Active Learning with Unbalanced Classes & Example-Generation Queries

Christopher H. Lin
Microsoft
Bellevue, WA
christol@microsoft.com

Mausam
Indian Institute of Technology
Delhi, India
mausam@cse.iitd.ac.in

Daniel S. Weld
University of Washington
Seattle, WA
weld@cs.washington.edu

Abstract

Machine learning in real-world high-skew domains is difficult, because traditional strategies for crowdsourcing labeled training examples are ineffective at locating the scarce minority-class examples. For example, both random sampling and traditional active learning (which reduces to random sampling when just starting) will most likely recover very few minority-class examples. To bootstrap the machine learning process, researchers have proposed tasking the crowd with finding or generating minority-class examples, but such strategies have their weaknesses as well. They are unnecessarily expensive in well-balanced domains, and they often yield samples from a biased distribution that is unrepresentative of the one being learned. This paper extends the traditional active learning framework by investigating the problem of intelligently switching between various crowdsourcing strategies for obtaining labeled training examples in order to optimally train a classifier. We start by analyzing several such strategies (*e.g.*, annotate an example, generate a minority-class example, *etc.*), and then develop a novel, skew-robust algorithm, called `MB-CB`, for the control problem. Experiments show that our method outperforms state-of-the-art `GL-Hybrid` by up to 14.3 points in F1 AUC, across various domains and class-frequency settings.

Introduction

In high-skew environments, where class frequencies are extremely imbalanced, traditional strategies for obtaining labeled training examples perform poorly. Traditional labeling queries, which task crowd workers with labeling randomly-selected or even intelligently-selected examples (*e.g.* via standard active learning) are ineffective because the probability that any given example belongs to the minority-class is virtually zero (Attenberg and Provost 2010). Furthermore, heuristic labeling methods, such as distant supervision (Craven and Kumlien 1999b) or data programming (Ehrenberg et al. 2016), are only applicable when a good knowledge base or pretrained predictor is available.

To address these problems, which are ubiquitous in real-world supervised machine learning (Piskorski and Yangarber 2013; Patterson et al. 2016), Attenberg *et al.* (2010) propose *guided learning*, a method for obtaining labeled training examples that uses generation queries, or tasks that ask crowd workers to *find* or *generate*, as opposed to just *label*, training examples. Because of the increased human effort, guided learning is more expensive per example. However, Attenberg *et al.* show that in highly-skewed domains, the added cost translates to training sets with more balanced class frequencies and thus more effective initial learning. Guided learning can quickly obtain minority-class examples, while other strategies like active learning flounder, looking for needles in a haystack.

Of course, guided learning isn't always appropriate and shouldn't be used indiscriminately. In balanced domains where examples of all classes are readily available, even asking workers to label randomly-selected examples will likely be cheaper than any generation strategy. And even in high-skew domains, Attenberg *et al.* show that after the classifier has been bootstrapped by guided learning, simply switching from generation to crowd-labeling of actively-selected examples can improve performance.

So how *should* a learner dynamically switch between generation, labeling, or other kinds of queries when gathering training examples? We observe that the key differences between these various queries are their costs and the distributions of data that they obtain; ultimately, the best strategy at any given time will differ depending on a number of factors, including the domain's class skew and the progress in training. Therefore, for optimal learning, solving the following meta-active learning problem is crucial: given (1) a set of *example-acquisition primitives* (EAPs), *i.e.*, various classes of queries for obtaining labeled training examples (*e.g.*, annotate a random example; generate a minority-class example), (2) a classifier previously trained on N examples (where N could be 0), (3) a labeled training set (possibly empty), (4) an unlabeled corpus, and (5) a budget, which EAP should be used next to obtain another labeled example in order to maximize performance of the classifier at the end of training? In our exploration of this problem, we make the following contributions:

- We propose a novel example acquisition primitive (EAP), enumerate five existing EAPs, and evaluate their effectiveness in training classifiers.
- We present a novel, online algorithm, called `MB-CB`, that adapts multi-armed bandit methods (Auer, Cesa-Bianchi, and Fischer 2002) to dynamically choose EAPs based

on evolving estimates of how cheaply they can obtain minority-class examples.

