/Median`) to perform similarly with the average (Fig. 3 - CAL), which is the pooling operation we decided to keep in our proposed algorithm.

Feature Space Since our approach is related to to acquiring data near the model's decision boundary, this effectively translates into using the `[CLS]` output embedding of BERT. Still, we opted to cover several possible alternatives to the representations, i.e. feature space, that can be used to find the neighbors with KNN. We divide our exploration into two categories: intrinsic representations from the current fine-tuned model and extrinsic using different methods. For the first category, we examine representing each example with the mean embedding layer of BERT (Fig. 3 - `Mean embedding`) or the mean output embedding (Fig. 3 - `Mean output`). We find both alternatives to perform worse than using the `[CLS]` token (Fig. 3 - CAL). The motivation for the second category is to evaluate whether acquiring contrastive examples in the *input* feature space, i.e. representing the raw text, is meaningful (Gardner et al., 2020) [9]. We thus examine contextual representations from a pretrained BERT language model (Fig. 3 - `BERT-pr [CLS]`) (not fine-tuned in the task or domain) and non-contextualized TF-IDF vectors (Fig. 3 - `TF-IDF`). We find both approaches, along with `Mean embedding`, to largely underperform compared to our approach that acquires ambiguous data near the model decision boundary.

[9] This can be interpreted as comparing the effectiveness of selecting data near the *model* decision boundary vs. the *task* decision boundary, i.e. data that are similar for the task itself or for the humans (in terms of having the same raw input/vocabulary), but are from different classes.

6 Analysis

Finally, we further investigate CAL and all acquisition functions considered (baselines), in terms of diversity, representativeness and uncertainty. Our aim is to provide insights on what data each method tends to select and what is the uncertainty-diversity trade-off of each approach. Table 3 shows the results of our analysis averaged across datasets. We denote with L the labeled set, U the unlabeled pool and Q an acquired batch of data points from U [10].

6.1 Diversity & Uncertainty Metrics

Diversity in input space (DIV.-I) We first evaluate the diversity of the actively acquired data in the input feature space, i.e. raw text, by measuring the overlap between tokens in the sampled sentences Q and tokens from the rest of the data pool U. Following Yuan et al. (2020), we compute DIV.-I as the Jaccard similarity between the set of tokens from the sampled sentences Q, $\mathcal{V}_Q$, and the set of tokens from the unsampled sentences $\mathcal{U}\backslash\mathcal{Q}$, $\mathcal{V}_{Q'}$, $\mathcal{J}(\mathcal{V}_Q, \mathcal{V}_{Q'}) = \frac{|\mathcal{V}_Q \cap \mathcal{V}_{Q'}|}{|\mathcal{V}_Q \cup \mathcal{V}_{Q'}|}$. A high DIV.-I value indicates high diversity because the sampled and unsampled sentences have many tokens in common.

Diversity in feature space (DIV.-F) We next evaluate diversity in the (model) feature space, using the `[CLS]` representations of a trained BERT model [11]. Following Zhdanov (2019) and Ein-Dor et al. (2020), we compute DIV.-F of a set Q as $\left(\frac{1}{|U|}\sum_{x_i \in U} \min_{x_j \in Q} d(\Phi(x_i), \Phi(x_j))\right)^{-1}$, where $\Phi(x_i)$ denotes the `[CLS]` output token of example x_i obtained by the model which was trained using L, and $d(\Phi(x_i), \Phi(x_j))$ denotes the Euclidean distance between x_i and x_j in the feature space.

Uncertainty (UNC.) To measure uncertainty, we use the model $\mathcal{M}_f$ trained on the entire training dataset (Figure 2 - `Full supervision`). As in Yuan et al. (2020), we use the logits from the fully trained model to estimate the uncertainty of an example, as it is a reliable estimate due to its high performance after training on many examples, while

[10] In the previous sections we used $\mathcal{D}_{\textbf{lab}}$ and $\mathcal{D}_{\textbf{pool}}$ to denote the labeled and unlabeled sets and we change the notation here to L and U, respectively, for simplicity.

[11] To enable an appropriate comparison, this analysis is performed after the initial BERT model is trained with the initial training set and each AL strategy has selected examples equal to 2% of the pool (first iteration). Correspondingly, all strategies select examples from the same unlabeled set U while using outputs from the same BERT model.

