Practical Obstacles to Deploying Active Learning

David Lowell
Northeastern University
lowell.d@husky.neu.edu

Zachary C. Lipton
Carnegie Mellon University
zlipton@cmu.edu

Byron C. Wallace
Northeastern University
b.wallace@northeastern.edu

Abstract

Active learning (AL) is a widely-used training strategy for maximizing predictive performance subject to a fixed annotation budget. In AL one iteratively selects training examples for annotation, often those for which the current model is most uncertain (by some measure). The hope is that active sampling leads to better performance than would be achieved under independent and identically distributed (i.i.d.) random samples. While AL has shown promise in retrospective evaluations, these studies often ignore practical obstacles to its use. In this paper we show that while AL may provide benefits when used with specific models and for particular domains, the benefits of current approaches do not generalize reliably across models and tasks. This is problematic because in practice one does not have the opportunity to explore and compare alternative AL strategies. Moreover, AL couples the training dataset with the model used to guide its acquisition. We find that subsequently training a *successor model* with an actively-acquired dataset does not consistently outperform training on i.i.d. sampled data. Our findings raise the question of whether the downsides inherent to AL are worth the modest and inconsistent performance gains it tends to afford.

1 Introduction

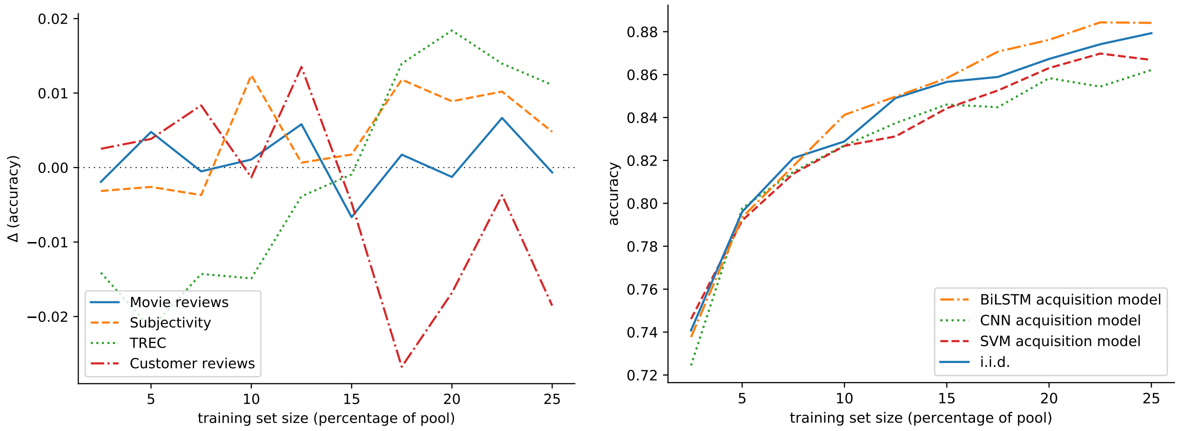
Although deep learning now achieves state-of-the-art results on a number of supervised learning tasks (Johnson and Zhang, 2016; Ghaddar and Langlais, 2018), realizing these gains requires large annotated datasets (Shen et al., 2018). This data dependence is problematic because labels are expensive. Several lines of research seek to reduce the amount of supervision required to achieve acceptable predictive performance, including semi-supervised (Chapelle et al., 2009), transfer (Pan and Yang, 2010), and *active learning* (AL) (Cohn et al., 1996; Settles, 2012).

In AL, rather than training on a set of labeled data sampled at i.i.d. random from some larger population, the learner engages the annotator in a cycle of learning, iteratively selecting training data for annotation and updating its model. *Pool-based* AL (the variant we consider) proceeds in rounds. In each, the learner applies a heuristic to score unlabeled instances, selecting the highest scoring instances for annotation.[1] Intuitively, by selecting training data cleverly, an active learner might achieve greater predictive performance than it would by choosing examples at random.

The more informative samples come at the cost of violating the standard i.i.d. assumption upon which supervised machine learning typically relies. In other words, the training and test data no longer reflect the same underlying data distribution. Empirically, AL has been found to work well with a variety of tasks and models (Settles, 2012; Ramirez-Loaiza et al., 2017; Gal et al., 2017a; Zhang et al., 2017; Shen et al., 2018). However, academic investigations of AL typically omit key real-world considerations that might overestimate its utility. For example, once a dataset is actively acquired with one model, it is seldom investigated whether this training sample will confer benefits if used to train a second model (vs i.i.d. data). Given that datasets often outlive learning algorithms, this is an important practical consideration.

