Michelle Yuan, Hsuan-Tien Lin, and **Jordan Boyd-Graber**. **Cold-start Active Learning through Self-Supervised Language Modeling**. *Empirical Methods in Natural Language Processing*, 2020, 9 pages.


```
    @inproceedings{Yuan:Lin:Boyd-Graber-2020,
Title = {Cold-start Active Learning through Self-Supervised Language Modeling},
Author = {Michelle Yuan and Hsuan-Tien Lin and Jordan Boyd-Graber},
Booktitle = {Empirical Methods in Natural Language Processing},
Year = {2020},
Location = {The Cyberverse Simulacrum of Punta Cana, Dominican Republic},
Url = {http://cs.umd.edu/~jbg//docs/2020_emnlp_alps.pdf},
}
```


Accessible Abstract: Labeling data is a fundamental bottleneck in machine learning, especially for NLP, due to annotation cost and time. For medical text, obtaining labeled data is challenging because of privacy issues or shortage in expertise. Thus, active learning can be employed to recognize the most relevant examples and then query labels from an oracle. However, developing a strategy for selecting examples to label is non-trivial. Active learning is difficult to use in cold-start; all examples confuse the model because it has not trained on enough data. Fortunately, modern NLP provides an additional source of information: pre-trained language models. In our paper, we propose an active learning strategy called ALPS to find sentences that perplex the language model. We evaluate our approach on sentence classification datasets spanning across different domains. Results show that ALPS is an efficient active learning strategy that is competitive with state-of-the-art approaches.

Links:

- Video [http://youtu.be/s470dWinVNY]
- Code [https://github.com/forest-snow/alps]

Contact Jordan Boyd-Graber (jbg@boydgraber.org) for questions about this paper.

Cold-start Active Learning through Self-supervised Language Modeling

Michelle Yuan*
University of Maryland
myuan@cs.umd.edu

Hsuan-Tien Lin
National Taiwan University
htlin@csie.ntu.edu.tw

Jordan Boyd-Graber
University of Maryland
jbg@umiacs.umd.edu

Abstract

Active learning strives to reduce annotation costs by choosing the most critical examples to label. Typically, the active learning strategy is contingent on the classification model. For instance, uncertainty sampling depends on poorly calibrated model confidence scores. In the cold-start setting, active learning is impractical because of model instability and data scarcity. Fortunately, modern NLP provides an additional source of information: pre-trained language models. The pre-training loss can find examples that surprise the model and should be labeled for efficient fine-tuning. Therefore, we treat the language modeling loss as a proxy for classification uncertainty. With BERT, we develop a simple strategy based on the masked language modeling loss that minimizes labeling costs for text classification. Compared to other baselines, our approach reaches higher accuracy within less sampling iterations and computation time.

1 Introduction

Labeling data is a fundamental bottleneck in machine learning, especially for NLP, due to annotation cost and time. The goal of active learning (AL) is to recognize the most relevant examples and then query labels from an oracle. For instance, policymakers and physicians want to quickly fine-tune a text classifier to understand emerging medical conditions (Voorhees et al., 2020). Finding labeled data for medical text is challenging because of privacy issues or shortage in expertise (Dernoncourt and Lee, 2017). Using AL, they can query labels for a small subset of the most relevant documents and immediately train a robust model.

Modern transformer models dominate the leaderboards for several NLP tasks (Devlin et al., 2019; Yang et al., 2019). Yet the price of adopting transformer-based models is to use more data. If these models are not fine-tuned on enough examples, their accuracy drastically varies across different hyperparameter configurations (Dodge et al., 2020). Moreover, computational resources are a major drawback as training one model can cost thousands of dollars in cloud computing and hundreds of pounds in carbon emissions (Strubell et al., 2019). These problems motivate further work in AL to conserve resources.

Another issue is that traditional AL algorithms, like uncertainty sampling (Lewis and Gale, 1994), falter on deep models. These strategies use model confidence scores, but neural networks are poorly calibrated (Guo et al., 2017). High confidence scores do not imply high correctness likelihood, so the sampled examples are not the most uncertain ones (Zhang et al., 2017). Plus, these strategies sample one document on each iteration. The single-document sampling requires training the model after each query and increases the overall expense.

These limitations of modern NLP models illustrate a twofold effect: they show a greater need for AL *and* make AL more difficult to deploy. Ideally, AL could be most useful during low-resource situations. In reality, it is impractical to use because the AL strategy depends on warm-starting the model with information about the task (Ash and Adams, 2019). Thus, a fitting solution to AL for deep classifiers is a *cold-start* approach, one that does not rely on classification loss or confidence scores.

To develop a cold-start AL strategy, we should extract knowledge from pre-trained models like BERT (Devlin et al., 2019). The model encodes syntactic properties (Tenney et al., 2019), acts as a database for general world knowledge (Petroni et al., 2019; Davison et al., 2019), and can detect out-of-distribution examples (Hendrycks et al., 2020). Given the knowledge already encoded in pre-trained models, the annotation for a new task

*Work done while visiting National Taiwan University.

should focus on the information missing from pre-training. If a sentence contains many words that perplex the language model, then it is possibly unusual or not well-represented in the pre-training data. Thus, the self-supervised objective serves as a surrogate for classification uncertainty.

We develop ALPS (Active Learning by Processing Surprisal), an AL strategy for BERT-based models.[1] While many AL methods randomly choose an initial sample, ALPS selects the first batch of data using the masked language modeling loss. As the highest and most extensive peaks in Europe are found in the Alps, the ALPS algorithm finds examples in the data that are both surprising and substantial. To the best of our knowledge, ALPS is the first AL algorithm that only relies on a self-supervised loss function. We evaluate our approach on four text classification datasets spanning across three different domains. ALPS outperforms AL baselines in accuracy and algorithmic efficiency. The success of ALPS highlights the importance of self-supervision for cold-start AL.

2 Preliminaries

We formally introduce the setup, notation, and terminology that will be used throughout the paper.

Pre-trained Encoder Pre-training uses the language modeling loss to train encoder parameters for generalized representations. We call the model input $x = (w_i)_{i=1}^{l}$ a "sentence", which is a sequence of tokens w from a vocabulary $\mathcal{V}$ with sequence length l. Given weights W, the encoder h maps x to a d-dimensonal hidden representation $h(x; W)$. We use BERT (Devlin et al., 2019) as our data encoder, so h is pre-trained with two tasks: masked language modeling (MLM) and next sentence prediction. The embedding $h(x; W)$ is computed as the final hidden state of the `[CLS]` token in x. We also refer to $h(x; W)$ as the BERT embedding.

Fine-tuned Model We fine-tune BERT on the downstream task by training the pre-trained model and the attached sequence classification head. Suppose that f represents the model with the classification head, has parameters $\theta = (W, V)$, and maps input x to a C-dimensional vector with confidence scores for each label. Specifically, $f(x; \theta) = \sigma(V \cdot h(x; W))$ where σ is a softmax function.