	DIV.-I	DIV.-F	UNC.	REPR.
RANDOM	0.766	0.356	0.132	1.848
BERTKM	0.717	**0.363**	0.145	2.062
ENTROPY	0.754	0.323	**0.240**	2.442
ALPS	**0.771**	0.360	0.126	2.038
BADGE	0.655	0.339	0.123	2.013
CAL	0.768	0.335	0.231	**2.693**

Table 3: Uncertainty and diversity metrics across acquisition functions, averaged for all datasets.

it offers a fair comparison across all acquisition strategies. First, we compute predictive entropy of an input x when evaluated by model $\mathcal{M}_f$ and then we take the average over all sentences in a sampled batch Q. We use the average predictive entropy to estimate uncertainty of the acquired batch Q for each method $-\frac{1}{|Q|}\sum_{x\in Q}\sum_{c=1}^{C} p(y=c|x)\log p(y=c|x)$. As a sampled batch Q we use the full actively acquired dataset after completing our AL iterations (with 15% of the data).

Representativeness (REPR.) We finally analyze the representativeness of the acquired data as in Ein-Dor et al. (2020). We aim to study whether AL strategies tend to select outlier examples that do not properly represent the overall data distribution. We rely on the KNN-density measure proposed by Zhu et al. (2008), where the density of an example is quantified by one over the average distance between the example and its K most similar examples (i.e., K nearest neighbors) within U, based on the `[CLS]` representations as in DIV.-F. An example with high density degree is less likely to be an outlier. We define the representativeness of a batch Q as one over the average KNN-density of its instances using the Euclidean distance with K=10.

6.2 Discussion

We first observe in Table 3 that ALPS acquires the most diverse data across all approaches. This is intuitive since ALPS is the most linguistically-informed method as it essentially acquires data that are difficult for the language modeling task, thus favoring data with a more diverse vocabulary. All other methods acquire similarly diverse data, except BADGE that has the lowest score. Interestingly, we observe a different pattern when evaluating diversity in the model feature space (using the `[CLS]` representations). BERTKM has the highest DIV.-F score, as expected, while CAL and ENTROPY have the lowest. This supports our hypothesis that uncertainty sampling tends to acquire uncertain but similar examples, while CAL by definition constrains its search in similar examples in the feature space that lie close to the decision boundary (contrastive examples). As for uncertainty, we observe that ENTROPY and CAL acquire the most uncertain examples, with average entropy almost twice as high as all other methods. Finally, regarding representativeness of the acquired batches, we see that CAL obtains the highest score, followed by ENTROPY, with the rest AL strategies to acquire less representative data.

Overall, our analysis validates assumptions on the properties of data expected to be selected by the various acquisition functions. Our findings show that diversity in the raw text does not necessarily correlate with diversity in the feature space. In other words, low DIV.-F does not translate to low diversity in the distribution of acquired tokens (DIV.-I), suggesting that CAL can acquire similar examples in the feature space that have sufficiently diverse inputs. Furthermore, combining the results of our AL experiments (Figure 2) and our analysis (Table 3) we conclude that the best performance of CAL, followed by ENTROPY, is due to acquiring uncertain data. We observe that the most notable difference, in terms of selected data, between the two approaches and the rest is uncertainty (UNC.), suggesting perhaps the superiority of uncertainty over diversity sampling. We show that CAL improves over ENTROPY because our algorithm "guides" the focus of uncertainty sampling by not considering redundant uncertain data that lie away from the decision boundary and thus improving representativeness. We finally find that RANDOM is evidently the worst approach, as it selects the least diverse and uncertain data on average compared to all methods.

7 Related Work

Uncertainty Sampling Uncertainty-based acquisition for AL focuses on selecting data points that the model predicts with low confidence. A simple uncertainty-based acquisition function is *least confidence* (Lewis and Gale, 1994) that sorts data in descending order from the pool by the probability of not predicting the most confident class. Another approach is to select samples that maximize the predictive entropy. Houlsby et al. (2011)

propose Bayesian Active Learning by Disagreement (BALD), a method that chooses data points that maximize the mutual information between predictions and model's posterior probabilities. Gal et al. (2017) applied BALD for deep neural models using Monte Carlo dropout (Gal and Ghahramani, 2016) to acquire multiple uncertainty estimates for each candidate example. Least confidence, entropy and BALD acquisition functions have been applied in a variety of text classification and sequence labeling tasks, showing to substantially improve data efficiency (Shen et al., 2017; Siddhant and Lipton, 2018; Lowell and Lipton, 2019; Kirsch et al., 2019; Shelmanov et al., 2021; Margatina et al., 2021).