[1] This may be done either deterministically, by selecting the top-k instances, or stochastically, selecting instances with probabilities proportional to heuristic scores.

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing
and the 9th International Joint Conference on Natural Language Processing, pages 21–30,
Hong Kong, China, November 3–7, 2019.



(a) Performance of AL relative to i.i.d. across corpora.

(b) Transferring actively acquired training sets.

Figure 1: We highlight practical issues in the use of AL. (a) AL yields inconsistent gains, relative to a baseline of i.i.d. sampling, across corpora. (b) Training a BiLSTM with training sets actively acquired based on the uncertainty of other models tends to result in worse performance than training on i.i.d. samples.

In contrast to experimental (retrospective) studies, in a real-world setting, an AL practitioner is not afforded the opportunity to retrospectively analyze or alter their scoring function. One would instead need to expend significant resources to validate that a given scoring function performs as intended for a particular model and task. This would require i.i.d. sampled data to evaluate the comparative effectiveness of different AL strategies. However, collection of such additional data would defeat the purpose of AL, i.e., obviating the need for a large amount of supervision. To confidently use AL in practice, one must have a reasonable belief that a given AL scoring (or *acquisition*) function will produce the desired results *before they deploy it* (Attenberg and Provost, 2011).

Most AL research does not explicitly characterize the circumstances under which AL may be expected to perform well. Practitioners must therefore make the implicit assumption that a given active acquisition strategy is likely to perform well under *any* circumstances. Our empirical findings suggest that this assumption is not well founded and, in fact, common AL algorithms behave inconsistently across model types and datasets, often performing no better than random (i.i.d.) sampling (1a). Further, while there is typically *some* AL strategy which outperforms i.i.d. random samples for a given dataset, *which* heuristic varies.

Contributions. We highlight important but often overlooked issues in the use of AL in practice. We report an extensive set of experimental results on classification and sequence tagging tasks that suggest AL typically affords only marginal performance gains at the somewhat high cost of non-i.i.d. training samples, which do not consistently transfer well to subsequent models.

2 The (Potential) Trouble with AL

We illustrate inconsistent comparative performance using AL. Consider Figure 1a, in which we plot the relative gains (Δ) achieved by a BiLSTM model using a maximum-entropy active sampling strategy, as compared to the same model trained with randomly sampled data. Positive values on the y-axis correspond to cases in which AL achieves better performance than random sampling, 0 (dotted line) indicates no difference between the two, and negative values correspond to cases in which random sampling performs better than AL. Across the four datasets shown, results are decidedly mixed.

And yet realizing these equivocal gains using AL brings inherent drawbacks. For example, acquisition functions generally depend on the underlying model being trained (Settles, 2009, 2012), which we will refer to as the *acquisition model*. Consequently, the collected training data and the acquisition model are *coupled*. This coupling is problematic because manually labeled data tends to have a longer shelf life than models, largely because it is expensive to acquire. However, progress in machine learning is fast. Consequently, in many settings, an actively acquired dataset may remain in use (much) longer than the source model used to acquire it. In these cases, a few natural questions arise: How does a *successor* model S fare, when trained on data collected via an acquisition model A? How does this compare to training S on natively acquired data? How does it compare to training S on i.i.d. data?

For example, if we use uncertainty sampling under a support vector machine (SVM) to acquire a training set $\mathcal{D}$, and subsequently train a Convolutional Neural Network (CNN) using $\mathcal{D}$, will the CNN perform better than it would have if trained on a dataset acquired via i.i.d. random sampling? And how does it perform compared to using a training corpus actively acquired using the CNN?

Figure 1b shows results for a text classification example using the Subjectivity corpus (Pang and Lee, 2004). We consider three models: a Bidirectional Long Short-Term Memory Network (BiLSTM) (Hochreiter and Schmidhuber, 1997), a Convolutional Neural Network (CNN) (Kim, 2014; Zhang and Wallace, 2015), and a Support Vector Machine (SVM) (Joachims, 1998). Training the LSTM with a dataset actively acquired using either of the other models yields predictive performance that is *worse* than that achieved under i.i.d. sampling. Given that datasets tend to outlast models, these results raise questions regarding the benefits of using AL in practice.

We note that in prior work, Tomanek and Morik (2011) also explored the transferability of actively acquired datasets, although their work did not consider modern deep learning models or share our broader focus on practical issues in AL.