- We perform experiments with both synthetic and real data comparing the behavior of various control algorithms in multiple skew settings; these show that our bandit-based algorithm can yield up to a 14.3 point gain in F1 AUC, compared to the state-of-the-art baseline.

In the rest of this paper, we assume the binary classification setting and that the positive class is the minority class.

Example-Acquisition Primitives

We begin by listing existing EAPs, example-acquisition primitives (also known as query types (Settles 2012)), and proposing a novel one (LABEL-PREDPOS). We categorize the primitives into three types: labeling primitives, generation primitives, and machine primitives. Labeling and generation primitives rely on crowd annotators, while machine primitives can be executed without any human involvement.

Labeling primitives refer to strategies for choosing examples for labeling. Any active learning algorithm is a labeling primitive. These are usually cheap and simple, but as we have discussed earlier, ineffective in high skew domains. Generation primitives ask the crowd to generate or find examples; they are more costly per example, but reliably produce examples of any class, albeit often from a different distribution than desired. Machine primitives are used to create heuristically-labeled training data with no human involvement; thus, they are free, but can be noisy.

Labeling Primitives

- LABEL-RANDOM samples a *random* unlabeled example for labeling by a crowd worker; this is the traditional (non active) way that researchers have gathered i.i.d. data for supervised learning.
- LABEL-ACTIVE asks a crowd worker to label an example from the unlabeled corpus, selected using an active learning technique (*e.g.*, uncertainty sampling (Lewis and Catlett 1994)). Different active learning schemes are considered separate primitives for the downstream decision algorithm.
- LABEL-PREDPOS asks a worker to label an example that the current classifier predicts to be positive. We expect this novel primitive to be helpful in high-skew domains, as it may find many positive examples cheaply. It may also improve precision by correcting false positives.

Generation Primitives

- GENERATE-POSITIVE tasks a crowd worker with generating/finding a positive example (Attenberg and Provost 2010); this primitive is the quintessential guided learning strategy, and should improve recall. Note that generation of examples can be more difficult in some domains than in others (*e.g.*, vision vs. natural language processing).
- GENERATE-NEGATE tasks a worker with minimally modifying a positive example to turn it into a negative example. By creating "near-miss" negatives, we expect this primitive may will allow a classifier to quickly identify important features.

Machine Primitives

- ADD-RANDOM-NEG picks a random unlabeled example and inserts it into the training set as a negative example. This primitive, commonly used in the context of distant supervision (Craven and Kumlien 1999a; Mintz et al. 2009), does not require any crowd work. In domains with high class skew, it provides relatively clean negative examples. However, in more balanced domains, many positive examples may be inserted erroneously as negative examples.

Given an unlabeled corpus with an unknown class skew, our goal is to sequence these EAPs for efficient and cost-effective training. We describe our control algorithm next.

EAP Controller for Training Classifiers

Our decision making algorithm, which we call MB−CB (Make Balanced – Cost Bound), selects the next primitive to enhance the training set. In order to be robust to high skews, it pays special attention to positive class examples. It has two main parts. The "Cost Bound" part selects EAPs based on cost analysis for obtaining minority class (positive) examples, and the "Make Balanced" part is a heuristic to artificially make the training set balanced (if needed). We first describe the intuition behind the "Cost Bound" part.

We observe that a necessary condition for an effective EAP is that it should obtain positive examples cost-effectively. Labeling primitives work well in balanced settings because they are cheap and positive examples are common. In contrast, generation primitives are expensive, but may be cost effective in high-skew domains, since they are guaranteed to produce an example with the desired label. For example, suppose LABEL-RANDOM costs \$0.03 per example and GENERATE-POSITIVE costs \$0.15 per example; if fewer than 2% of the examples in the unlabeled corpus are positive, then GENERATE-POSITIVE will produce ten times as many positive examples per dollar. LABEL-ACTIVE will be at a similar disadvantage, at least until the classifier is partially trained.

The "Cost Bound" part of MB−CB operationalizes these insights. For every EAP, it computes the expected cost of obtaining a single positive example, and then chooses the cheapest primitive. Unfortunately, the expected cost of obtaining a positive from a labeling primitive is unknown and must be learned.