Diversity Sampling On the other hand, diversity or representative sampling is based on selecting batches of unlabeled examples that are representative of the unlabeled pool, based on the intuition that a representative set of examples once labeled, can act as a surrogate for the full data available. In the context of deep learning, Geifman and El-Yaniv (2017) and Sener and Savarese (2018) select representative examples based on core-set construction, a fundamental problem in computational geometry. Inspired by generative adversarial learning, Gissin and Shalev-Shwartz (2019) define AL as a binary classification task with an adversarial classifier trained to not be able to discriminate data from the training set and the pool. Other approaches based on adversarial active learning, use out-of-the-box models to perform adversarial attacks on the training data, in order to approximate the distance from the decision boundary of the model (Ducoffe and Precioso, 2018; Ru et al., 2020).

Hybrid There are several existing approaches that combine representative and uncertainty sampling. Such approaches include active learning algorithms that use meta-learning (Baram et al., 2004; Hsu and Lin, 2015) and reinforcement learning (Fang et al., 2017; Liu et al., 2018), aiming to learn a policy for switching between a diversity-based or an uncertainty-based criterion at each iteration. Recently, Ash et al. (2020) propose Batch Active learning by Diverse Gradient Embeddings (BADGE) and Yuan et al. (2020) propose Active Learning by Processing Surprisal (ALPS), a cold-start acquisition function specific for pretrained language models. Both methods construct representations for the unlabeled data based on uncertainty, and then use them for clustering; hence combining both uncertainty and diversity sampling. The effectiveness of AL in a variety of NLP tasks with pretrained language models, e.g. BERT (Devlin et al., 2019), has empirically been recently evaluated by Ein-Dor et al. (2020), showing substantial improvements over random sampling.

8 Conclusion & Future Work

We present CAL, a novel acquisition function for AL that acquires *contrastive examples*; data points which are similar in the model feature space and yet the model outputs maximally different class probabilities. Our approach uses information from the feature space to create neighborhoods for each unlabeled example, and predictive likelihood for ranking the candidate examples. Empirical experiments on various in-domain and out-of-domain scenarios demonstrate that CAL performs better than other acquisition functions in the majority of cases. After analyzing the actively acquired datasets obtained with all methods considered, we conclude that entropy is the hardest baseline to beat, but our approach improves it by guiding uncertainty sampling in regions near the decision boundary with more informative data.

Still, our empirical results and analysis show that there is no single acquisition function to outperform all others consistently *by a large margin*. This demonstrates that there is still room for improvement in the AL field.

Furthermore, recent findings show that in specific tasks, as in Visual Question Answering (VQA), complex acquisition functions might not outperform random sampling because they tend to select *collective outliers* that hurt model performance (Karamcheti et al., 2021). We believe that taking a step back and analyzing the behavior of standard acquisition functions, e.g. with Dataset Maps (Swayamdipta et al., 2020), might be beneficial. Especially, if similar behavior appears in other NLP tasks too.

Another interesting future direction for CAL, related to interpretability, would be to evaluate whether acquiring contrastive examples for the *task* (Kaushik et al., 2020; Gardner et al., 2020) is more beneficial than contrastive examples for the *model*, as we do in CAL.

Acknowledgments

KM and NA are supported by Amazon through the Alexa Fellowship scheme.

References

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. 2020. Deep batch active learning by diverse, uncertain gradient lower bounds. In *Proceedings of the International Conference on Learning Representations*.

Yoram Baram, Ran El-Yaniv, and Kobi Luz. 2004. Online choice of active learning algorithms. *Journal of Machine Learning Research*, 5:255–291.

Zalán Bodó, Zsolt Minier, and Lehel Csató. 2011. Active learning with clustering. In *Proceedings of the Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, volume 16, pages 127–139.

Klaus Brinker. 2003. Incorporating diversity in active learning with support vector machines. In *Proceedings of the International Conference on Machine Learning*, pages 59–66.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the International Conference on Machine Learning*, volume 119, pages 1597–1607.

David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. 1996. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4(1):129–145.

Sanjoy Dasgupta. 2011. Two faces of active learning. *Theoretical Computer Science*, 412(19):1767–1781. Algorithmic Learning Theory (ALT 2009).

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

Shrey Desai and Greg Durrett. 2020. Calibration of pre-trained transformers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 295–302.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *ArXiv*.

Melanie Ducoffe and Frederic Precioso. 2018. Adversarial active learning for deep networks: a margin based approach.

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 Conference on Empirical Methods in Natural Language Processing*, pages 7949–7962.