Let D be the labeled data for our classification task where the labels belong to set $\mathcal{Y} = \{1, ..., C\}$. During fine-tuning, we take a base classifier f with weights W_0 from a pre-trained encoder h and fine-tune f on D for new parameters θ_t. Then, the predicted classification label is $\hat{y} = \arg\max_{y \in \mathcal{Y}} f(x; \theta_t)_y$.

[1] https://github.com/forest-snow/alps

Algorithm 1 AL for Sentence Classification

Require: Inital model $f(x; \theta_0)$ with pre-trained encoder $h(x; W_0)$, unlabeled data pool $\mathcal{U}$, number of queries per iteration k, number of iterations T, sampling algorithm $\mathcal{A}$

$\mathcal{D} = \{\}$
for iterations $t = 1, \ldots, T$ **do**
  if $\mathcal{A}$ is cold-start for iteration t **then**
    $M_t(x) = f(x; \theta_0)$
  else
    $M_t(x) = f(x; \theta_{t-1})$
  $\mathcal{Q}_t \leftarrow$ Apply $\mathcal{A}$ on model $M_t(x)$, data $\mathcal{U}$
  $\mathcal{D}_t \leftarrow$ Label queries $\mathcal{Q}_t$
  $\mathcal{D} = \mathcal{D} \cup \mathcal{D}_t$
  $\mathcal{U} = \mathcal{U} \setminus \mathcal{D}_t$
  $\theta_t \leftarrow$ Fine-tune $f(x; \theta_0)$ on $\mathcal{D}$
return $f(x; \theta_T)$

AL for Sentence Classification Assume that there is a large unlabeled dataset $U = \{(x_i)\}_{i=1}^{n}$ of n sentences. The goal of AL is to sample a subset $D \subset U$ efficiently so that fine-tuning the classifier f on subset D improves test accuracy. On each iteration t, the learner uses strategy $\mathcal{A}$ to acquire k sentences from dataset U and queries for their labels (Algorithm 1). Strategy $\mathcal{A}$ usually depends on an acquisition model M_t (Lowell and Lipton, 2019). If the strategy depends on model warm-starting, then the acquisition model M_t is f with parameters θ_{t-1} from the previous iteration. Otherwise, we assume that M_t is the pre-trained model with parameters θ_0. After T rounds, we acquire labels for Tk sentences. We provide more concrete details about AL simulation in Section 5.

3 The Uncertainty–Diversity Dichotomy

This section provides background on prior work in AL. First, we discuss two general AL strategies: uncertainty sampling and diversity sampling. Then, we explain the dichotomy between the two concepts and introduce BADGE (Ash et al., 2020), a SOTA method that attempts to resolve this issue. Finally, we focus on the limitations of BADGE and other AL strategies to give motivation for our work.

Dasgupta (2011) describes uncertainty and diversity as the "two faces of AL". While uncertainty sampling efficiently searches the hypothesis space by finding difficult examples to label, diversity sampling exploits heterogeneity in the feature space (Xu et al., 2003; Hu et al., 2010; Bodó et al., 2011). Uncertainty sampling requires model warm-starting because it depends on model predictions, whereas diversity sampling can be a cold-start approach. A successful AL strategy should integrate both aspects, but its exact implementation is an open research question. For example, a naïve idea is to use a fixed combination of strategies to sample points. Nevertheless, Hsu and Lin (2015) experimentally show that this approach hampers accuracy. BADGE optimizes for both uncertainty and diversity by using confidence scores and clustering. This strategy beats uncertainty-based algorithms (Wang and Shang, 2014), sampling through bandit learning (Hsu and Lin, 2015), and CORESET (Sener and Savarese, 2018), a diversity-based method for convolutional neural networks.

3.1 BADGE

The goal of BADGE is to sample a diverse and uncertain batch of points for training neural networks. The algorithm transforms data into representations that encode model confidence and then clusters these transformed points. First, an unlabeled point x passes through the trained model to obtain its predicted label $\hat{y}$. Next, a **gradient embedding** g_x is computed for x such that it embodies the gradient of the cross-entropy loss on $(f(x;\theta), \hat{y})$ with respect to the parameters of the model's last layer. The gradient embedding is

$$(g_x)_i = (f(x;\theta)_i - \mathbb{1}(\hat{y} = i))h(x;W). \quad (1)$$

The i-th block of g_x is the hidden representation $h(x;W)$ scaled by the difference between model confidence score $f(x;\theta)_i$ and an indicator function $\mathbb{1}$ that indicates whether the predictive label $\hat{y}$ is label i. Finally, BADGE chooses a batch to sample by applying k-MEANS++ (Arthur and Vassilvitskii, 2006) on the gradient embeddings. These embeddings consist of model confidence scores and hidden representations, so they encode information about both uncertainty and the data distribution. By applying k-MEANS++ on the gradient embeddings, the chosen examples differ in feature representation and predictive uncertainty.

3.2 Limitations

BADGE combines uncertainty and diversity sampling to profit from advantages of both methods but also brings the downsides of both: reliance on warm-starting and computational inefficiency.

3.2.1 Model Uncertainty and Inference

Dodge et al. (2020) observe that training is highly unstable when fine-tuning pre-trained language models on small datasets. Accuracy significantly varies across different random initializations. The model has not fine-tuned on enough examples, so model confidence is an unreliable measure for uncertainty. While BADGE improves over uncertainty-based methods, it still relies on confidence scores $f(x;\theta)_i$ when computing the gradient embeddings (Equation 1). Also, it uses labels inferred by the model to compensate for lack of supervision in AL, but this inference is inaccurate for ill-trained models. Thus, warm-start methods may suffer from problems with model uncertainty or inference.

3.2.2 Algorithmic Efficiency

Many diversity-based methods involve distance comparison between embedding representations, but this computation can be expensive, especially in high-dimensional space. For instance, CORESET is a farthest-first traversal in the embedding space where it chooses the farthest point from the set of points already chosen on each iteration (Sener and Savarese, 2018). The embeddings may appropriately represent the data, but issues, like the "curse of dimensionality" (Beyer et al., 1999) and the "hubness problem" (Tomasev et al., 2013), persist. As the dimensionality increase, the distance between any two points converges to the same value. Moreover, the gradient embeddings in BADGE have dimensionality of Cd for a C-way classification task with data dimensionality of d (Equation 1). These issues make distance comparison between gradient embeddings less meaningful and raises costs to compute those distances.

