Active two-phase learning for classification of large datasets with extreme class-skew

Tarun Gupta*
Sedat Gokalp*
targupt@amazon.com
sggokalp@amazon.com
Amazon Inc.
Seattle, Washington, USA

ABSTRACT

Active learning is a commonly used technique to reduce the amount of labeled data necessary for supervised learning. In this paper, we focus on collection of labeled examples in a domain with large unlabeled dataset and extreme class imbalance. This scenario presents several challenges to Active learning. Traditional active learning strategies can face acute difficulty in locating minority class examples and can fail completely due to the well-known cold start problem. The problem is further complicated by scale as for large datasets it can be expensive to execute Active learning computations on the whole domain set. Additionally, the active learning strategies can turn out to be impractical or inefficient for interactive use due to high computation time for the iterative selection cycles. In this paper, we proposed a two-phase approach that takes into account both high-class imbalance and the scale of the input data space. Specifically, our approach employs two active learners in a tiered fashion - first phase active learner efficiently learns a domain classifier (a filter function) defined on the entire input space and second learner tries to efficiently learn final ML classifier defined on the output of filter. The second-phase selects informative instances from a smaller pool of unlabeled examples which doesn't require operating on the full dataset. The proposed method allows active learning to be applied to large datasets with class skew. The two-phases are interleaved (rather than isolated) and allow for a bi-directional flow of information. The combined two-phase learner progressively expands knowledge of input data space and uses successive first phase and second phase strategies to switch between learning the decision boundary and expanding domain boundary. Given labeled data at certain iteration, the second-phase focuses on exploiting the decision boundary (up to a performance threshold) and then, first-phase focuses on exploiting given information to intelligently search and expand the domain. We demonstrate the effectiveness of our strategy for product classification on sample of Amazon catalog dataset. Our results show that the proposed method achieves a fast solution with competitive performance in extreme imbalanced setting.

*Both authors contributed equally to this research.

Woodstock '18, June 03–05, 2018, Woodstock, NY
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

CCS CONCEPTS

• **Computing methodologies → Active learning settings**; • **Theory of computation → Active learning**.

KEYWORDS

active learning, domain classification, exploration exploitation

ACM Reference Format:
Tarun Gupta and Sedat Gokalp. 2018. Active two-phase learning for classification of large datasets with extreme class-skew. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Classification is supervised machine learning method that relies on collecting labeled dataset for classifying unseen examples. In practice, classification tasks are presented with a large pool of unlabeled data and labeling the full set can be impossible due to annotation costs. Several strategies have been proposed to optimize the selection of the most relevant examples for labeling, a process referred to as active learning.

1.1 Challenge of data scale and skew

Machine learning applications at Amazon have several additional challenges than classical textbook cases. We often tackle with large scale data as the number of products aka ASINs (ASIN: Amazon Standard Identification Number) go well beyond billions. The Amazon catalog is steadily growing as expected due to addition of new items by sellers, introduction of new marketplaces etc. Amazon data is found to be skewed in many dimensions. For binary ASIN classification tasks, data can be heavily skewed (positive data can be orders of magnitude smaller than the negative data). A skewed classification dataset is one in which there are many more samples of one class (called majority class) than the other class (called minority class). This is a common scenario in many other real-world applications (like fraud detection) where minority examples (fraud cases for fraud detection) are rare and sparse compared to majority examples which are abundant. Basic approaches like uniform sampling do not guarantee coverage of regions of interest that are interesting for drawing the classification boundary. Due to class imbalance, it can be difficult even for active learning (AL) to collect samples of minority class which are necessary for learning the classifier. Traditional strategies like random sampling or even Active learning strategies can be ineffective or fail completely [19], [20]. The scale of data presents the second challenge to active learning. Active learning strategies operate over a fixed domain space and pick examples from the domain space to refine the classifier. If the domain space is too large, it can be expensive to execute the classifier on the whole space. Most previous active learning literature is indeed performed on small or moderate-scale datasets [18], [3].

1.2 High-skew active learning

In cases with extreme class imbalance, traditional active learning strategies can fail to locate minority examples. To address this challenge, information retrieval systems have been suggested. [19], [20] have proposed guided learning (i.e. "search" based technique) where the annotator is tasked with finding class-specific instances. Guided learning can be effective in quickly finding minority examples while active learning strategies can fail. In addition, a hybrid strategy that employs both search and active labeling outperforms either standalone approach. In text classification, it is convenient to use keyword searches to query for examples. Even in image classification problems with high-skew, extending active learning algorithms with keyword searches have been shown to improve performance [22]. Theoretical analysis has also shown the power of adding a search component into active learning [21].

1.3 High-skew Machine Learning

Another consideration is that given an imbalanced dataset, we want to learn from such dataset with high predictive performance. Cascades and related techniques use the idea of multiple classifiers to focus on the region of interest. Sculley et al. [23] use a series of models to find adversarial ads - a high-recall coarse model that rejects part of data space, followed by a finely grained models were used to detect positive data (adversarial class). They find that using cascade model leads to significant improvements in performance. Yih et. al. [24] similarly use a two-phase approach to email-spam filter models. Figure 1 from [24] illustrates how a two-phase approach can learn a more complex (and accurate) decision boundary for data which is not linearly separable in the original data space.