Meng Fang, Yuan Li, and Trevor Cohn. 2017. Learning how to active learn: A deep reinforcement learning approach. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 595–605.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the International Conference on Machine Learning*, volume 48, pages 1050–1059.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep Bayesian active learning with image data. In *Proceedings of the International Conference on Machine Learning*, volume 70, pages 1183–1192.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323.

Yonatan Geifman and Ran El-Yaniv. 2017. Deep active learning over the long tail. *CoRR*, abs/1711.00941.

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

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2021. Supervised contrastive learning for pre-trained language model fine-tuning. In *Proceedings of the International Conference on Learning Representations*.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 2744–2751.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. 2011. Bayesian active learning for classification and preference learning. *ArXiv*.

Wei-Ning Hsu and Hsuan-Tien Lin. 2015. Active learning by learning. In *Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence*, pages 2659–2665.

Siddharth Karamcheti, Ranjay Krishna, Li Fei-Fei, and Christopher Manning. 2021. Mind your outliers! investigating the negative impact of outliers on active learning for visual question answering. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*, pages 7265–7281.

Divyansh Kaushik, Eduard Hovy, and Zachary Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data. In *Proceedings of the International Conference on Learning Representations*.

Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. 2019. BatchBALD: Efficient and diverse batch acquisition for deep bayesian active learning. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 7026–7037.

Wuwei Lan, Siyu Qiu, Hua He, and Wei Xu. 2017. A continuously growing dataset of sentential paraphrases. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1224–1234.

David D. Lewis and Jason Catlett. 1994. Heterogeneous uncertainty sampling for supervised learning. In *Machine Learning Proceedings 1994*, pages 148–156.

David D. Lewis and William A. Gale. 1994. A sequential algorithm for training text classifiers. In *In Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Ming Liu, Wray Buntine, and Gholamreza Haffari. 2018. Learning how to actively learn: A deep imitation learning approach. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1874–1883.

David Lowell and Zachary C Lipton. 2019. Practical obstacles to deploying active learning. *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing*, pages 21–30.

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 Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150.

Katerina Margatina, Loïc Barrault, and Nikolaos Aletras. 2021. Bayesian active learning with pretrained language models. *CoRR*, abs/2104.08320.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the International Conference on Neural Information Processing Systems*, page 3111–3119.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

Nicholas Roy and Andrew McCallum. 2001. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of the International Conference on Machine Learning*, page 441–448.

Dongyu Ru, Jiangtao Feng, Lin Qiu, Hao Zhou, Mingxuan Wang, Weinan Zhang, Yong Yu, and Lei Li. 2020. Active sentence learning by adversarial uncertainty sampling in discrete space. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4908–4917, Online. Association for Computational Linguistics.

Ozan Sener and Silvio Savarese. 2018. Active learning for convolutional neural networks: A core-set approach. In *Proceedings of the International Conference on Learning Representations*.

Burr Settles. 2009. Active learning literature survey. Computer sciences technical report.

Artem Shelmanov, Dmitri Puzyrev, Lyubov Kupriyanova, Denis Belyakov, Daniil Larionov, Nikita Khromov, Olga Kozlova, Ekaterina Artemova, Dmitry V. Dylov, and Alexander Panchenko. 2021. Active learning for sequence tagging with deep pre-trained models and Bayesian uncertainty estimates. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*, pages 1698–1712.

Dan Shen, Jie Zhang, Jian Su, Guodong Zhou, and Chew-Lim Tan. 2004. Multi-criteria-based active learning for named entity recognition. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 589–596.

Yanyao Shen, Hyokun Yun, Zachary Lipton, Yakov Kronrod, and Animashree Anandkumar. 2017. Deep active learning for named entity recognition. In *Proceedings of the Workshop on Representation Learning for NLP*, pages 252–256.

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 Conference on Empirical Methods in Natural Language Processing*, pages 2904–2909.

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 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642.

Kihyuk Sohn. 2016. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 9275–9293.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2019. Representation learning with contrastive predictive coding.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

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 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. 2020. Cold-start active learning through self-supervised language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7935–7948, Online. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28, pages 649–657. Curran Associates, Inc.

Fedor Zhdanov. 2019. Diverse mini-batch active learning.

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 International Conference on Computational Linguistics*, pages 1137–1144.