MB−CB learns the expected costs by executing primitives, which results in an exploration-exploitation setting. We model the problem using a multi-armed bandit, where the arms correspond to EAPs, and the reward of each arm is the negative expected cost of obtaining a single positive example from that EAP. Any control algorithm that tries to solve this problem must make a tradeoff between exploiting the knowledge it currently has (by executing the primitive it believes is cheapest), and exploring to update the model (by executing primitives in order to learn more about their non-stationary costs).

Algorithm 1 MB−CB

Input: EAPs $\mathcal{V}$, budget b, exploration constant c_e, desired skew (the desired # negatives per positive in training set) r, batch size k
$costSoFar = 0$;
$p_c = \{\}$ //Track estimated cost of positive per primitive
$p_\alpha = \{\}$ // Track # positives obtained per primitive
$p_\beta = \{\}$ // Track # negatives obtained per primitive
$p_n = \{\}$ //Track # times each primitive is called
for $v \in \mathcal{V}$ **do**
  $p_c[v] = p_n[v] = p_\alpha[v] = p_\beta[v] = 0$
end for
while $costSoFar < b$ **do**
  $bestAction = None$, $bestCost = \infty$

  /* For every primitive, compute the cost of a single positive based on historical costs and a UCB exploration term. */
  for $v \in \mathcal{V}$ **do**
    $cost = p_c[v] - \sqrt{c_e \frac{\log \sum_{v \in \mathcal{V}} p_n[v]}{p_n[v]}}$
    if $cost < bestCost$ **then**
      $bestAction = v$
      $bestCost = cost$
    end if
  end for
  Execute $bestAction$ k times, tracking $numPos$ and $numNeg$.
  Insert all $numPos$ positive examples into training set.

  /* Balance the training set by discarding or adding negative examples */
  if $isGenerationPrimitive(bestAction)$ **then**
    if $numNeg < r \cdot numPos$ **then**
      Insert $(r \cdot numPos) - numNeg$ randomly selected examples, labeled negative, into training set
    end if
  else if $isLabelingPrimitive(bestAction)$ **then**
    Insert at most $r \cdot numPos$ of the obtained negative examples into training set.
  end if

  /* Update the historical data for the chosen primitive */
  $p_\alpha[bestAction] = p_\alpha[bestAction] + numPos$
  $p_\beta[bestAction] = p_\beta[bestAction] + numNeg$
  $expectedNumPos = (numPos + numNeg) \cdot \frac{p_\alpha[bestAction]}{p_\alpha[bestAction] + p_\beta[bestAction]}$
  $p_c[bestAction] = \frac{bestAction.cost}{expectedNumPos}$
  $p_n[bestAction] = p_n[bestAction] + 1$
  $costSoFar = costSoFar + bestAction.cost$
end while

MB−CB manages this tradeoff by adapting the UCB algorithm (Auer, Cesa-Bianchi, and Fischer 2002) from the multi-armed bandit literature. (We also implement a similar algorithm using Thompson sampling (Thompson 1933), but we omit the results because of space limitations and the performance is very similar to MB−CB.) It maintains a lower bound on the cost of a single positive example for every primitive. Each lower bound is computed using an exploitation term (determined using the history of costs from the respective primitive) and an exploration term (determined based on the number of times the primitive has been executed). As each primitive is executed, its corresponding exploration bonus decreases. An exploration constant c_e determines the relative value of exploration and exploitation.

At each timestep, MB−CB selects the EAP with the lowest bound and executes a batch of k. This produces an observation about the cost of positive examples, which MB−CB uses to update the lower bound for that primitive. We note that the costs of primitives can be non-stationary, since they depend on the classifier's evolving precision. We tried modeling the problem using non-stationary bandits (Garivier and Moulines 2011; Cortes et al. 2017), but did not obtain significant improvements over MB−CB's simpler approach.

Of course positive examples are only part of the story, and MB−CB needs to ensure that it adds enough (but not too many!) negative examples as well. The "Make-Balanced" part of MB−CB enforces the desired skew (an input to the algorithm) by either discarding excess negatives or inserting additional, randomly-selected examples labeled as negative, as needed. Artificially bounding the training set skew by undersampling negatives or oversampling positives is a common practice in domains with high class imbalance (e.g. (Weiss and Provost 2003; Zhu and Hovy 2007)). Algorithm 1 shows the pseudocode for MB−CB.