3 Experimental Questions and Setup

We seek to answer two questions empirically: (1) How reliably does AL yield gains over sampling i.i.d.? And, (2) What happens when we use a dataset actively acquired using one model to train a different (successor) model? To answer these questions, we consider two tasks for which AL has previously been shown to confer considerable benefits: text classification and sequence tagging (specifically NER).[2]

To build intuition, our experiments address both linear models and deep networks more representative of the current state-of-the-art for these tasks. We investigate the standard strategy of acquiring data and training using a single model, and also the case of acquiring data using one model and subsequently using it to train a second model. Our experiments consider all possible (acquisition, successor) pairs among the considered models, such that the standard AL scheme corresponds to the setting in which the acquisition and successor models are same. For each pair (A, S), we first simulate iterative active data acquisition with model A to label a training dataset $\mathcal{D}_A$. We then train the successor model S using $\mathcal{D}_A$.

In our evaluation, we compare the relative performance (accuracy or F1, as appropriate for the task) of the successor model trained with corpus $\mathcal{D}_A$ to the scores achieved by training on comparable amounts of native and i.i.d. sampled data. We simulate pool-based AL using labeled benchmark datasets by withholding document labels from the models. This induces a pool of unlabeled data $\mathcal{U}$. In AL, it is common to *warm-start* the acquisition model, training on some modest amount of i.i.d. labeled data $\mathcal{D}_w$ before using the model to score candidates in $\mathcal{U}$ (Settles, 2009) and commencing the AL process. We follow this convention throughout.

Once we have trained the acquisition model on the warm-start data, we begin the simulated AL loop, iteratively selecting instances for labeling and adding them to the dataset. We denote the dataset acquired by model A at iteration t by $\mathcal{D}_A^t$; $\mathcal{D}_A^0$ is initialized to $\mathcal{D}_w$ for all models (i.e., all values of A). At each iteration, the acquisition model is trained with $\mathcal{D}_A^t$. It then scores the remaining unlabeled documents in $\mathcal{U} \setminus \mathcal{D}_A^t$ according to a standard uncertainty AL heuristic. The top n candidates $\mathcal{C}_A^t$ are selected for (simulated) annotation. Their labels are revealed and they are added to the training set: $\mathcal{D}_A^{t+1} \leftarrow \mathcal{D}_A^t \cup \mathcal{C}_A^t$. At the experiment's conclusion (time step T), each acquisition model A will have selected a (typically distinct) subset of $\mathcal{U}$ for training.

Once we have acquired datasets from each acquisition model $\mathcal{D}_A$, we evaluate the performance of each possible successor model when trained on $\mathcal{D}_A$. Specifically, we train each successor model S on the acquired data $\mathcal{D}_A^t$ for all t in the range $[0, T]$, evaluating its performance on a held-out test set (distinct from $\mathcal{U}$). We compare the performance achieved in this case to that obtained using an i.i.d. training set of the same size.

We run this experiment ten times, averaging results to create summary learning curves, as shown

[2] Recent works have shown that AL is effective for these tasks even when using modern, neural architectures (Zhang et al., 2017; Shen et al., 2018), but do not address our primary concerns regarding replicability and transferability.

in Figure 1. All reported results, including i.i.d. baselines, are averages of ten experiments, each conducted with a distinct $\mathcal{D}_w$. These learning curves quantify the comparative performance of a particular model achieved using the same amount of supervision, but elicited under different acquisition models. For each model, we compare the learning curves of each acquisition strategy, including active acquisition using a *foreign* model and subsequent transfer, active acquisition without changing models (i.e., typical AL), and the baseline strategy of i.i.d. sampling.

4 Tasks

We now briefly describe the models, datasets, acquisition functions, and implementation details for the experiments we conduct with active learners for text classification (4.1) and NER (4.2).

4.1 Text Classification

Models We consider three standard models for text classification: Support Vector Machines (SVMs), Convolutional Neural Networks (CNNs) (Kim, 2014; Zhang and Wallace, 2015), and Bidirectional Long Short-Term Memory (BiLSTM) networks (Hochreiter and Schmidhuber, 1997). For SVM, we represent texts via sparse, TF-IDF bag-of-words (BoW) vectors. For neural models (CNN and BiLSTM), we represent each document as a sequence of word embeddings, stacked into an $l \times d$ matrix where l is the length of the sentence and d is the dimensionality of the word embeddings. We initialize all word embeddings with pre-trained GloVe vectors (Pennington et al., 2014).