In this paper we address the challenges in active learning for imbalanced data classification in large datasets. In our setting, examples are organized into groupings called *indices* or *buckets*. Each index can contain any number of examples and occupies a region of space. To motivate the notion of index lets consider sentence sentiment classification. In sentence classification, sentences are examples and they are grouped at paragraph-level. In this case, "paragraph" can be an index. Section 2.1.1 describes index examples from Amazon catalog. The basic idea of the two-phase approach is to breakdown the learning (active learning and classification) into two phases. In the first-phase, a novel active learner is employed to collect labels for these "indices" and discover positive indices. The first phase employs an initial seed search query to gather the "first" positive index. Examples from list of indices labeled positive form the domain set and define the domain boundary. The first-phase essentially rejects all negative labeled indices (parts of input space)

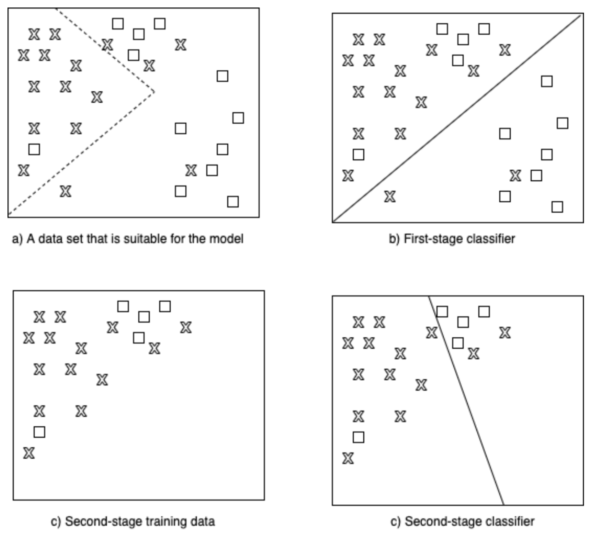

a) A data set that is suitable for the model
b) First-stage classifier
c) Second-stage training data
c) Second-stage classifier

Figure 1: Illustration of the Two-Stage Filtering Approach: Squares and Crosses represent positive and negative examples, respectively. Figure reproduced from [24]

and only allows positive labeled parts to enter second-phase. In the second-phase, a traditional active learner is employed within the domain set to collect labels for examples and refine the decision boundary for machine learning classification. The first-phase classifier is a coarse-model intended to capture positive data with high-recall and the second-phase classifier works on improving precision of the domain set. During an active learning session, the two classifiers complement each other and relay information to each other. When the second-phase classifier updates, the first-phase active learner searches outward to discover candidate indexes that are likely False-Negatives (FNs). When the first-phase classifier updates and adds new indices, second-phase classifier focuses on improving precision on new domain set.

Our approach differs from existing ones in several aspects. Firstly, two complementary classifiers are employed to collectively sample most informative instances. The two-phases distribute the burden of optimizing for recall (first-phase) and precision (second-phase). Our algorithm switches between the two strategies to progressively improve knowledge of input space by refining the decision boundary and expanding the domain. Secondly, the two-phase classifier doesn't need to classify all examples and only requires a search functionality over the full input space. This is in contrast to traditional active learning which executes on full input space and becomes computationally expensive for large datasets. Thirdly, while previous studies indicate that a two-phase (or "cascade") approach can be used to improve the predictive performance of classification on fully labeled imbalanced dataset, our paper utilizes the two-phase framework to perform active learning and improve computational performance (training/prediction speed) of the system. Finally, while traditional active learning collects labels at instance level which can be expensive, we collect labels at two levels of granularity. In the first-phase, labels are collected at a coarse level of granularity to quickly explore the input space and identify the domain-set.

The rest of this paper is organized as follows. First, we describe the two-phase framework and introduce the key components. Then, we describe our experiments to evaluate our approaches on two data sets in the Amazon Global Trade Services and Direct Import Products domain. Finally, we discuss the results of our experiments.

2 TWO-PHASE LEARNING

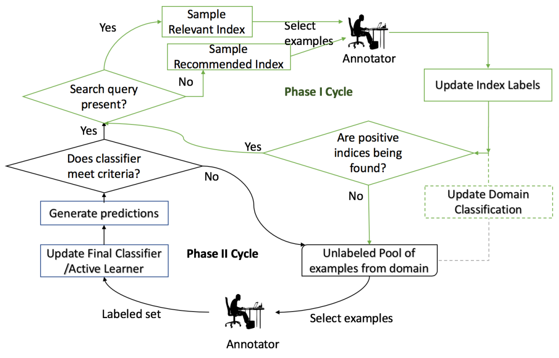


Figure 2: Two-phase learning cycle

We now describe our new method, two-phase learning and introduce its key components. The basic idea is to breakdown the problem of building quality classifier from a highly-skewed large dataset into phases. In phase-I, the goal of the system is to query samples to build a high-recall coarse model that defines the domain space (boundary). A key component of phase-I is the index model. The large volume of input space is sliced into a smaller number of indices. These indices have several properties as discussed in 2.1. It is actually the indices rather than the individual examples that are labeled in phase-I. List of positive labeled indices is the domain classification which defines the domain space for traditional active learning. This model rejects part of the large input space. In the phase-II, system uses traditional AL strategies on the smaller subspace to find and refine the final classification model boundary.