Experiments

We now present a series of experiments with both real and synthetic data to answer three questions. The first question explores the relative effectiveness of the various EAPs for obtaining negative examples, the second question quantifies the value of our novel EAP (LABEL-PREDPOS), and the last question investigates the effectiveness of MB−CB at selecting EAPs:

1. How cost-effective is generating near-miss negative examples (GENERATE-NEGATE) compared to other ways of generating negative examples like random labeling (LABEL-RANDOM) or inserting random examples as negative (ADD-RANDOM-NEG)?
2. In a high-skew domain, is it better to request labels for likely-positive examples (LABEL-PREDPOS) or simply use uncertainty sampling (LABEL-ACTIVE)?
3. Overall, how does MB−CB compare to baselines and state-of-the-art guided learning algorithms in different domains and across varying skews?

Data Sets

LD and Modified LD: To answer our first experimental question, we consider the task of relation extraction, which involves determining whether a natural language sentence expresses a given relation between two given entities. We use two relation extraction datasets: one from Liu *et al.* (2016), which we denote LD, and an extension, which we crowdsource ourselves, Modified LD.

LD contains examples of five relations, with gold labels inferred from labels provided by crowdsourced workers. In particular, it contains 471 positive and 17,632 negative examples of "*Born in*," 1,375 positive and 16,635 negative examples of "*Died in*," 1,339 positive and 16,136 negative examples of "*Traveled to*," 1,175 positive and 14,231 negative examples of "*Lived in*," and 1,203 positive and 16,230 negative examples of "*Nationality*."

Modified LD enhances LD via crowdsourcing with Amazon Mechanical Turk. We provide workers with a relation and a positive example from LD, and ask them to minimally modify it to turn it into a negative example for that relation. For example, a good submission for the relation "*Died in*" and the sentence "He died yesterday in Prague" might be "He did not die yesterday", whereas an incorrect submission for the same relation and sentence might be "He died the day before yesterday in Prague", because the sentence still expresses that someone died somewhere. We run this task once for each of the positive examples in LD to obtain an equally-sized set of negative examples.

In an effort to increase the diversity of examples that workers submit, the task also provides a list of "taboo" words (Hasbro 2000; von Ahn and Dabbish 2004) that workers are barred from using in the sentences they submit. A word becomes "taboo" if the number of times it has been used has exceeded a threshold. We use a threshold of 20. The taboo list is computed by using the words that appear in the modified sentence but not in the original sentence (excluding stop words).

News Aggregator Data Sets: To answer our second and third experimental questions, we use two topic modeling datasets, which we denote as NADS (News Aggregator Data Set) and NADS-Generate. We use different (20x larger) datasets for the last two questions because these two questions involve the testing of intelligent algorithms that will almost always take different sequences of actions, which means that they require many more examples to sample from. Moreover, the scale of this dataset is also amenable to high skew experiments (such as 1:1000), which was not feasible using LD.

NADS is a dataset from UCI Machine Learning Repository (Bache and Lichman 2013), and consists of 422,937 news headlines that are labeled as one of four possible topics. 152,746 are labeled as "Entertainment," 108,465 are labeled as "Science and Technology," 115,920 are labeled as "Business," and 45,615 are labeled as "Health." We construct NADS-Generate by asking crowdsourced workers to find examples of news headlines of the appropriate topic (*e.g.*, Business) on the web. NADS-Generate contains 1,000 generated headlines for each topic. We note that workers are free to find headlines from any source, and hence, this generated distribution will likely be different from the distribution in NADS.

Experimental Setup

While we make use of pre-annotated corpora, we set the cost of all labeling EAPs to be \$0.03; this wage is consistent with prior work on crowd-labeling, *e.g.* (Liu et al. 2016). After preliminary experimentation, we set the cost of GENERATE-NEGATE to be \$0.10, and GENERATE-POSITIVE to be \$0.15, in order to produce an effective hourly wage equivalent to that paid for labeling.