We initialize vector representations for all words for which we do not have pre-trained embeddings uniformly at random. For the CNN, we impose a maximum sentence length of 120 words, truncating sentences exceeding this length and padding shorter sentences. We used filter sizes of 3, 4, and 5, with 128 filters per size. For BiLSTMs, we selected the maximum sentence length such that 90% of sentences in $\mathcal{D}^t$ would be of equal or lesser length.[3] We trained all neural models using the Adam optimizer (Kingma and Ba, 2014), with a learning rate of 0.001, $\beta_1 = 0.9$, $\beta_1 = 0.999$, and $\epsilon = 10^{-8}$.

Datasets We perform text classification experiments using four benchmark datasets. We reserve 20% of each dataset (sampled at i.i.d. random) as test data, and use the remaining 80% as the pool of unlabeled data $\mathcal{U}$. We sample 2.5% of the remaining documents randomly from $\mathcal{U}$ for each $\mathcal{D}_w$. All models receive the same $\mathcal{D}_w$ for any given experiment.

- **Movie Reviews**: This corpus consists of sentences drawn from movie reviews. The task is to classify sentences as expressing positive or negative sentiment (Pang and Lee, 2005).

- **Subjectivity**: This dataset consists of statements labeled as either objective or subjective (Pang and Lee, 2004).

- **TREC**: This task entails categorizing questions into 1 of 6 categories based on the subject of the question (e.g., questions about people, locations, and so on) (Li and Roth, 2002). The TREC dataset defines standard train/test splits, but we generate our own for consistency in train/validation/test proportions across corpora.

- **Customer Reviews**: This dataset is composed of product reviews. The task is to categorize them as positive or negative (Hu and Liu, 2004).

4.2 Named Entity Recognition

Models We consider transfer between two NER models: Conditional Random Fields (CRF) (Lafferty et al., 2001) and Bidirectional LSTM-CNNs (BiLSTM-CNNs) (Chiu and Nichols, 2015).

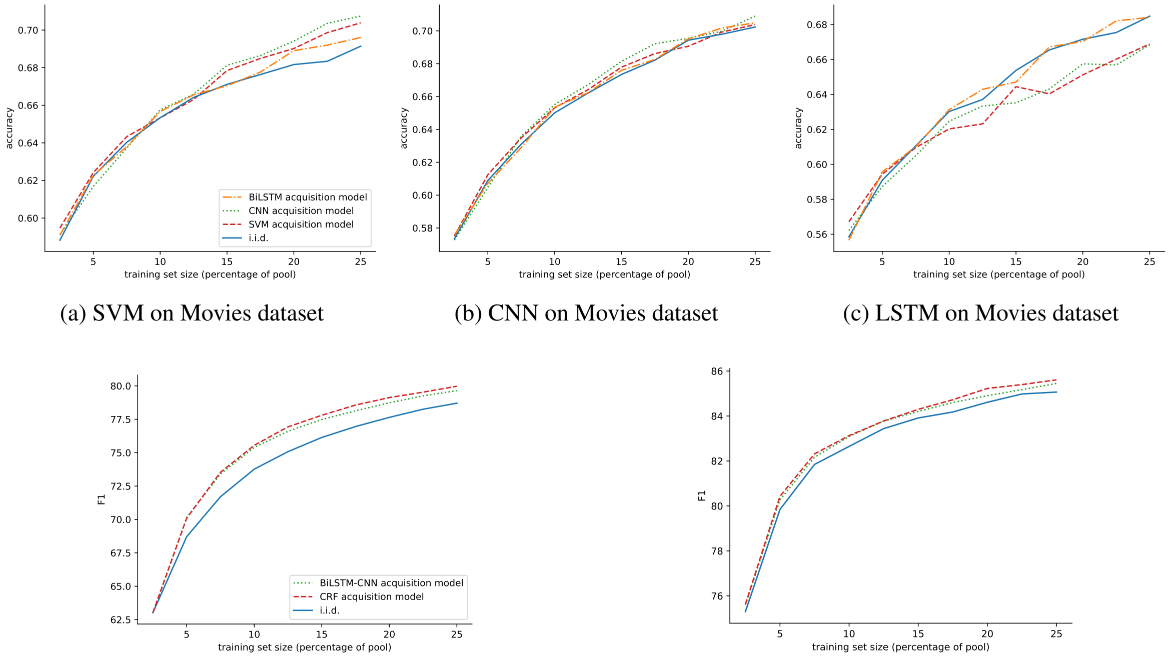
For the CRF model we use a set of features including word-level and character-based embeddings, word suffix, capitalization, digit contents, and part-of-speech tags. The BiLSTM-CNN model[4] initializes word vectors to pre-trained GloVe vector embeddings (Pennington et al., 2014). We learn all word and character level features from scratch, initializing with random embeddings.