4 A Self-supervised Active Learner

Cold-start AL is challenging because of the shortage in labeled data. Prior work, like BADGE, often depend on model uncertainty or inference, but these measures can be unreliable if the model has not trained on enough data (Section 3.2.1). To overcome the lack of supervision, what if we apply self-supervision to AL? For NLP, the language modeling task is self-supervised because the label

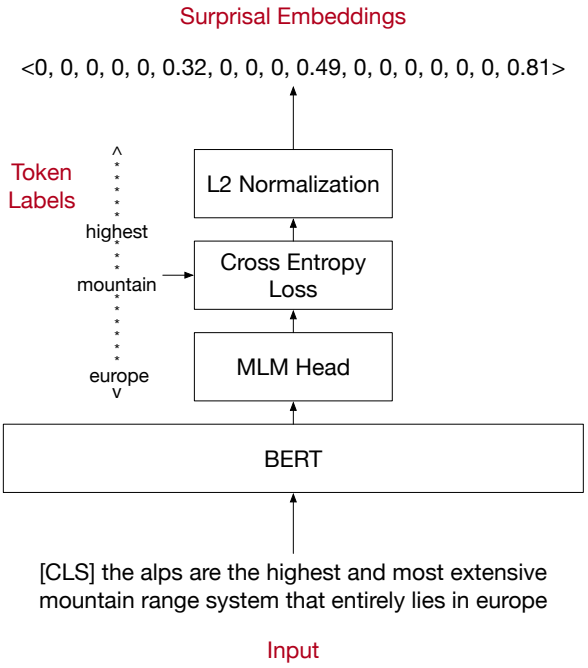


Figure 1: To form surprisal embedding s_x for sentence x, we pass in unmasked x through the BERT MLM head and compute cross-entropy loss for a random 15% subsample of tokens against the target labels. The unsampled tokens have entries of zero in s_x. ALPS clusters these surprisal embeddings to sample sentences for AL.

for each token is the token itself. If the task has immensely improved transfer learning, then it may reduce generalization error in AL too.

For our approach, we adopt the uncertainty-diversity BADGE framework for clustering embeddings that encode information about uncertainty. However, rather than relying on the classification loss gradient, we use the MLM loss to bootstrap uncertainty estimates. Thus, we combine uncertainty and diversity sampling for cold-start AL.

4.1 Masked Language Modeling

To pre-train BERT with MLM, input tokens are randomly masked, and the model needs to predict the token labels of the masked tokens. BERT is bidirectional, so it uses context from the left and right of the masked token to make predictions. BERT also uses next sentence prediction for pre-training, but this task shows minimal effect for fine-tuning (Liu et al., 2019). So, we focus on applying MLM to AL. The MLM head can capture syntactic phenomena (Goldberg, 2019) and performs well on psycholinguistic tests (Ettinger, 2020).

Algorithm 2 Single iteration of ALPS

Require: Pre-trained encoder $h(x; W_0)$, unlabeled data pool $\mathcal{U}$, number of queries k

for sentences $x \in \mathcal{U}$ **do**
  Compute s_x with MLM head of $h(x; W_0)$
$\mathcal{M} = \{s_x \mid x \in \mathcal{U}\}$
$\mathcal{C} \leftarrow k$-MEANS cluster centers of $\mathcal{M}$
$\mathcal{Q} = \{\arg\min_{x \in \mathcal{U}} \|c - s_x\| \,|c \in \mathcal{C}\}$
return $\mathcal{Q}$

4.2 ALPS

Surprisal Embeddings Inspired by how BADGE forms gradient embeddings from the classification loss, we create **surprisal embeddings** from language modeling. For sentence x, we compute surprisal embedding s_x by evaluating x with the MLM objective. To evaluate MLM loss, BERT randomly masks 15% of the tokens in x and computes cross-entropy loss for the masked tokens against their true token labels. When computing surprisal embeddings, we make one crucial change: *none of the tokens are masked when the input is passed into* BERT. However, we still randomly choose 15% of the tokens in the input to evaluate with cross-entropy against their target token labels. The unchosen tokens are assigned a loss of zero as they are not evaluated (Figure 1).

These decisions for not masking input (Appendix A.1) and evaluating only 15% of tokens (Appendix A.2) are made because of experiments on the validation set. Proposition 1 provides insight on the information encoded in surprisal embeddings. Finally, the surprisal embedding is l_2-normalized as normalization improves clustering (Aytekin et al., 2018). If the input sentences have a fixed length of l, then the surprisal embeddings have dimensionality of l. The length l is usually less than the hidden size of BERT embeddings.

Proposition 1. *For an unnormalized surprisal embedding s_x, each nonzero entry $(s_x)_i$ estimates $I(w_i)$, the surprisal of its corresponding token within the context of sentence x.*

Proof. Extending notation from Section 2, assume that m is the MLM head, with parameters $\phi = (W, Z)$, which maps input x to a $l \times |\mathcal{V}|$ matrix $m(x; \phi)$. The ith row $m(x; \phi)_i$ contains prediction scores for w_i, the ith token in x. Suppose that w_i is the jth token in vocabulary $\mathcal{V}$. Then, $m(x; \phi)_{i,j}$ is the likelihood of predicting w_i correctly.

Now, assume that context is the entire input x and define the language model probability p_m as,

$$p_m(w_i \mid x) = m(x; \phi)_{i,j}. \quad (2)$$

Salazar et al. (2020) have a similar definition as Equation 2 but instead have defined it in terms of the masked input. We argue that their definition can be extended to the unmasked input x. During BERT pre-training, the MLM objective is evaluated on the `[MASK]` token for 80% of the time, random token for 10% of the time, and the original token for 10% of the time. This helps maintain consistency across pre-training and fine-tuning because `[MASK]` never appears in fine-tuning (Devlin et al., 2019). Thus, we assume that m estimates occurrence of tokens within a maskless context as well.

Next, the information-theoretic surprisal (Shannon, 1948) is defined as $I(w) = -\log p(w \mid c)$, the negative log likelihood of word w given context c. If w_i is sampled and evaluated, then the ith entry of the unnormalized surprisal embedding is,

$$\begin{aligned}(s_x)_i &= -\log m(x; \phi)_{i,j} = -\log p_m(w_i \mid x) \\ &= I(w_i).\end{aligned}$$

□

Proposition 1 shows that the surprisal embeddings comprise of estimates for token-context surprisal. Intuitively, these values can help with AL because they highlight the information missing from the pre-trained model. For instance, consider the sentences: "this is my favorite television show" and "they feel ambivalent about catholic psychedelic synth folk music". Tokens from the latter have higher surprisal than those from the former. If this is a sentiment classification task, the second sentence is more confusing for the classifier to learn. The surprisal embeddings indicate sentences challenging for the pre-trained model to understand and difficult for the fine-tuned model to label.