The two-phase system leverages two active-learning cycles, one for each phase of the two-phase classifier. During active learning, the system leverages a keyword-search API, a hash-map that can provide examples present in an index, a list of indices that define the working domain set for phase-II, a binary classification model and its predictions. The learning process starts with an initial "seed-search" query (based on class rationale) to retrieve a sample of positive examples. The seed search query can be the class name. For example, "cigar" can be a seed search to identify "cigars" in the catalog. These examples are naturally distributed across various indices. The relevance AL ranker ranks the unlabeled indices by relevance i.e. dominance in the relevant examples. For example, to identify "cigar" in Amazon catalog a keyword-search for "cigar" may return 1000 examples which may belong to different indices i_1, i_2, i_3 ranked by relevance$\{i_1 : 700, i_2 : 200, i_3 : 100\}$. The system iteratively presents a sample of examples from most relevant index (i_1 followed by i_2 and so on) to acquire index labels. Once an index is

Algorithm 1 Active Two-Phase Learning Algorithm

```
1:  $b$: batch size
2:  $I$: set of all candidate indices
3:  $U_I$: pool of unlabeled indices
4:  $I_p = \emptyset$: pool of indices labeled positive
5:  $I_n = \emptyset$: pool of indices labeled negative
6:  $\mathcal{D}_{in}$: set of all examples inside domain boundary defined by $I_p$
7:  $\mathcal{D}_{out} = \mathcal{D} - \mathcal{D}_{in}$: set of all examples outside domain boundary

8:  $L = \emptyset$: pool of labeled examples in domain $\mathcal{D}_{in}$
9:  $U = \emptyset$: pool of unlabeled examples in domain $\mathcal{D}_{in}$
10: $M = \emptyset$: model trained on labeled examples
11: $probs$: model $M$ predicted class-probabilities for examples
12: repeat
13:   while phase-I stopping criterion is not met do
14:     # Phase I: Domain Classifier
15:     $q$ search query
16:     select top index $i$ using first-phase AL strategy
        $selectTopIndex(q, M, I_p, probs, U_I)$ # Alg. 2
17:     select $K$ examples from index $i$ for human review
18:     if one positive example is found then
19:       index $i$ is labeled positive and added to positive labeled
          indices $I_p$
20:       retrieve examples from index $i$ and add them to domain
          set $\mathcal{D}_{in}$
21:     else
22:       add index $i$ to $I_n$
23:     end if
24:     $U_I = U_I - i$
25:   end while
26:   while phase-II stopping criterion is not met do
27:     # Phase II: Final Classifier
28:     rank examples from $U = \mathcal{D}_{in} - L$ using selection strategy
29:     label $b$ examples and add them to $L$
30:     train the learner on $L$ and execute on $\mathcal{D}_{in}$
31:     populate $probs$ for a tiny fraction $\epsilon$ of examples in un-
        charted space $\mathcal{D}_{out}$
32:   end while
33:   Switch to Phase I
34: until both criterion are met
```

labeled positive, that index is added to the domain classification and all examples in that index are added to domain-set $\mathcal{D}_{in}$. Once an index is labeled negative, phase-I AL eliminates the negative-index for further exploration. The system exploits information received so far to continue to find relevant or recommended indices (see section 2.1.2 for details on rankers and Algorithm 2) and collect labels for these indices. The system also auto-populates search queries to help locate positive examples (hence, indices) and ease the burden on annotator. After new positive indices are labeled, they are used to extend the domain classification and positive indices continue to add new regions to the domain space. The domain classification continues to enhance knowledge of the input space at a coarse-level, and enables smarter exploration of unlabeled indices. For example, the neighborhood ranker ranks unlabeled indices by "closeness" to existing positive indices. The goal of phase-I is to find a coarse domain boundary in a input data space sliced at index-level granularity such that the domain contains as many positive examples as possible. The phase-I active learning loop continues until no more relevant indices can found.

While phase-I reveals part of input-space that contains positive data with high-recall, such an approach is likely include a lot of negative data in the domain space. Phase-II trains a binary classifier within this subspace to improve performance of final classification. Phase-II active learning focuses on region near the final classification decision boundary and iteratively refines it (up to convergence criteria). Phase-II classifier can also unveil promising candidate indices in unexplored region of input space based on location of those indices in decision space. Phase-II combined with Phase-I continue to refine the final decision boundary and expand the domain boundary until improvements stop happening.

Algorithm 1 summarizes our proposed method and figure 2 visualizes the dual active-learning cycles. In following sections, we dive deeper into some key components.

2.1 Phase I: Active *Index* Learning for Domain Classification

The goal of Active *Index* Learning is to perform high-recall retrieval i.e. sample as much from the minority class as possible. In our setting, examples are naturally organized into tight groupings called *indices* or *buckets* and it is the indices, instead of individual examples, that are labeled in Phase I. An index mapping $\mathcal{F}$ is defined for any point x in input space $\mathcal{X}$ which maps elements in input space $\mathcal{F} \in \mathcal{X} \rightarrow \mathcal{I}$ to a bucket $i \in \mathcal{I}$. The domain classifier is a function $h : \mathcal{I} \in \mathcal{R} \rightarrow \{-1, 0, 1\}$ that evaluates an index $i \in \mathcal{I}$ in the input data space. We want to find all possible indices where $h(\mathcal{F}(x)) = 1$ i.e. filter list of indices to keep ones with positive label. This approach is motivated by the fact that given an input space with a large volume, for space exploration, it may be cheaper to acquire labels at coarse index-level of granularity rather than directly acquire example labels. For example, an input space with 10MM examples bucketed across 10 indices will require only 10 index-labels. The labels provided for the index may not be appropriate at the example-level of granularity but still provide a coarse understanding of space.