Datasets We perform NER experiments on the CoNLL-2003 and OntoNotes-5.0 English datasets. We used the standard test sets for both corpora, but merged training and validation sets to form $\mathcal{U}$. We initialize each $\mathcal{D}_w$ to 2.5% of $\mathcal{U}$.

- **CoNLL-2003**: Sentences from Reuters news with words tagged as person, location, organization, or miscellaneous entities using an IOB

[3] Passing longer sentences to the BiLSTM degraded performance in preliminary experiments.

[4] Implementation of BiLSTM-CNN is based on https://github.com/asiddhant/Active-NLP.



(a) SVM on Movies dataset

(b) CNN on Movies dataset

(c) LSTM on Movies dataset

(d) CRF on OntoNotes dataset

(e) BiLSTM-CNN on OntoNotes dataset

Figure 2: Sample learning curves for the text classification task on the Movie Reviews dataset and the NER task on the OntoNotes dataset using the maximum entropy acquisition function (we report learning curves for all models and datasets in the Appendix). Individual plots correspond to successor models. Each line corresponds to an acquisition model, with the blue line representing an i.i.d. baseline.

scheme (Tjong Kim Sang and De Meulder, 2003). The corpus contains 301,418 words.

- **OntoNotes-5.0**: A corpus of sentences drawn from a variety of sources including newswire, broadcast news, broadcast conversation, and web data. Words are categorized using eighteen entity categories annotated using the IOB scheme (Weischedel et al., 2013). The corpus contains 2,053,446 words.

4.3 Acquisition Functions

We evaluate these models using three common active learning acquisition functions: classical uncertainty sampling, query by committee (QBC), and Bayesian active learning by disagreement (BALD).

Uncertainty Sampling For text classification we use the entropy variant of uncertainty sampling, which is perhaps the most widely used AL heuristic (Settles, 2009). Documents are selected for annotation according to the function

$$\underset{\mathbf{x}\in\mathcal{U}}{\operatorname{argmax}} -\sum_{j} P(y_j|\mathbf{x}) \log P(y_j|\mathbf{x}),$$

where $\mathbf{x}$ are instances in the pool $\mathcal{U}$, j indexes potential labels of these (we have elided the instance index here) and $P(y_j|\mathbf{x})$ is the predicted probability that $\mathbf{x}$ belongs to class y_j (this estimate is implicitly conditioned on a model that can provide such estimates). For SVM, the equivalent form of this is to choose documents closest to the decision boundary.

For the NER task we use maximized normalized log-probability (MNLP) (Shen et al., 2018) as our AL heuristic, which adapts the least confidence heuristics to sequences by normalizing the log probabilities of predicted tag sequence by the sequence length. This avoids favoring selecting longer sentences (owing to the lower probability of getting the entire tag sequence right).

Documents are sorted in ascending order according to the function

$$\max_{y_1,\ldots,y_n} \frac{1}{n} \sum_{i=1}^{n} \log P(y_i|y_1, ..., y_{n-1}, \mathbf{x})$$

Where the max over y assignments denotes the most likely set of tags for instance $\mathbf{x}$ and n is the sequence length. Because explicitly calculating the most likely tag sequence is computationally expensive, we follow (Shen et al., 2018) in using a greedy decoding (i.e., beam search with width 1) to determine the model's prediction.

Text classification

	Acquisition model							
		10% of pool				20% of pool		
Successor	i.i.d.	SVM	CNN	LSTM	i.i.d.	SVM	CNN	LSTM
				Movie reviews				
SVM	65.3	**65.3**	65.8	65.7	68.2	**69.0**	69.4	68.9
CNN	65.0	65.3	**65.5**	65.4	69.4	69.1	**69.5**	69.5
LSTM	63.0	62.0	62.5	**63.1**	67.2	65.1	65.8	**67.0**
				Subjectivity				
SVM	85.2	**85.6**	85.3	85.5	87.5	**87.6**	87.4	87.6
CNN	85.3	85.2	**86.3**	86.0	87.9	87.6	**88.4**	88.6
LSTM	82.9	82.7	82.7	**84.1**	86.7	86.3	85.8	**87.6**
				TREC				
SVM	68.5	**68.3**	66.8	68.5	74.1	**74.7**	73.2	74.3
CNN	70.9	70.5	**69.0**	70.0	76.1	77.7	**77.3**	78.0
LSTM	65.2	64.5	63.6	**63.8**	71.5	72.7	71.0	**73.3**
				Customer reviews				
SVM	68.8	**70.5**	70.3	68.5	73.6	**74.2**	72.9	71.1
CNN	70.6	70.9	**71.7**	68.2	74.1	74.5	**74.8**	71.5
LSTM	66.1	67.2	65.1	**65.9**	68.0	66.6	66.5	**66.3**

Table 1: Text classification accuracy, evaluated for each combination of acquisition and successor models using uncertainty sampling. Accuracies are reported for training sets composed of 10% and 20% of the document pool. Colors indicate performance relative to i.i.d. baselines: Blue indicates that a model fared better, red that it performed worse, and black that it performed the same.