A Appendix

A.1 Data & Hyperparameters

In this section we provide details of all the datasets we used in this work and the hyperparparameters used for training the model. For QNLI, IMDB and SST-2 we randomly sample 10% from the training set to serve as the validation set, while for AG-NEWS and QQP we sample 5%. For the DBPEDIA dataset we undersample both training and validation datasets (from the standard splits) to facilitate our AL simulation (i.e. the original dataset consists of 560K training and 28K validation data examples). For all datasets we use the standard test set, apart from SST-2, QNLI and QQP datasets that are taken from the GLUE benchmark (Wang et al., 2019) we use the development set as the held-out test set and subsample a development set from the training set.

For all datasets we train BERT-BASE (Devlin et al., 2019) from the HuggingFace library (Wolf et al., 2020) in Pytorch (Paszke et al., 2019). We train all models with batch size 16, learning rate $2e-5$, no weight decay, AdamW optimizer with epsilon $1e-8$. For all datasets we use maximum sequence length of 128, except for IMDB that contain longer input texts, where we use 256. To ensure reproducibility and fair comparison between the various methods under evaluation, we run all experiments with the same five seeds that we randomly selected from the range $[1, 9999]$. We evaluate the model 5 times per epoch on the development set following Dodge et al. (2020) and keep the one with the lowest validation loss. We use the code provided by Yuan et al. (2020) for ALPS, BADGE and BERTKM.

A.2 Efficiency

In this section we compare the efficiency of the acquisition functions considered in our experiments. We denote m the number of labeled data in $\mathcal{D}_{\textbf{lab}}$, n the number of unlabeled data in $\mathcal{D}_{\textbf{pool}}$, C the number of classes in the downstream classification task, d the dimension of embeddings, t is fixed number of iterations for k-MEANS, l the maximum sequence length and k the acquisition size. In our experiments, following (Yuan et al., 2020), $k = 100$, $d = 768$, $t = 10$, and $l = 128$[12]. ALPS requires $\mathcal{O}(tknl)$ considering that the surprisal embeddings are computed. BERTKM and BADGE, the most computationally heavy approaches, require $\mathcal{O}(knd)$ and $\mathcal{O}(Cknd)$ respectively, given that gradient embeddings are computed for BADGE [13]. On the other hand, ENTROPY only requires n forward passes though the model, in order to obtain the logits for all the data in $\mathcal{D}_{\textbf{pool}}$. Instead, our approach, CAL, first requires $m + n$ forward passes, in order to acquire the logits and the CLS representations of the the data (in $\mathcal{D}_{\textbf{pool}}$ and $\mathcal{D}_{\textbf{lab}}$) and then one iteration for all data in $\mathcal{D}_{\textbf{pool}}$ to obtain the scores.

We present the runtimes in detail for all datasets and acquisition functions in Tables 4 and 5. First, we define the total *acquisition time* as a sum of two types of times; *inference* and *selection* time. Inference time is the time that is required in order to pass all data from the model to acquire predictions or probability distributions or model encodings (representations). This is explicitly required for the uncertainty-based methods, like ENTROPY, and our method CAL. The remaining time is considered *selection* and essentially is the time for all necessary computations in order to rank and select the b most important examples from $\mathcal{D}_{\textbf{pool}}$.

We observe in Table 4 that the diversity-based functions do not require this explicit inference time, while for ENTROPY it is the only computation that is needed (taking the argmax of a list of uncertainty scores is negligible). CAL requires both inference and selection time. We can see that inference time of CAL is a bit higher than ENTROPY because we do $m+n$ forward passes instead of n, that is equivalent to both $\mathcal{D}_{\textbf{pool}}$ and $\mathcal{D}_{\textbf{lab}}$ instead of only $\mathcal{D}_{\textbf{pool}}$. The selection time for CAL is the *for-loop* as presented in our Algorithm 1. We observe that it is often less computationally expensive than the inference step (which is a simple forward pass through the model). Still, there is room for improvement in order to reduce the time complexity of this step.

In Table 5 we present the total time for all datasets (ordered with increasing $\mathcal{D}_{\textbf{pool}}$ size) and the average time for each acquisition function, as a means to rank their efficiency. Because we do not apply all acquisition functions to all datasets we compute three different average scores in order to ensure fair comparison. AVG.-ALL is the average time across all 7 datasets and is used to compare RANDOM, ALPS, ENTROPY and CAL. AVG.-3 is the average time across the first 3 datasets (IMDB, SST-2 and DBPEDIA) and is used to compare all

[12] Except for IMDB where $l = 256$.

[13] This information is taken from Section 6 of Yuan et al. (2020).