An index i satisfies several properties. First, index stores a list of examples and hence, represents a region of input space. Second, we assume that every example in a index labeled negative is actually negative, whereas at least one instance in a index labeled positive is actually positive. This means that once an index is labeled positive all examples within the index are included for Phase-II. On the other hand, if an index is labeled negative, all examples within the index are eliminated. Collectively all indices labeled positive defined the domain space. Third, each index has a neighborhood map. Given an index i from a list of indices $\mathcal{I}$, the neighborhood $N[i]$ of i is a subset of indices from $\mathcal{I}$. This mapping allows phase-I AL to explore surrounding regions and expand the domain space.

For example, given an input space with 10 examples $\{x_i\}_1^{10} \in \mathcal{X}$ and 10 examples bucketed across 4 indices s.t. $\{i_1 : [x_1, x_2, x_4], i_2 : [x_3, x_5, x_6], i_3 : [x_7, x_8], i_4 : [x_9, x_{10}]\}$ with a neighborhood map $\{i_1 : [i_2, i_4], i_2 : [i_1], i_3 : []\}$, if i_1, i_3 have been labeled positive the domain $\mathcal{D}_{in}$ contains all examples present in positive-indices i_1, i_3 i.e. $\mathcal{D}_{in} = [x_1, x_2, x_4, x_7, x_8]$. Illustrative examples of indices and neighborhood maps are provided in next section.

2.1.1 Indexing techniques. Amazon catalog can be indexed with respect to various dimensions including keywords in key textual attributes, various other key attributes and any additional calculated values we introduce. We can create hashing mechanisms ranging from completely data agnostic and cheap to data aware and expensive.

Locality-sensitive hashing (LSH), for example, reduces the dimensionality of high-dimensional data. LSH ensures that (1) similar products map to the same buckets with high probability, and (2) similarity between two ASINs can be estimated based on similarity of their hash values. LSH is very cheap to fit and predict, however may have a very unbalanced data separation. This may be addressed through more complex clustering solutions with an added cost of execution.

k-nearest neighbor maps can be created for each index key. Some indexing mechanisms like LSH inherently provides kNN maps for free, while some others require analysis. For example, we can build language models to discover the neighbors of "bag" keyword are "tote", "purse", and "container". Similarly, graph distance can be used for object type to discover the neighbors. For example, neighbors of object type "scented candles" are "jar candles" and "pillars."

2.1.2 Ranker Types. Active learning in phase-I employs several rankers that utilize existing information (domain) to explore the unexplored input space and find new relevant and promising indices (i.e. regions of space) to expand the domain (and increase recall). This includes finding indices which are surrounding current domain (i.e. indices) and indices which are likely to contain positive examples based on phase-II machine learning model feedback. Algorithm 2 highlights these rankers in action during phase-I active learning.

Relevant Indices Finding positive examples is difficult in high-skew domains. This ranker queries indices that are dominant in annotator's search queries.

Probe Neighbouring Indices This ranker queries indices that are closest to the list of already labeled positive indices. This ranker is based on the assumption that similar inputs should have similar outputs. The AL goal is to generate queries be informative and also close to where we started identify positive examples, since deviating too far from initial region is less likely to return similar examples.

Probe Positive Predicted Indices Finding positive examples is difficult in high-skew domains. This ranker queries indices that the current phase-II classifier predicts as positive. We expect that this simple approach can find positive examples and improve recall. To surface positive-predicted indices we maintain a tiny sampling ϵ of unlabeled indices and model's predictions. Predictions for index-samples can help surface indices with high concentration of positive examples. In addition to simplicity, this approach does not require additional training cost and little prediction cost.

2.1.3 Recommended Search. To conduct a comprehensive exploration of data space and discover new regions, it is necessary to try alternative search queries that can locate minority class examples. For example, for identifying "candles" in Amazon catalog, user may

Algorithm 2 $selectTopIndex(q, M, I_p, probs, U_I)$
Phase I Selection Strategy

```
1:  $\mathcal{S}$: Search
2:  if $q == \emptyset$ then
3:      # Discovery
4:      Toss a coin $u \sim Unif(0,1)$
5:      if $u < w_1$ then
6:          rank unlabeled indices by neighborhood to already labeled
            positive indices $I_p$
7:      else if $w_1 < u < w_2$ then
8:          rank indices using $probs$ to find most positive-predicted
            unlabeled indices
9:      else
10:         populate $q$ with a recommended search query
11:     end if
12: else
13:     # Relevance
14:     retrieve relevant docs for $q$ using search $\mathcal{S}$
15:     use final model $M$ predicted probability scores (if available)
        to weight the relevant docs
16:     aggregate relevant documents by index and compute
        relevance-scores for each index
17:     rank unlabeled indices by relevance-score
18: end if
19: return top ranking unlabeled index
```

want to search for "tealights" or "candleholders." It can be strenuous for the user to construct variations of queries. Several approaches can be used to suggest search-terms. Firstly, term tf-idf scores can be computed for samples in positive indices and terms with high importance can be recommended. Secondly, synonyms of initial search term and calculated terms can expand the search set while maintaining closeness to initial region. Finally, positive predictive text features from phase-II model can also be used to recommend terms.