Named Entity Recognition

	Acquisition Model					
		10% of pool			20% of pool	
Successor	i.i.d.	CRF	BiLSTM-CNN	i.i.d.	CRF	BiLSTM-CNN
			CoNLL			
CRF	69.2	**70.5**	70.2	73.6	**74.4**	74.0
BiLSTM-CNN	87.4	87.4	**87.8**	89.1	89.6	**89.6**
			OntoNotes			
CRF	73.8	**75.5**	75.4	77.6	**79.1**	78.7
BiLSTM-CNN	82.6	83.1	**83.1**	84.6	85.2	**84.9**

Table 2: F1 measurements for the NER task, with training sets comprising 10% and 20% of the training pool.

Query by Committee For our QBC experiments, we use the bagging variant of QBC (Mamitsuka et al., 1998), in which a committee of n models is assembled by sampling with replacement n sets of m documents from the training data ($\mathcal{D}^t$ at each t). Each model is then trained using a distinct resulting set, and the pool documents that maximize their disagreement are selected. We use 10 as our committee size, and set m as equal to the number of documents in $\mathcal{D}^t$.

For the text classification task, we compute disagreement using Kullback-Leibler divergence (McCallum and Nigamy, 1998), selecting docu-

Dataset	# Classes	# Documents	Examples per Class
Movie Reviews	2	10662	5331, 5331
Subjectivity	2	10000	5000, 5000
TREC	6	5952	1300, 916, 95, 1288, 1344, 1009
Customer Reviews	2	3775	1368, 2407

Table 3: Text classification dataset statistics.

ments for annotation according to the function

$$\underset{\mathbf{x}\in\mathcal{U}}{\operatorname{argmax}}\frac{1}{C}\sum_{c=1}^{C}\sum_{j}P_c(y_j|\mathbf{x})\log\frac{P_c(y_j|\mathbf{x})}{P_C(y_j|\mathbf{x})}$$

where $\mathbf{x}$ are instances in the pool $\mathcal{U}$, j indexes potential labels of these instances, and C is the committee size. $P_c(y_j|\mathbf{x})$ is the probability that $\mathbf{x}$ belongs to class y_j as predicted by committee member c. $P_C(y_j|\mathbf{x})$ represents the consensus probability that $\mathbf{x}$ belongs to class y_j, $\frac{1}{C}\sum_{c=1}^{C}P_c(y_j|\mathbf{x})$.

For NER, we compute disagreement using the average per word vote-entropy (Dagan and Engelson, 1995), selecting sequences for annotation which maximize the function

$$-\frac{1}{n}\sum_{i=1}^{n}\sum_{m}\frac{V(y_i,m)}{C}\log\frac{V(y_i,m)}{C}$$

where n is the sequence length, C is the committee size, and $V(y_i,m)$ is the number of committee members who assign tag m to word i in their most likely tag sequence. We do not apply the QBC acquisition function to the OntoNotes dataset, as training the committee for this larger dataset becomes impractical.

Bayesian AL by Disagreement We use the Monte Carlo variant of BALD, which exploits an interpretation of dropout regularization as a Bayesian approximation to a Gaussian process (Gal et al., 2017b; Siddhant and Lipton, 2018). This technique entails applying dropout at test time, and then estimating uncertainty as the disagreement between outputs realized via multiple passes through the model. We use the acquisition function proposed in (Siddhant and Lipton, 2018), which selects for annotation those instances that maximize the number of passes through the model that disagree with the most popular choice:

$$\underset{\mathbf{x}\in\mathcal{U}}{\operatorname{argmax}}(1-\frac{\text{count}(\text{mode}(y_{\mathbf{x}}^1,...,y_{\mathbf{x}}^T))}{T})$$

where $\mathbf{x}$ are instances in the pool $\mathcal{U}$, $y_{\mathbf{x}}^i$ is the class prediction of the ith model pass on instance x, and T is the number of passes taken through the model. Any ties are resolved using uncertainty sampling over the mean predicted probabilities of all T passes.