In addition, for adaptive algorithms, instead of making a new decision at every timestep, we batch execute the next primitive 50 times ($k = 50$) to reduce computational costs. Thus, each GENERATE-POSITIVE costs \$7.50. We set the exploration constant $c_e = 1.0$ for `MB-CB`, thereby equally balancing between exploration and exploitation terms.

In all experiments, we vary the skew in the original unlabeled corpus artificially to understand each algorithm's behavior for datasets of varying class imbalance. For each skew s (i.e., unlabeled corpus has s negative examples for each positive one) and each EAP-choosing strategy, we train the target classifier at multiple cost points and compute the corresponding F1 scores. We then calculate the area under the F1-cost curve (cost-sensitive learning curve) to compute F1 AUC. We repeat this training many times and report the mean value for each skew.

Finally, we note that skew, 1:s, in the unlabeled corpus should not be confused with skew in the training set, 1:r. Typically, s may have a very high value, but r will be small (*e.g.*, 1 to 3) — achieved by under-sampling negatives.

The Value of Generating 'Near-Miss' Negatives

We first investigate whether or not the GENERATE-NEGATE EAP is more cost-effective than other primitives for generating negatives, such as ADD-RANDOM-NEG and LABEL-RANDOM. Specifically, we compare three different strategies: `Gen+Modify-` simulates GENERATE-POSITIVE by sampling positive examples from LD and uses GENERATE-NEGATE to obtain corresponding negative examples (from Modified LD). `Gen+Rand-` uses the same source for positive examples, but instead of using GENERATE-NEGATE to create negatives, it uses ADD-RANDOM-NEG, which randomly samples examples from LD. We compare these against a simple LABEL-RANDOM baseline, which generates both positives and negatives via random labeling over LD.

For every relation and every skew $s \in \{1, 9, 99\}$, we set a budget of $b = \frac{0.15\kappa}{2}$, where κ is the number of positive examples for the chosen relation (*e.g.*, $\kappa = 471$ for the "*Born in*" relation). We set the budget in this way in order to ensure we do not run out of examples during experimentation. Note that for both `Gen+Modify-` and `Gen+Rand-` we set $r = 1$, because of the limited number of modified negatives in the dataset. For fairness, we artificially maintain $r = 1$ in LABEL-RANDOM by discarding excess negative examples.

First, notice that all these strategies have very different cost profiles. `Gen+Modify-` is most expensive, since it uses two generate actions that cost 15 and 10 cents each. In contrast, `Gen+Rand-`'s way of generating negatives is free (but can be noisy at low skews); thus it spends *all* its budget generating positives. Finally, LABEL-RANDOM does not utilize any expensive generate actions, but must discard negatives to maintain training skew. Because of this, at each cost point, their training datasizes will be different, with `Gen+Modify-` being the smallest at low skews,

and LABEL-RANDOM being the largest. At high skews, LABEL-RANDOM will be small, since excess negatives are discarded, and Gen+Rand- will always generate a larger training set than Gen+Modify-.

We train logistic regression classifiers using the training sets constructed by the three strategies, using standard NLP features from the IE literature (Mintz et al. 2009). We evaluate using the test set from Liu *et al.* (2016). Figure 1 shows the results for the five relations. The error bars represent 95% confidence intervals.

We find that LABEL-RANDOM vastly outperforms the other strategies at low skews. Gen+Rand- is especially poor in this context, because ADD-RANDOM-NEG puts many false negatives into the training set. We also observe that at high skew, Gen+Rand- outperforms the other strategies. Presumably, it beats Gen+Modify- because GENERATE-NEGATE is costly, leading to a 40% smaller training set. Disappointingly, there doesn't appear to be a setting where the GENERATE-NEGATE EAP is helpful, as Gen+Modify- is dominated for every value of skew.

Overall, we conclude that although GENERATE-NEGATE can be more cost-effective than ADD-RANDOM-NEG in low skew settings, ultimately it is unlikely to be the best EAP to use in *any* skew setting. Thus, we do not continue to further investigate the GENERATE-NEGATE EAP.