The most surprising sentences contain many rare tokens. If we only train our model on the most surprising sentences, then it may not generalize well across different examples. Plus, we may sample several atypical sentences that are similar to each other, which is often an issue for uncertainty-based methods (Kirsch et al., 2019). Therefore, we incorporate clustering in ALPS to maintain diversity.

k-MEANS Clustering After computing surprisal embeddings for each sentence in the unlabeled pool, we use k-MEANS to cluster the surprisal embeddings. Then, for each cluster center, we select the sentence that has the nearest surprisal embedding to it. The final set of sentences are the queries to be labeled by an oracle (Algorithm 2). Although BADGE uses k-MEANS++ to cluster, experiments show that k-MEANS works better for surprisal embeddings (Appendix A.3).

5 Active Sentence Classification

We evaluate ALPS on sentence classification for three different domains: sentiment reviews, news articles, and medical abstracts (Table 1). To simulate AL, we sample a batch of 100 sentences from the training dataset, query labels for this batch, and then move the batch from the unlabeled pool to the labeled dataset (Algorithm 1). The initial encoder $h(x; \theta_0)$, is an already pre-trained, BERT-based model (Section 5.2). In a given iteration, we fine-tune the base classifier $f(x; \theta_0)$ on the labeled dataset and evaluate the fine-tuned model with classification micro-F_1 score on the test set. We do not fine-tune the model $f(x; \theta_{t-1})$ from the previous iteration to avoid issues with warm-starting (Ash and Adams, 2019). We repeat for ten iterations, collecting a total of 1,000 sentences.

5.1 Baselines

We compare ALPS against warm-start methods (Entropy, BADGE, FT-BERT-KM) and cold-start methods (Random, BERT-KM). For FT-BERT-KM, we use BERT-KM to sample data in the first iteration. For other warm-start methods, data is randomly sampled in the first iteration.

Entropy Sample k sentences with highest predictive entropy measured by $\sum_{i=1}^{C} (f(x; \theta)_i) \ln(f(x; \theta)_i)^{-1}$ (Lewis and Gale, 1994; Wang and Shang, 2014).

BADGE Sample k sentences based on diversity in loss gradient (Section 3.1).

BERT-KM Cluster pre-trained, l_2-normalized BERT embeddings with k-MEANS and sample the nearest neighbors of the k cluster centers. The algorithm is the same as ALPS except that BERT embeddings are used.

FT-BERT-KM This is the same algorithm as BERT-KM except the BERT embeddings $h(x; W_{t-1})$ from the previously fine-tuned model are used.

Dataset	Domain	Train	Dev	Test	# Labels
AG NEWS	News articles	110,000	10,000	7,600	4
IMDB	Sentiment reviews	17,500	7,500	25,000	2
PUBMED 20k RCT	Medical abstracts	180,040	30,212	30,135	5
SST-2	Sentiment reviews	60,615	6,736	873	2

Table 1: Sentence classification datasets used in experiments.

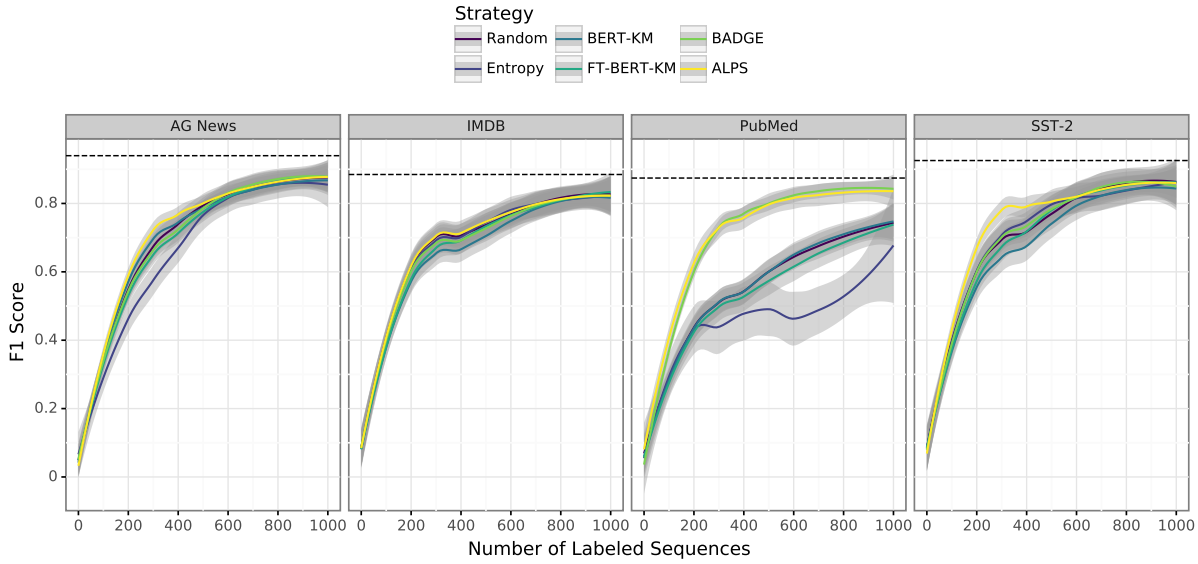


Figure 2: Test accuracy of simulated AL over ten iterations with 100 sentences queried per iteration. The dashed line is the test accuracy when the model is fine-tuned on the entire dataset. Overall, models trained with data sampled from ALPS have the highest test accuracy, especially for the earlier iterations.

5.2 Setup

For each sampling algorithm and dataset, we run the AL simulation five times with different random seeds. We set the maximum sequence length to 128. We fine-tune on a batch size of thirty-two for three epochs. We use AdamW (Loshchilov and Hutter, 2019) with learning rate of 2e-5, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a linear decay of learning rate.

For IMDB (Maas et al., 2011), SST-2 (Socher et al., 2013), and AG NEWS (Zhang et al., 2015), the data encoder is the uncased BERT-Base model with 110M parameters.[2] For PUBMED (Dernoncourt and Lee, 2017), the data encoder is SCIBERT, a BERT model pre-trained on scientific texts (Beltagy et al., 2019). All experiments are run on GeForce GTX 1080 GPU and 2.6 GHz AMD Opteron 4180 CPU processor; runtimes in Table 2.

[2] https://huggingface.co/transformers/

5.3 Results

The model fine-tuned with data sampled by ALPS has higher test accuracy than the baselines (Figure 2). For AG NEWS, IMDB, and SST-2, this is true in earlier iterations. We often see the most gains in the beginning for crowdsourcing (Felt et al., 2015). Interestingly, clustering the fine-tuned BERT embeddings is not always better than clustering the pre-trained BERT embeddings for AL. The fine-tuned BERT embeddings may require training on more data for more informative representations.