2.2 Phase II: Active learning for Final Classification

The second phase active learner is allowed to selectively query unlabeled instances from positive indices (or buckets). Pool-active learning starts with a pool of unlabeled examples and builds a ranking on the unlabeled set to select a set of samples to be labeled. The two most popular selection strategies used for text classification are uncertainty-based sampling (US) [10] and query-by-committee (QBC) [5] [6]. The main point of differentiation among various algorithms is the approach to quantify the informativeness (or confusion) of an unlabeled example. Other approaches include expected model change [1], expected error reduction [2] or variance reducing strategies to query unlabeled examples. A practical evaluation of these approaches on several datasets [3] indicates that variance and log-loss are best-performing. However, these have a disadvantage of being computationally and memory intensive. QBC and US are common practical choices for scalable systems as they require less memory and computational resources. For a detailed overview of active learning algorithms, see [18].

2.3 Stop Criteria

In first-phase, users are probed with relevant or interesting indices of examples to increase recall. Active learning for first-phase can be concluded when no new positive index values can be generated.

In second-phase, new labels are requested to improve the performance of the classifier. When the performance of the classifier converges, Active learning for second-phase can be stopped.

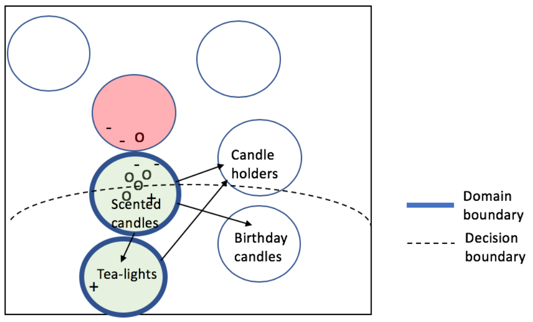


Figure 3: Illustrative Example of Two-phase Active learning Approach to identify "Candles". Amazon catalog is indexed with respect to object type and several indices have been labeled (+) green and (-)red. At given iteration, the domain is defined by positive labeled indices I_p = ['scented-candles', 'tealights']. ASINs within these indices form the domain set $\mathcal{D}_{in}$ for second-phase AL. For first-phase AL, currently 'candle-holders' is top-recommended index per neighborhood ranker and 'birthday-candles' is top-recommended index per positive-predicted ranker. Neighborhood tries to explore regions of space around existing domain boundary rather than searching large volume of input space. Positive-predicted ranker exploits existing decision boundary to find new regions in unexplored space that may contain positive-indices. Second-phase AL samples near the decision boundary via traditional AL.

3 EXPERIMENTS

We conduct experiments to compare performance of proposed strategy AL(dual-phase) with traditional active learning method. The results show that proposed method leads to faster active learning with gains in predictive performance.

3.1 Datasets

In our experiments, we use a 10MM sample of Amazon catalog as the dataset. Data is organized into indices using object type. There are total 25K indices present in the dataset. Dataset is individually annotated for two binary classification tasks - a Trade program class for identifying Category A and a DIPC programs class for identifying Category B.

We created semi-synthetic datasets for ASIN classification using the sample of catalog as the base. Several techniques including a rule-based classifier and machine-learning based classifier were used to help collect positive ground truth data (which is orders of magnitude smaller than negative data).

Category A (Global Trade Service Class) Positive data proportion is 0.043% (i.e. ~ 2000:1 skew)

Category B (Direct Import Products Compliance Class) Positive data proportion for this dataset is 0.026% (i.e ~ 5000:1 skew)

3.2 Other Considerations

Feature Engineering The ASIN texts are processed with stemming, stop-word removal, normalization and vectorized into numerical vectors. n-grams extracted out of text features. A classifier can simply use bag-of-words (term-frequency), or normalize values and calculate tf-idf (term-frequency * inverse-document-frequency). 1-hot encoding for categorical features where each value is represented by a column with normalization (lower-case, accent removal, etc.).

Train-test split A random 80-20 split of data is performed and 80% is used for training. Remaining 20% is used as test data for evaluation.

Search and Inverted Index For providing search functionality and recommended search term to user we create an inverted index using all the documents provided by user during training. This inverted index is a matrix of $m \times n$, where m represents number of terms generated using all documents and n represents number of documents. So, matrix will have value 1 at $m[x][y]$ if $term[x]$ is present in $document[y]$ else 0.

3.3 AL Algorithm

For phase-I active learning, we leverage object type as index. Object type is available for ASINs in US marketplace and naturally has a neighborhood definition where graph distance can be used for object type to discover the neighbors. For example, neighbors of "scented candles" are "jar candles" and "pillars." In one experiment, we also leverage KW-based index. For phase-II active learning, we leverage QBC as the technique to find uncertain instances for final classifier. Logistic Regression is used as the algorithm for final classifier.

The proposed technique is compared against an active learning baseline (QBC) on full dataset. This is the tradition or "single-phase" version of Active learning.

With random sampling, instances are selected from the entire pool of unlabeled data and therefore, due to the high-skew random sampling fails to sample even 1 positive example within 2K instances (for Category A dataset) and 5K instances (for Category B dataset). Initial seed data is provided for both techniques. For dual-phase learner, a seed search term is used to surface initial positive data. For single-phase learner, 15 initial positive examples are explicitly provided through search.