	DBPEDIA	IMDB	SST-2	QNLI	AGNEWS	PUBMED	QQP
RANDOM	$(0, 0)$	$(0, 0)$	$(0, 0)$	$(0, 0)$	$(0, 0)$	$(0, 0)$	$(0, 0)$
ALPS	$(0, 181)$	$(0, 222)$	$(0, 733)$	$(0, 1607)$	$(0, 2309)$	$(0, 5878)$	$(0, 14722)$
BERTKM	$(0, 467)$	$(0, 431)$	$(0, 4265)$	$(0, 8138)$	$(0, 9344)$	$(0, 25965)$	$(-, -)$
BADGE	$(0, 12871)$	$(0, 3816)$	$(0, 25640)$	$(-, -)$	$(-, -)$	$(-, -)$	$(-, -)$
ENTROPY	$(103, 1)$	$(107, 0)$	$(173, 0)$	$(331, 0)$	$(402, 0)$	$(596, 0)$	$(1070, 0)$
CAL	$(133, 49)$	$(212, 61)$	$(464, 244)$	$(528, 376)$	$(656, 628)$	$(1184, 1445)$	$(1541, 2857)$

Table 4: Runtimes (in seconds) for all datasets and acquisition functions. In each cell of the table we present a tuple (i, s) where i is the *inference time* and s the *selection time*. *Inference time* is the time for the model to perform a forward pass for all the unlabeled data in $\mathcal{D}_{\textbf{pool}}$ and *selection time* is the time that each acquisition function requires to rank all candidate data points and select b for annotation (for a single iteration). Since we cannot report the runtimes for *every* model in the AL pipeline (at each iteration the size of $\mathcal{D}_{\textbf{pool}}$ changes), we provide the median.

	DBPEDIA	IMDB	SST-2	QNLI	AGNEWS	PUBMED	QQP	AVG.-ALL	AVG.-3	AVG.-6
RANDOM	0	0	0	0	0	0	0	0	0	0
ALPS	181	222	733	1607	2309	5878	14722	3664	378	1821
BERTKM	467	431	4265	8138	9344	25965	–	–	1721	8101
BADGE	12871	3816	25640	–	–	–	–	–	14109	–
ENTROPY	104	107	173	331	402	596	1070	397	128	285
CAL	182	273	708	904	1284	2629	4398	1482	387	996

Table 5: Runtimes (in seconds) for all datasets and acquisition functions. In each cell of the table we present the total acquisition time (inference and selection). AVG.-ALL shows the average acquisition time for each acquisition function for all datasets, AVG.-6. for all datasets except QQP and AVG.-3 for the 3 first datasets only (DBPEDIA, IMDB, SST-2).

acquisition functions. Finally, AVG.-6 is the average time across all datasets apart from QQP and is used to compare RANDOM, ALPS, BERTKM, ENTROPY and CAL.

We first observe that ENTROPY is overall the most efficient acquisition function. According to the AVG.-ALL column, we observe that CAL is the second most efficient function, followed by ALPS. According to the AVG.-6 we observe the same pattern, with BERTKM to be the slowest method. Finally, we compare all acquisition functions in the 3 smallest (in terms of size of $\mathcal{D}_{\textbf{pool}}$) datasets and find that ENTROPY is the fastest method followed by ALPS and CAL that require almost 3 times more computation time. The other clustering methods, BERTKM and BADGE, are significantly more computationally expensive, requiring respectively 13 and 100(!) times more time than ENTROPY.

Interestingly, we observe the effect of the acquisition size (2% of $\mathcal{D}_{\textbf{pool}}$ in our case) and the size of $\mathcal{D}_{\textbf{pool}}$ in the clustering methods. As these parameters increase, the computation of the corresponding acquisition function increases dramatically. For example, we observe that in the 3 smallest datasets that ALPS requires similar time to CAL. However, when we increase b and m (i.e. as we move from DBPEDIA with $20K$ examples in $\mathcal{D}_{\textbf{pool}}$ to QNLI with $100K$ etc - see Table 1) we observe that the acquisition time of ALPS becomes twice as much as that of CAL. For instance, in QQP with acquisition size 3270 we see that ALPS requires 14722 seconds on average, while CAL 4398. This shows that even though our approach is more computationally expensive as the size of $\mathcal{D}_{\textbf{pool}}$ increases, the complexity is linear, while for the other hybrid methods that use clustering, the complexity grows exponentially.

A.3 Reproducibility

All code for data preprocessing, model implementations, and active learning algorithms is made available at https://github.com/mourga/contrastive-active-learning. For questions regarding the implementation, please contact the first author.