In the NER task, agreement is measured across the entire sequence. Because this acquisition function relies on dropout, we do not consider it for non-neural models (SVM and CRF).

5 Results

We compare transfer between all possible (acquisition, successor) model pairs for each task. We report the performance of each model under all acquisition functions both in tables compiling results (Table 1 and Table 2 for classification and NER, respectively) and graphically via learning curves that plot predictive performance as a function of train set size (Figure 2).

We report additional results, including all learning curves (for all model pairs and for all tasks), and tabular results (for all acquisition functions) in the Appendix. We also provide in the Appendix plots resembling 1a for all (model, acquisition function) pairs that report the difference between performance under standard AL (in which acquisition and successor model are the same) and that under commensurate i.i.d. data, which affords further analysis of the gains offered by standard AL. For text classification tasks, we report accuracies; for NER tasks, we report F1.

To compare the learning curves, we select incremental points along the x-axis and report the performance at these points. Specifically, we report results with training sets containing 10% and 20% of the training pool.

6 Discussion

Results in Tables 1 and 2 demonstrate that standard AL — where the acquisition and successor models are one and the same — performs incon-

	Successor							
	Movie Reiews		Subjectivity		TREC		Customer Reviews	
Acquisition Model	CNN	LSTM	CNN	LSTM	CNN	LSTM	CNN	LSTM
CNN	–	0.961	–	0.968	–	0.988	–	0.973
LSTM	0.989	–	0.996	–	0.992	–	0.980	–
SVM	0.991	0.961	0.997	0.970	0.990	0.987	0.991	0.974

Table 4: Average Spearman's rank correlation coefficients (over five runs) of cosine distances between test set representations learned with native active learning and distances between those learned with transferred actively acquired datasets, at the end of the AL process. Uncertainty is used as the acquisition function in all cases.

sistently across text classification datasets. In 75% of all combinations of model, dataset, and training set size, there exists some acquisition function that outperforms i.i.d. data. This is consistent with the prior literature indicating the effectiveness of AL. However, when implementing AL in a real, live setting, a practitioner would choose a *single* acquisition function ahead of time. To accurately reflect this scenario, we must consider the performance of individual acquisition functions across multiple datasets. Results for individual AL strategies are more equivocal. In our reported classification datapoints, standard AL outperforms i.i.d. sampling in only a slight majority (60.9%) of cases.

AL thus seems to yield modest (though inconsistent) improvements over i.i.d. random sampling, but our results further suggest that this comes at an additional cost: the acquired dataset may not generalize well to new learners. Specifically, models trained on *foreign* actively acquired datasets tend to underperform those trained on i.i.d. datasets. We observe this most clearly in the classification task, where only a handful of (acquisition, successor, acquisition function) combinations lead to performance greater than that achieved using i.i.d. data. Specifically, only 37.5% of the tabulated data points representing dataset transfer (in which acquisition and successor models differ) outperform the i.i.d. baseline.

Results for NER are more favorable for AL. For this task we observe consistent improved performance versus the i.i.d. baseline in both standard AL data points and transfer data points. These results are consistent with previous findings on transferring actively acquired datasets for NER (Tomanek and Morik, 2011).

In standard AL for text classification, the only (model, acquisition function) pairs that we observe to produce better than i.i.d. results with any regularity are uncertainty with SVM or CNN, and BALD with CNN. When transferring actively acquired datasets, we do not observe consistently better than i.i.d. results with *any* combination of acquisition model, successor model, and acquisition function. The success of AL appears to depend very much on the dataset. For example, AL methods – both in the standard and acquisition/successor settings – perform much more reliably on the Subjectivity dataset than any other. In contrast, AL performs consistently poorly on the TREC dataset.

Our findings suggest that AL is brittle. During experimentation, we also found that performance often depends on factors that one may think are minor design decisions. For example, our setup largely resembles that of Siddhant and Lipton (2018), yet initially we observed large discrepancies in results. Digging into this revealed that much of the difference was due to our use of word2vec (Mikolov et al., 2013) rather than GloVe (Pennington et al., 2014) for word embedding initializations. That small decisions like this can result in relatively pronounced performance differences for AL strategies is disconcerting.

A key advantage afforded by neural models is representation learning. A natural question here is therefore whether the representations induced by the neural models differs as a function of the acquisition strategy. To investigate this, we measure pairwise distances between instances in the learned feature space after training. Specifically, for each test instance we calculate its cosine similarity to all other test instances, inducing a ranking. We do this in the three different feature spaces learned by the CNN and LSTM models, respectively, after sampling under the three acquisition models.