For PUBMED, test accuracy greatly varies between the strategies. The dataset belongs to a specialized domain and is class-imbalanced, so naïve methods show poor accuracy. Entropy sampling has the lowest accuracy because the classification entropy is uninformative in early iterations. The models fine-tuned on data sampled by ALPS and BADGE have about the same accuracy. Both methods strive to optimize for uncertainty and diversity, which alleviates problems with class imbalance.

	AG NEWS	PUBMED
Random	<1	<1
Entropy	7	10
ALPS	14	24
BADGE	23	70
BERT-KM	28	58
FT-BERT-KM	33	79

Table 2: Average runtime (minutes) per sampling iteration during AL simulation for large datasets. BADGE, FT-BERT-KM, and BERT-KM take much longer to run.

Our experiments cover the first ten iterations because we focus on the cold-start setting. As sampling iterations increase, test accuracy across the different methods converges. Both ALPS and BADGE already approach the model trained on the full training dataset across all tasks (Figure 2). Once the cold-start issue subsides, uncertainty-based methods can be employed to further query the most confusing examples for the model to learn.

6 Analyzing ALPS

Sampling Efficiency Given that the gradient embeddings are computed, BADGE has a time complexity of $\mathcal{O}(Cknd)$ for a C-way classification task, k queries, n points in the unlabeled pool, and d-dimensional BERT embeddings. Given that the surprisal embeddings are computed, ALPS has a time complexity of $\mathcal{O}(tknl)$ where t is the fixed number of iterations for k-MEANS and l is the maximum sequence length. In our experiments, $k = 100$, $d = 768$, $t = 10$, and $l = 128$. In practice, t will not change much, but n and C could be much higher. For large dataset PUBMED, the average runtime per iteration is 24 minutes for ALPS and 70 minutes for BADGE (Table 2). So, ALPS can match BADGE's accuracy more quickly.

Diversity and Uncertainty We estimate diversity and uncertainty for data sampled across different strategies. For diversity, we look at the overlap between tokens in the sampled sentences and tokens from the rest of the data pool. A diverse batch of sentences should share many of the same tokens with the data pool. In other words, the sampled sentences can represent the data pool because of the substantial overlap between their tokens. In our simulations, the entire data pool is the training dataset (Section 5). So, we compute the Jaccard similarity between $\mathcal{V}_D$, set of tokens from the sampled sentences $\mathcal{D}$, and $\mathcal{V}_{D'}$, set of tokens from the unsampled sentences $\mathcal{U} \setminus \mathcal{D}$,

$$G_d(\mathcal{D}) = J(\mathcal{V}_D, \mathcal{V}_{D'}) = \frac{|\mathcal{V}_D \cap \mathcal{V}_{D'}|}{|\mathcal{V}_D \cup \mathcal{V}_{D'}|}. \quad (3)$$

If G_d is high, this indicates high diversity because the sampled and unsampled sentences have many tokens in common. If G_d is low, this indicates poor diversity and representation.

To measure uncertainty, we use $f(x, \theta_*)$, the classifier trained on the full training dataset. In our experiments, classifier $f(x, \theta_*)$ has high accuracy (Figure 2) and inference is stable after training on many examples. Thus, we can use the logits from the classifier to understand its uncertainty toward a particular sentence. First, we compute predictive entropy of sentence x when evaluated by model $f(x, \theta_*)$. Then, we take the average of predictive

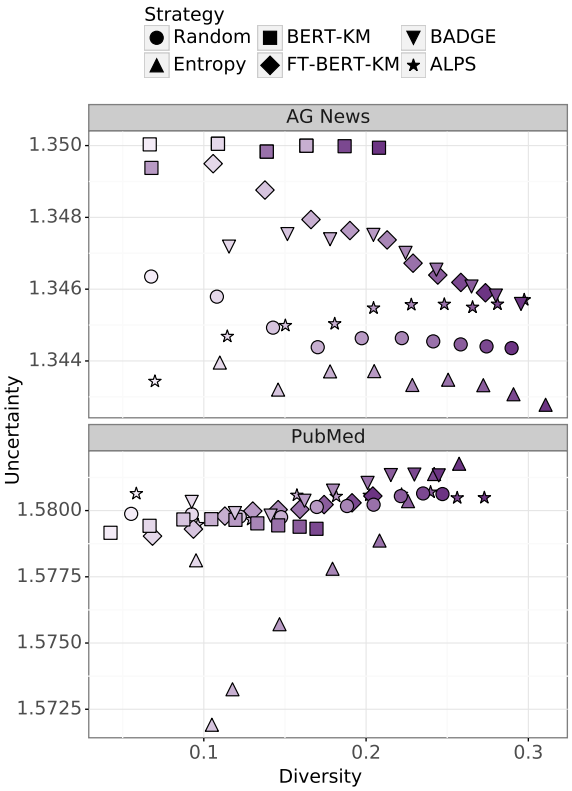
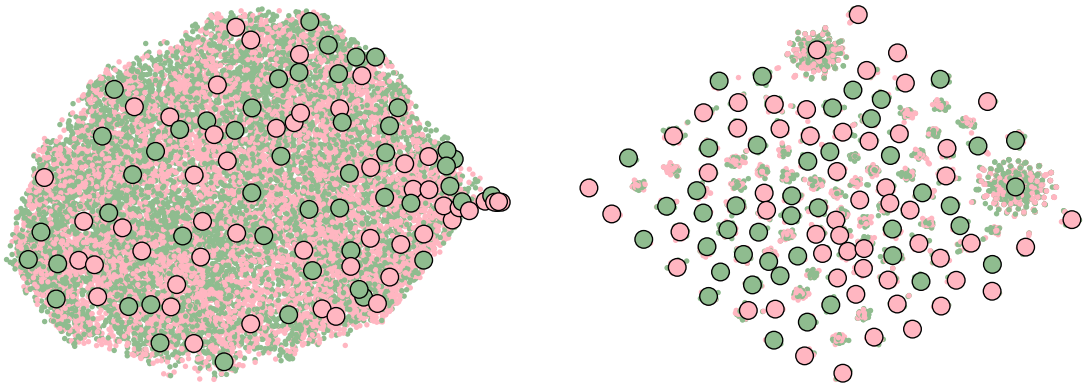


Figure 3: Plot of diversity against uncertainty estimates from AL simulations for AG NEWS and PUBMED. Each point represents a sampled batch of sentences from the AL experiments. The shape indicates the strategy used to sample the sentences. The color indicates the sample iteration. The lightest color corresponds to the first iteration and the darkest color represents the tenth iteration. While uncertainty estimates are similar across different batches, ALPS shows a consistent increase in diversity without drops in uncertainty.