3.4 Acceptance Criteria

For our application, it is mandatory that data is classified at a certain precision level. Specifically, a PPV (positive predictive value) constraint is placed for accepting positive classifications and NPV (negative predictive value) constraint for negative classifications. So, instead of a single threshold that dichotomizes the population, we end up with two cutoffs. The class acceptance-cutoff u is chosen to identify class products with acceptable-certainty (i.e. PPV aka Precision) and class rejection-cutoff l is chosen to identify "not-class" products with acceptable-certainty (i.e. NPV). The threshold u is optimized using a validation set such that the 95% confidence interval (approximated using Wilson's method) for precision is above target-precision. Similarly, the threshold l is optimized for target-NPV. If there are multiple thresholds satisfying both constraints, threshold which maximizes f1-score is chosen. For ASINs with probability-score p between the two cutoffs, uncertainty exists, and prediction of class membership is "unknown" and left for human review. The construction of this "grey-zone" implies three possible classification responses - "negative", "positive" or "unknown":

$$class = \begin{cases} +1, & \text{if } 1 \geq p \geq u \\ -1, & \text{if } 0 \leq p \leq l \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

Given N instances, the fraction of data classified (relative to number of all available examples in dataset) is also known as coverage [7], [8]. The coverage constraint for launch is set at 95% for all sessions.

3.5 Evaluation Metric

Classification accuracy is not a good metric to evaluate classifiers in applications with class imbalance problem. Considering an imbalance ratio of 99 to 1, a classifier classifies everything negative will achieve an accuracy of 99% but of no practical use. We evaluate two measures to compare the selection strategies numerically: labeling effort and area under PR curve (AUC-PR).

Precision-recall (PR) curve displays relationship between precision and recall at all possible thresholds for binary classification. Area under PR curve (AUC-PR) is commonly used for performance evaluation in imbalanced data classification and shows the effectiveness of model.

Active learning aims at keeping the number of labeled samples as low as possible. The labeling effort is then the minimum number of samples needed by each of the other strategies before stopping. This measure indicates how efficiently a selection strategy uses the data.

4 RESULTS

In this section, we will study the performance of proposed "dual-phase" strategy and compare against the traditional "single-phase" AL baseline.

Active learning strategies focus on instances that are close to the decision boundary and confusing to the classifier. Therefore, imbalance ratio of samples collected via active learning is much smaller than imbalance ratio of entire dataset. Tables 1, 2 confirm that active learning approaches ends up with a more balanced labeled data distribution than entire dataset. Dual-phase approach ends up with nearly-perfectly balanced labeled data distribution compared to single-phase approach with a positive proportion of 46.6% (i.e. 527 positive samples out of 1130 active collected sample) vs 13.5% for Category B dataset at 1130 labels and 57.5% vs 6.8% for Category A dataset at 1628 labels. This demonstrates the effectiveness of dual-phase approach in locating minority class examples.

Table 1: Performance for Category B with precision constraint = 86% (*stop criteria not met)

Metric	Dual-Phase	Single-Phase
Labels	1130	1130*
Positive Labels (Ratio)	527 (46.6%)	152 (13.5%)
Precision	93.93%	0%
Recall	96.67%	0%
AUC-PR	96.79%	77.57%

Table 2: Performance for Category A with precision constraint = 95% (*stop criteria not met)

Metric	Dual-Phase	Single-Phase
Labels	1628	1628*
Positive Labels (Ratio)	936 (57.5%)	111 (6.8%)
Precision	95.3%	0%
Recall	93.96%	0%
AUC-PR	96.78%	57.97%

Figure 4 shows the evolution of learning curve for dual-phase active learning experiments. The curve highlights the domain expansion as more and more positive indices are found. The curve also highlights the successive Phase-I and Phase-II loops during the session. AUC-PR curve shows that the performance of the final "dual-phase" classification improves over the course of a session.

Figure 5 shows the evolution of learning curves for single-phase active learning and how it pits against dual-phase learning. The dual-phase AL session is stopped at 1130 labels for Category B and 1628 labels for Category A. Table 1, 2 show the test performance of ML model trained on actively sampled labels. Single-phase learner performs poorly compared to dual-phase learners as measured by AUC-PR (77.57% vs 96.79% for Category B and 57.97% vs 96.78% for Category A) and single-phase learners fail to reach the acceptance criteria.

From computational performance perspective, benchmarking shows that AL(dual-phase) is orders of magnitude faster than AL(baseline) with dual-phase learner predicting only on a fraction of the full dataset. This fraction keeps increasing as shown in Figure 4. For Category B, dual-phase learner only predicts on 0.12% of the data with prediction time of 0.013s while single-phase learner predicts 100% of data in 3.04s. For Category A, dual-phase learner only predicts on 0.07% of data with prediction time of 0.07s vs single-phase learner predicts on 100% of data with prediction time of 3.05s. While our strategy predicts only a portion of instances of dataset, it achieves higher prediction performance than the baseline that processed all instances.

4.1 Comparison of indexing mechanisms

Table 3 shows comparison of performance when using a KW-based index vs an object type index for the Category B class. KW-index leads to much looser groupings of ASINs. At individual index level, each KW-index can add much higher-recall at the cost of low-precision. Therefore, first-phase classifier only requires a handful of KW-indices compared to object type-indices to recover most positive examples. However, KW-index leads to a $\sim$ 3x bigger domain size and hence, requires processing more data and patterns. The AUC-PR comparison shows superior performance using object type index.