We quantify dissimilarities between the rankings induced under different representations via Spearman's rank correlation coefficients. We repeat this for all instances in the test set, and average over these coefficients to derive an overall similarity measure, which may be viewed as quantifying the similarity between learned feature spaces via average pairwise similarities within them. As reported in Table 4, despite the aforementioned differences in predictive performance, the learned representations seem to be similar. In other words, sampling under foreign acquisition models does not lead to notably different representations.

7 Conclusions

We extensively evaluated standard AL methods under varying model, domain, and acquisition function combinations for two standard NLP tasks (text classification and sequence tagging). We also assessed performance achieved when transferring an actively sampled training dataset from an acquisition model to a distinct successor model. Given the longevity and value of training sets and the frequency at which new ML models advance the state-of-the-art, this should be an anticipated scenario: Annotated data often outlives models.

Our findings indicate that AL performs unreliably. While a specific acquisition function and model applied to a particular task and domain may be quite effective, it is not clear that this can be predicted ahead of time. Indeed, there is no way to retrospectively determine the relative success of AL without collecting a relatively large quantity of i.i.d. sampled data, and this would undermine the purpose of AL in the first place. Further, even if such an i.i.d. sample were taken as a diagnostic tool early in the active learning cycle, relative success early in the AL cycle is not necessarily indicative of relative success later in the cycle, as illustrated by Figure 1a.

Problematically, even in successful cases, an actively sampled training set is linked to the model used to acquire it. We have found that training successor models with this set will often result in performance worse than that attained using an equivalently sized i.i.d. sample. Results are more favorable to AL for NER, as compared to text classification, which is consistent with prior work (Tomanek and Morik, 2011).

In short, the relative performance of individual active acquisition functions varies considerably over datasets and domains. While AL often does yield gains over i.i.d. sampling, these tend to be marginal and inconsistent. Moreover, this comes at a relatively steep cost: The acquired dataset may be disadvantageous for training subsequent models. Together these findings raise serious concerns regarding the efficacy of active learning in practice.

8 Acknowledgements

This work was supported in part by the Army Research Office (ARO), award W911NF1810328.

References

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

Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. 2009. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542.

Jason PC Chiu and Eric Nichols. 2015. Named entity recognition with bidirectional lstm-cnns. *arXiv preprint arXiv:1511.08308*.

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

Ido Dagan and Sean P Engelson. 1995. Committee-based sampling for training probabilistic classifiers. In *Machine Learning Proceedings 1995*, pages 150–157. Elsevier.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017a. Deep bayesian active learning with image data. In *International Conference on Machine Learning*, pages 1183–1192.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017b. Deep bayesian active learning with image data. *CoRR*, abs/1703.02910.

Abbas Ghaddar and Phillippe Langlais. 2018. Robust lexical features for improved neural network named-entity recognition. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1896–1907. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM.

Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer.

Rie Johnson and Tong Zhang. 2016. Supervised and semi-supervised text categorization using lstm for region embeddings. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 526–534. JMLR.org.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.

Naoki Abe Hiroshi Mamitsuka et al. 1998. Query learning strategies using boosting and bagging. In *Machine learning: proceedings of the fifteenth international conference (ICML98)*, volume 1. Morgan Kaufmann Pub.

Andrew Kachites McCallum and Kamal Nigamy. 1998. Employing em and pool-based active learning for text classification. In *Proc. International Conference on Machine Learning (ICML)*, pages 359–367. Citeseer.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the ACL*.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Maria E. Ramirez-Loaiza, Manali Sharma, Geet Kumar, and Mustafa Bilgic. 2017. Active learning: an empirical study of common baselines. *Data Mining and Knowledge Discovery*, 31(2):287–313.

B. Settles. 2009. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison.

Burr Settles. 2012. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114.

Yanyao Shen, Hyokun Yun, Zachary C. Lipton, Yakov Kronrod, and Animashree Anandkumar. 2018. Deep active learning for named entity recognition. In *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. *arXiv preprint arXiv:1808.05697*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2003*, pages 142–147. Edmonton, Canada.

Katrin Tomanek and Katharina Morik. 2011. Inspecting sample reusability for active learning. In *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, pages 169–181.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*.

Ye Zhang, Matthew Lease, and Byron C Wallace. 2017. Active discriminative text representation learning. In *AAAI*.

Ye Zhang and Byron Wallace. 2015. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*.