(a) BERT embeddings with k-MEANS centers

(b) Surprisal embeddings with k-MEANS centers

Figure 4: T-SNE plots of BERT embeddings and surprisal embeddings for each sequence in the IMDB training dataset. The enlarged points are the centers determined by k-MEANS (left) and k-MEANS++ (right). The points are colored according to their classification labels. In both sets of embeddings, we cannot clearly separate the points from their labels, but the distinction between clusters in surprisal embeddings seems more obvious.

entropy over all sentences in a sampled batch $\mathcal{D}$. We use the average predictive entropy to esimate uncertainty of the sampled sentences,

$$G_u(\mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} \sum_{i=1}^{C} (f(x;\theta_*)_i) \ln(f(x;\theta_*)_i)^{-1}. \tag{4}$$

We compute G_d and G_u for batches sampled in the AL experiments of AG NEWS and PUBMED. Diversity is plotted against uncertainty for batches sampled across different iterations and AL strategies (Figure 3). For AG NEWS, G_d and G_u are relatively low for ALPS in the first iteration. As iterations increase, samples from ALPS increase in diversity and decrease minimally in uncertainty. Samples from other methods have a larger drop in uncertainty as iterations increase. For PUBMED, ALPS again increases in sample diversity without drops in uncertainty. In the last iteration, ALPS has the highest diversity among all the algorithms.

Surprisal Clusters Prior work use k-MEANS to cluster feature representations as a cold-start AL approach (Zhu et al., 2008; Bodó et al., 2011). Rather than clustering BERT embeddings, ALPS clusters surprisal embeddings. We compare the clusters between surprisal embeddings and BERT embeddings to understand the structure of the surprisal clusters. First, we use t-SNE (Maaten and Hinton, 2008) to plot the embeddings for each sentence in the IMDB training set (Figure 4). The labels are not well-separated for both embedding sets, but the surprisal embeddings seem easier to cluster. To quantitively measure cluster quality, we use the Silhouette Coefficient for which larger values indicate desirable clustering (Rousseeuw, 1987). The surprisal clusters have a coefficient of 0.38, whereas the BERT clusters have a coefficient of only 0.04.

These results, along with the classification experiments, show that naïvely clustering BERT embeddings is not suited for AL. Possibly, more complicated clustering algorithms can capture the intrinsic structure of the BERT embeddings. However, this would increase the algorithmic complexity and runtime. Alternatively, one can map the feature representations to a space where simple clustering algorithms work well. During this transformation, important information for AL must be preserved and extracted. Our approach uses the MLM head, which has already been trained on extensive corpora, to map the BERT embeddings into the surprisal embedding space. As a result, simple k-MEANS can efficiently choose representative sentences.

Single-iteration Sampling In Section 5, we sample data iteratively (Algorithm 1) to fairly compare the different AL algorithms. However, ALPS does not require updating the classifier because it only depends on the pre-trained encoder. Rather than sampling data in small batches and re-training the model, ALPS can sample a batch of k sentences in one iteration (Algorithm 2). Between using ALPS iteratively and deploying the algorithm for a single iteration, the difference is insignificant (Table 3). Plus, sampling 1,000 sentences only takes about 97 minutes for PUBMED and 7 minutes for IMDB.

Dataset	k	Iterative	Single
IMDB	200	0.63 ± 0.04	0.61 ± 0.03
	500	0.74 ± 0.05	0.76 ± 0.04
	1000	0.82 ± 0.01	0.82 ± 0.01
PUBMED	200	0.63 ± 0.03	0.64 ± 0.03
	500	0.80 ± 0.02	0.82 ± 0.01
	1000	0.84 ± 0.00	0.84 ± 0.00

Table 3: Test accuracy on IMDB and PubMed between different uses of ALPS for various k, the number of sentences to query. We compare using ALPS iteratively (Iterative) as done in Section 5 with using ALPS to query all k sentences in one iteration (Single). The test accuracy does not change much, showing that ALPS is flexible to apply in different settings.

With this flexibility in sampling, ALPS can accommodate different budget constraints. For example, re-training the classifier may be costly, so users want a sampling algorithm that can query k sentences all at once. In other cases, annotators are not always available, so the number of obtainable annotations is unpredictable. Then, users would prefer an AL strategy that can query a variable number of sentences for any iteration. These cases illustrate practical needs for a cold-start algorithm like ALPS.

7 Related Work

Active learning has shown success in tasks, such as named entity recognition (Shen et al., 2004), word sense disambiguation (Zhu and Hovy, 2007), and sentiment analysis (Li et al., 2012). Wang and Shang (2014) are the first to adapt prior AL work to deep learning. However, popular heuristics (Settles, 2009) for querying individual points do not work as well in a batch setting. Since then, more research has been conducted on batch AL for deep learning. Zhang et al. (2017) propose the first work on AL for neural text classification. They assume that the classifier is a convolutional neural network and use expected gradient length (Settles et al., 2008) to choose sentences that contain words with the most label-discriminative embeddings. Besides text classification, AL has been applied to neural models for semantic parsing (Duong et al., 2018), named entity recognition (Shen et al., 2018), and machine translation (Liu et al., 2018).

ALPS makes use of BERT, a model that excels at transfer learning. Other works also combine AL and transfer learning to select training data that reduce generalization error. Rai et al. (2010) measures domain divergence from the source domain to select the most informative texts in the target domain. Wang et al. (2014) use AL to query points for a target task through matching conditional distributions. Additionally, combining word-level and document-level annotations can improve knowledge transfer (Settles, 2011; Yuan et al., 2020).

In addition to uncertainty and diversity sampling, other areas of deep AL focus on Bayesian approaches (Siddhant and Lipton, 2018; Kirsch et al., 2019) and reinforcement learning (Fang et al., 2017). An interesting research direction can integrate one of these approaches with ALPS.

8 Conclusion

Transformers are powerful models that have revolutionized NLP. Nevertheless, like other deep models, their accuracy and stability require fine-tuning on large amounts of data. AL should level the playing field by directing limited annotations most effectively so that labels complement, rather than duplicate, unsupervised data. Luckily, transformers have generalized knowledge about language that can help acquire data for fine-tuning. Like BADGE, we project data into an embedding space and then select the most representative points. Our method is unique because it only relies on self-supervision to conduct sampling. Using the pre-trained loss guides the AL process to sample diverse and uncertain examples in the cold-start setting. Future work may focus on finding representations that encode the most important information for AL.

Acknowledgments

We thank Kuen-Han Tsai, Chien-Min Yu, Si-An Chen, Pedro Rodriguez, Eleftheria Briakou, and the anonymous reviewers for their feedback. Michelle Yuan is supported by JHU Human Language Technology Center of Excellence (HLTCOE). Jordan Boyd-Graber is supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the BETTER Program contract #2019-19051600005. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

David Arthur and Sergei Vassilvitskii. 2006. k-means++: The advantages of careful seeding. Technical report, Stanford.

Jordan T. Ash and Ryan P. Adams. 2019. On warm-starting neural network training. *arXiv preprint arXiv:1910.08475*.

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*.