Table 3: Performance for Category B with precision constraint = 86% (*stop criteria not met)

KPI	Object Type Index	KW Index
Index Labels (Domain Size)	58 (0.12%)	3 (0.33%)
Domain Precision	21.92%	1.35%
Labels	1130	1130*
Positive Labels (ratio)	527 (46.6%)	450 (39.8%)
AUC-PR	96.79%	85.65%

5 CONCLUSIONS

The two-phase learning framework breaks down the the problem of active learning into two phases of identifying a coarse domain boundary in phase-I and refining a fine decision boundary within the domain in phase-II. Traditional active learning while computationally expensive to apply on a large datasets can be applicable to a smaller subspace via two-phase framework. The active learning method proposed actively focuses on locating minority examples in phase-I. Experimental results indicate that two-phase learning is able to indeed achieve more balanced distribution of labeled data. In phase-II, a traditional AL can be employed to refine the decision boundary in domain and improve classification performance. By successively switching between phase-I and phase-II, more regions of space are added into domain and refined by final classifier. Experiments reveal that dual-phase active learning achieves a 20-40% improvement in AUC-PR over traditional active learning.

In future work, we plan to optimize the components of the two-phase learner. We have identified several promising directions for investigation including leveraging multiple indices to vary index granularity and explore additional rankers for phase-I domain expansion.

REFERENCES

[1] Settles et al. (2007) Multiple-Instance Active Learning, *Neural Information Processing Systems*

[2] Roy, N. & McCallum, A. (2001) Toward optimal active learning through sampling estimation of error reduction, *ICML*

[3] Schein, A.I. & Ungar L.H. (2007) Active Learning for Logistic Regression: An Evaluation, *Machine Learning, Volume 68, Issue 3, October 2007, pages 235-265*

[4] https://w.amazon.com/bin/view/Jade/Science/

[5] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. (1997) Selective sampling using the query by committee algorithm. *Machine Learning, 28:133–168*

[6] H. S. Seung, M. Opper, and H. Sompolinsky. (1992) Query by committee. *In Proc. 5th Annu. Workshop on Comput. Learning Theory, pages 287–294. ACM Press, New York, NY*

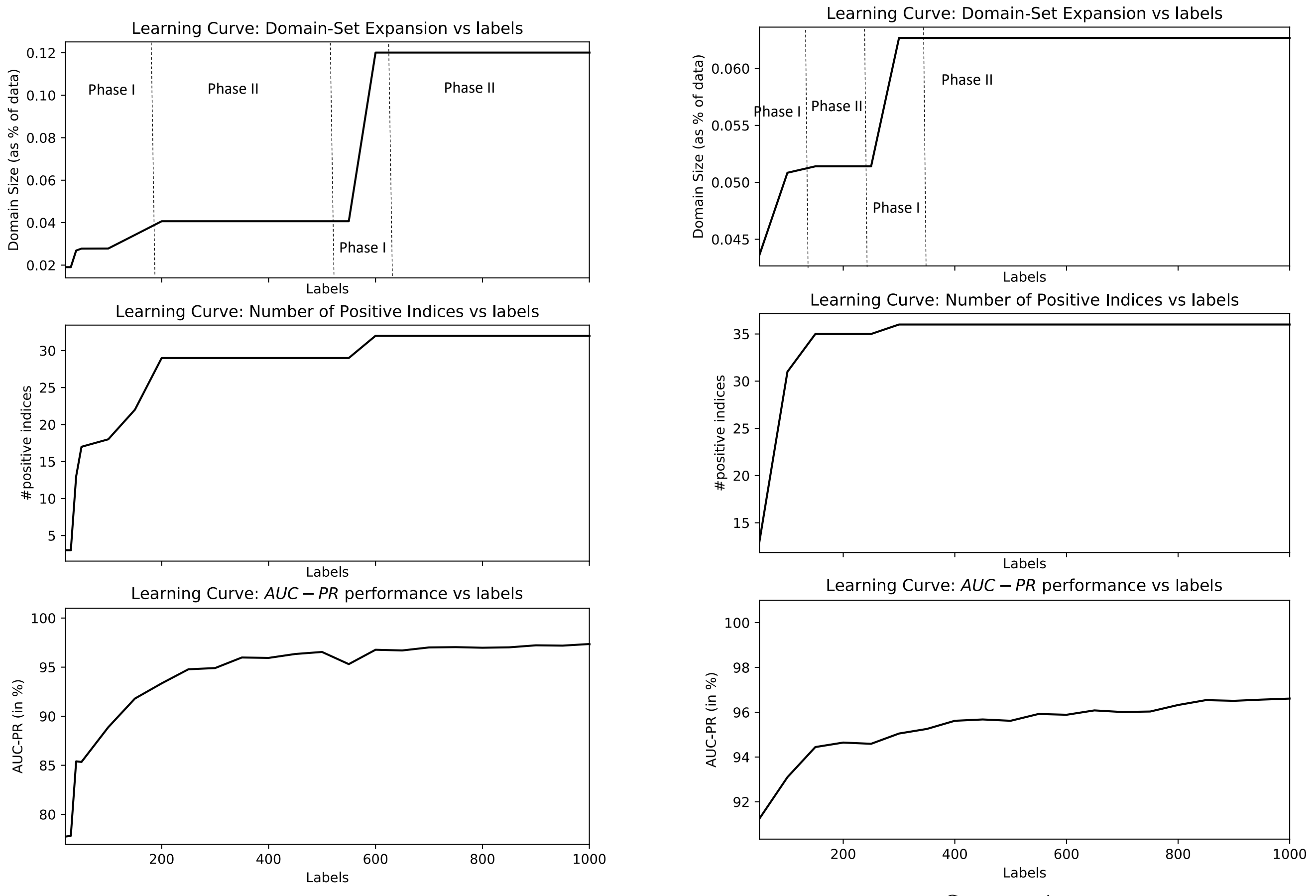
[7] http://robotics.stanford.edu/ ronnyk/glossary.html