Caglar Aytekin, Xingyang Ni, Francesco Cricri, and Emre Aksu. 2018. Clustering and unsupervised anomaly detection with L2 normalized deep auto-encoder representations. In *International Joint Conference on Neural Networks*.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of Empirical Methods in Natural Language Processing*.

Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. 1999. When is "nearest neighbor" meaningful? In *International Conference on Database Theory*.

Zalán Bodó, Zsolt Minier, and Lehel Csató. 2011. Active learning with clustering. In *Active Learning and Experimental Design Workshop in Conjunction with AISTATS 2010*.

Sanjoy Dasgupta. 2011. Two faces of active learning. *Theoretical computer science*, 412(19):1767–1781.

Joe Davison, Joshua Feldman, and Alexander M. Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of Empirical Methods in Natural Language Processing*.

Franck Dernoncourt and Ji Young Lee. 2017. PubMed 200k RCT: a dataset for sequential sentence classification in medical abstracts. *International Joint Conference on Natural Language Processing*, 2:308–313.

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

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*.

Long Duong, Hadi Afshar, Dominique Estival, Glen Pink, Philip R Cohen, and Mark Johnson. 2018. Active learning for deep semantic parsing. In *Proceedings of the Association for Computational Linguistics*.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

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

Paul Felt, Eric Ringger, Kevin Seppi, Kevin Black, and Robbie Haertel. 2015. Early gains matter: A case for preferring generative over discriminativecrowdsourcing models. In *Proceedings of the Association for Computational Linguistics*.

Yoav Goldberg. 2019. Assessing BERT's syntactic abilities. *arXiv preprint arXiv:1901.05287*.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. *Journal of Machine Learning Research*, 70:1321–1330.

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 Association for Computational Linguistics*.

Wei-Ning Hsu and Hsuan-Tien Lin. 2015. Active learning by learning. In *Association for the Advancement of Artificial Intelligence*.

Rong Hu, Brian Mac Namee, and Sarah Jane Delany. 2010. Off to a good start: Using clustering to select the initial training set in active learning. In *Florida Artificial Intelligence Research Society Conference*.

Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. 2019. BatchBALD: Efficient and diverse batch acquisition for deep Bayesian active learning. In *Proceedings of Advances in Neural Information Processing Systems*.

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

Shoushan Li, Shengfeng Ju, Guodong Zhou, and Xiaojun Li. 2012. Active learning for imbalanced sentiment classification. In *Proceedings of Empirical Methods in Natural Language Processing*.

Ming Liu, Wray Buntine, and Gholamreza Haffari. 2018. Learning to actively learn neural machine translation. In *Conference on Computational Natural Language Learning*.

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. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of the International Conference on Learning Representations*.

David Lowell and Zachary C. Lipton. 2019. Practical obstacles to deploying active learning. In *Proceedings of Empirical Methods in Natural Language Processing*.

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 Association for Computational Linguistics*.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of Empirical Methods in Natural Language Processing*.

Piyush Rai, Avishek Saha, Hal Daumé III, and Suresh Venkatasubramanian. 2010. Domain adaptation meets active learning. In *Conference of the North American Chapter of the Association for Computational Linguistics*.

Peter J. Rousseeuw. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 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. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Burr Settles. 2011. Closing the loop: Fast, interactive semi-supervised annotation with queries on features and instances. In *Proceedings of Empirical Methods in Natural Language Processing*.

Burr Settles, Mark Craven, and Soumya Ray. 2008. Multiple-instance active learning. In *Proceedings of Advances in Neural Information Processing Systems*.

Claude Elwood Shannon. 1948. A mathematical theory of communication. *Bell system technical journal*, 27.

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 Association for Computational Linguistics*.

Yanyao Shen, Hyokun Yun, Zachary C. Lipton, Yakov Kronrod, and Animashree Anandkumar. 2018. Deep active learning for named entity recognition. In *Proceedings of the International Conference on Learning Representations*.

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 Empirical Methods in Natural Language Processing*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of Empirical Methods in Natural Language Processing*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the Association for Computational Linguistics*.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, et al. 2019. What do you learn from context? Probing for sentence structure in contextualized word representations. In *Proceedings of the International Conference on Learning Representations*.

Nenad Tomasev, Milos Radovanovic, Dunja Mladenic, and Mirjana Ivanovic. 2013. The role of hubness in clustering high-dimensional data. *IEEE Transactions on Knowledge and Data Engineering*, 26(3):739–751.

Ellen Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R. Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. 2020. TREC-COVID: Constructing a pandemic information retrieval test collection. *arXiv preprint arXiv:2005.04474*.

Dan Wang and Yi Shang. 2014. A new active labeling method for deep learning. In *International Joint Conference on Neural Networks*.

Xuezhi Wang, Tzu-Kuo Huang, and Jeff Schneider. 2014. Active transfer learning under model shift. In *Proceedings of the International Conference on Learning Representations*.

Zhao Xu, Kai Yu, Volker Tresp, Xiaowei Xu, and Jizhi Wang. 2003. Representative sampling for text classification using support vector machines. In *Proceedings of the European Conference on Information Retrieval*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. In *Proceedings of Advances in Neural Information Processing Systems*.

Michelle Yuan, Mozhi Zhang, Benjamin Van Durme, Leah Findlater, and Jordan Boyd-Graber. 2020. Interactive refinement of cross-lingual word embeddings. In *Proceedings of Empirical Methods in Natural Language Processing*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of Advances in Neural Information Processing Systems*.

Ye Zhang, Matthew Lease, and Byron C. Wallace. 2017. Active discriminative text representation learning. In *Association for the Advancement of Artificial Intelligence*.

Jingbo Zhu and Eduard Hovy. 2007. Active learning for word sense disambiguation with methods for addressing the class imbalance problem. In *Proceedings of Empirical Methods in Natural Language Processing*.

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 International Conference on Computational Linguistics*.