[8] Thomas P. Trappenberg (2005) Coverage-performance estimation for classification with ambiguous data. *European Symposium on Artificial Neural Networks, Bruges, Belgium, April 27-29, 2005, Proceedings: 411-416*

[9] Li, M., Sethi, I. K., 2006. Confidence-based active learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (8), 1251–1261*

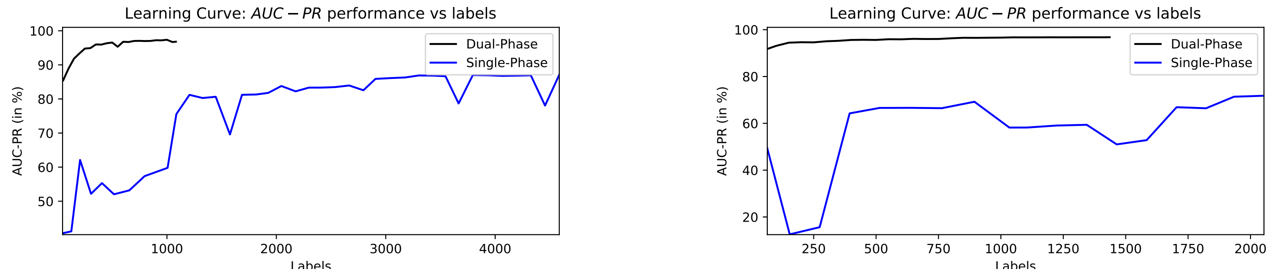
[10] Cohn, D. A., Atlas, L., Ladner, R. E., 1994. *Improving generalization with active learning. Machine Learning 15 (2), 201–221*

[11] G. Beatty, E. Kochis, and M. Bloodgood, The use of unlabeled data versus labeled data for stopping active learning for text classification. *Proceedings of the 2019*



Category B Category A

Figure 4: Learning Curve for Dual-Phase Learning



Category B Category A

Figure 5: Learning Curve for Single-Phase Learning

IEEE 13th International Conference on Semantic Computing (ICSC). Newport Beach, CA, USA: IEEE, January 2019

[12] M. Bloodgood and K. Vijay-Shanker, A method for stopping active learning based on stabilizing predictions and the need for user- adjustable stopping *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009). Boulder, Colorado: Association for Computational Linguistics, June 2009, pp. 39–47*

[13] Schohn Greg and David Cohn. 2000. Less is more: Active learning with support vector machines. *Proceedings of the Seventeenth International Conference on Machine Learning, pp. 839-846*

[14] Zhu Jingbo, Huizhen Wang and Eduard Hovy. 2008. Learning a stopping criterion for active learning for word sense disambiguation and text classification. *In Proceedings of the Third International Joint Conference on Natural Language Processing, pp. 366-372*

[15] F. Laws and H. Schutze 2008. Stopping criteria for active learning of named entity recognition, *in Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1, ser. COLING '08. Stroudsburg, PA, USA: Association for Computational Linguistics, 2008, pp. 465–472*

[16] A. Vlachos, A stopping criterion for active learning, *Computer Speech and Language, vol. 22, no. 3, pp. 295–312, 2008*

[17] Andrew McCallum and Kamal Nigam. Employing EM in pool-based active learning for text classification. *In Proc. of the Int. Conf. on Machine Learning (ICML), pages 359–367, 1998*

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

[19] J Attenberg, S Ertekin. Class Imbalance and Active Learning *Imbalanced Learning: Foundations, Algorithms, and Applications, 101, 2013*

[20] Attenberg, J., and Provost, F. Why label when you can search? Alternatives to active learning for applying human re- sources to build classification models under extreme class imbalance. *In KDD-10. 2010.*

[21] Alina Beygelzimer, Daniel J Hsu, John Langford, and Chicheng Zhang. Search improves label for active learning. *In Advances in Neural Information Processing Systems, pages 3342–3350, 2016.*

[22] Vijayanarasimhan, S., and Grauman, K. Large-scale live active learning: Training object detectors with crawled data and crowds. *In CVPR, 2011*

[23] D. Sculley, Matthew Eric Otey, Michael Pohl, Bridget Spitznagel, John Hainsworth, and Yunkai Zhou. Detecting adversarial advertisements in the wild. *In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '11, pages 274–282, New York, NY, USA, 2011. ACM*

[24] W. Yih, J. Goodman, and G. Hulten. Learning at low false positive rates. *In Proceedings of the Third Conference on Email and Anti-Spam (CEAS), 2006.*