A Appendices

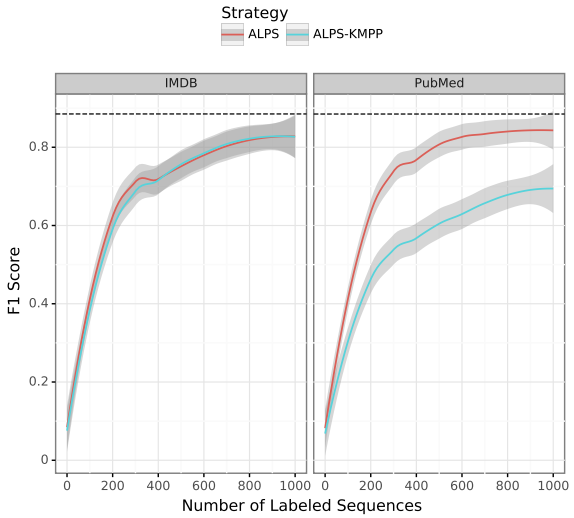
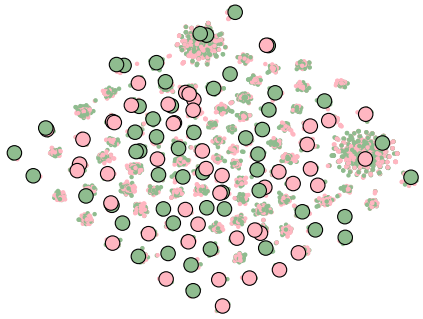
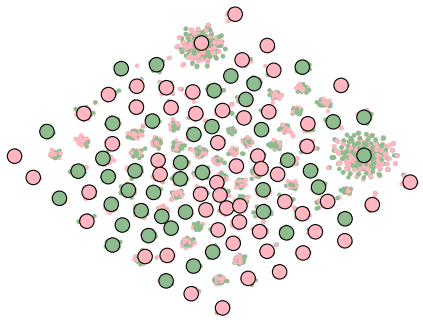


Figure 5: Comparing validation accuracy between using k-MEANS and k-MEANS++ to select centroids in the surprisal embeddings. Using k-MEANS reaches higher accuracy.



(a) Surprisal embeddings with k-MEANS++ centers



(b) Surprisal embeddings with k-MEANS centers

Figure 6: T-SNE plots of surprisal embeddings for IMDB training data. The centers are either picked by k-MEANS++ (right) or k-MEANS (left). There is less overlap between the centers with k-MEANS compared to k-MEANS++. So, using k-MEANS is better for exploiting diversity in the surprisal embedding space.

A.1 Token Masking

In our preliminary experiments on the validation set, we notice improvement in accuracy after passing in the original input with no masks (Table 4). The purpose of the `[MASK]` token during pretraining is to train the token embeddings to learn context so that it can predict the token labels. Since we are not training the token embeddings to learn context, masking the tokens does not help much for AL. We use AL for fine-tuning, so the input should be in the same format for AL and fine-tuning. Otherwise, there is a mismatch between the two stages.

A.2 Token Sampling for Evaluation

When BERT evaluates MLM loss, it only focuses on the masked tokens, which are from a 15% random subsample of tokens in the sentence. We experiment with varying this subsample percentage on the validation set (Table 4). We try sampling 10%, 15%, 20%, and 100%. Overall, we notice that mean accuracy are roughly the same, but variance in accuracy across different runs is slightly higher for percentages other than 15%.

After the second AL iteration, we notice that accuracy mean and variance between the different token sampling percentages converge. So, the token sampling percentage makes more of a difference in early stages of AL. Devlin et al. (2019) show that the difference in accuracy between various mask strategies is minimal for fine-tuning BERT. We believe this can also be applied to what we have observed for ALPS.

A.3 k-MEANS vs. k-MEANS++

The state-of-the-art baseline BADGE applies k-MEANS++ on gradient embeddings to select points to query. Initially, we also use k-MEANS++ on the surprisal embeddings but validation accuracy is only slightly higher than random sampling. Since k-MEANS++ is originally an algorithm for robust initialization of k-MEANS, we instead apply k-MEANS on the surprisal embeddings. As a result, we see more significant increase in accuracy over baselines, especially for PubMed (Figure 5). Additionally, the t-SNE plots show that k-MEANS selects centers that are further apart compared to the ones chosen by k-MEANS++ (Figure 6). This shows that k-MEANS can help sample a more diverse batch of data.

	IMDB		SST-2	
	$k = 100$	$k = 200$	$k = 100$	$k = 200$
ALPS	0.60 ± 0.03	$\mathbf{0.69 \pm 0.04}$	$\mathbf{0.57 \pm 0.06}$	$\mathbf{0.64 \pm 0.04}$
ALPS-tokens-0.1	$\mathbf{0.61 \pm 0.05}$	0.63 ± 0.11	0.56 ± 0.07	0.63 ± 0.04
ALPS-tokens-0.2	0.55 ± 0.07	0.65 ± 0.05	$\mathbf{0.57 \pm 0.05}$	0.63 ± 0.05
ALPS-tokens-1.0	0.59 ± 0.05	0.65 ± 0.07	0.56 ± 0.05	0.62 ± 0.05
ALPS-masked	0.59 ± 0.03	0.63 ± 0.09	0.56 ± 0.03	0.60 ± 0.02

Table 4: Comparison of validation accuracy between the variants of ALPS to sample data for IMDB and SST-2 in the first two iterations. ALPS-tokens-p varies the percentage p of tokens evaluated with MLM loss when computing surprisal embeddings. ALPS-masked passes in the input with masks as originally done in pre-training. Overall, we observe that ALPS has higher mean and smaller variance in accuracy.

	AG NEWS	PUBMED
ALPS	Jason Thomas matches a career-high with 26 points and American wins its fifth straight by beating visiting Ohio, 64-55, Saturday at Bender Arena (Sports)	The results showed that physical activity and exercise capacity in the intervention group was significantly higher than the control group after the intervention . (results)
	Sainsbury says it will take a 550 million pound hit to profits this year as it invests to boost sales and reverse falling market share (Business)	Flumazenil was administered after the completion of endoscopy under sedation to reduce recovery time and increase patient safety . (objective)
Random	Bernhard Langer and Hal Sutton stressed the importance of playing this year's 135th Ryder Cup . . . (Sports)	The study population consisted of 20 interns and medical students (methods)
	BLOOMFIELD TOWNSHIP, Mich. – When yesterday's Ryder Cup pairings were announced, Bernhard Langer knew his team had been given an opportunity. (Sports)	The subject , health care provider , and research staff were blinded to the treatment . (methods)

Table 5: Sample sentences from AG News and PubMed while using ALPS and Random in the first iteration. For ALPS, highlighted tokens are the ones that have a nonzero entry in the surprisal embedding. Compared to random sampling, ALPS samples sentences with more diverse content.

A.4 Sample Sentences

Section 6 quantitatively analyzes diversity of ALPS. Here, we take a closer look at the kind of sentences that are sampled by ALPS. Table 5 compares sentences that are chosen by ALPS and random sampling in the first AL iteration. The tokens highlighted are the ones evaluated with surprisal loss. Random sampling can fall prey to data idiosyncracies. For example, AG News has sixty-two articles about the German golfer Bernhard Langer, and random sampling picks multiple articles about him on one of five runs. For PubMed, many sentences labeled as "methods" are simple sentences with a short, independent clause. While random sampling chooses many sentences of this form, ALPS seems to avoid this problem. Since the surprisal embedding encodes the fluctuation in information content across the sentence, ALPS is less likely to repeatedly choose sentences with similar patterns in surprisal. This may possibly diversify syntactic structure in a sampled batch.