Uncertainty Sampling *vs.* Predicted Positives

We now study the relative value of LABEL-PREDPOS and LABEL-ACTIVE. Since finding positive examples is crucial in high-skew domains, we conjecture that examples thought by the current classifier to be positive should be especially promising. Having the workers label these points should generate more true positive training examples than a standard active learning algorithm like uncertainty sampling (Lewis and Catlett 1994). This experiment aims to test this hypothesis.

To compare LABEL-PREDPOS and LABEL-ACTIVE, we implement two versions of MB-CB, which select between two EAPs each. Both versions use GENERATE-POSITIVE as one of the EAPs, but differ on the second. MB-CB(Pos) uses LABEL-PREDPOS, whereas MB-CB(Active) uses uncertainty sampling as its second EAP.

Recall that MB-CB artificially bounds the class ratio to 1:r. In contrast to the previous experiment, we set $r = 3$ because the larger dataset that we use (NADS) allows us to utilize a training set with slightly more minority examples than majority examples, which tends to work well in skewed domains (Weiss and Provost 2003). Any time the algorithms pick GENERATE-POSITIVE, they automatically execute three ADD-RANDOM-NEG actions for each generated positive. Any time the algorithms pick LABEL-PREDPOS or LABEL-ACTIVE, if n is the number of obtained positive examples, then the strategy will keep all n positive examples in the training set, but keep at most $3n$ of the obtained negatives and discard the rest. We make an exception if $n = 0$. In this case, we pretend $n = 1$ and keep 3 negative examples so that we are always adding some data with each EAP execution and avoid infinite loops.

To compare the two algorithms, we train logistic regression classifiers using a unigram bag of words model. We first set a topic to be the positive class (*e.g.*, "Health"). For each skew $s \in \{1, 9, 49, 99, 499, 999\}$, we run each algorithm 10 times using a budget of $100. For each run of an algorithm, we construct a new synthetic dataset from NADS in the following manner: we first construct a *generation set* by sampling 2000 positive examples from NADS. Anytime an algorithm executes the GENERATE-POSITIVE action, we randomly sample examples from this generation set. Then, we construct a test set by sampling 100 positive examples and $100 \times s$ negative examples. Finally, we construct an unlabeled corpus by sampling from the remaining examples as many positive examples as possible while maintaining the desired skew s. When an algorithm executes labeling or machine primitives, we sample from this set. As before we plot the area under the F1-cost curve, averaged over the 10 runs.

Figure 2(a) shows our results for the "Health" domain. To our surprise, we see that MB-CB(Active) dominates MB-CB(Pos). This result is unexpected, because MB-CB(Pos) uses a primitive designed specifically to locate positive examples, and yet it loses to uncertainty sampling, even at high skews.

To find out why, we investigate the behavior of the two algorithms by comparing how often they execute generation primitives versus labeling primitives. Figure 2(b) shows the case for extreme skew, $s = 999$. Note that both algorithms start with 100% labeling actions (because they are cheaper), but both become disenchanted by low yield and switch to generation. After the classifiers have been trained with some generated positives (increasing recall), they switch back to labeling. But MB-CB(Active) does significantly more labeling, which means that it must be finding more positives during labeling.

Figure 2(c) confirms the analysis, showing that MB-CB(Active) actually obtains many more positive examples from labeling than does MB-CB(Pos). Only when the budget is nearly exhausted does MB-CB(Pos) catch up. We find similar results on the other three domains (graphs omitted for space). This suggests that classifiers are unable to distinguish between classes in the early stages of learning, because otherwise MB-CB(Pos) would be able to identify positive examples sooner.

Overall, we conclude that LABEL-PREDPOS's benefits over time-tested uncertainty sampling are unclear. By the time classifiers are more competent at identifying positive examples, explicitly finding such examples may be less impactful, because positive examples are most useful early on, when recall is low.

Performance of MB-CB

Having ruled out the LABEL-PREDPOS and GENERATE-NEGATE EAPs in the previous two experiments, we finalize our best MB-CB algorithm as MB-CB(Active), one which switches between two EAPs – GENERATE-POSITIVE and LABEL-ACTIVE, while using ADD-RANDOM-NEG to manage the class ratio of the generation primitive. (We also rule out LABEL-RANDOM as it is outperformed by LABEL-ACTIVE. We omit this result for lack of space.) Indeed, we

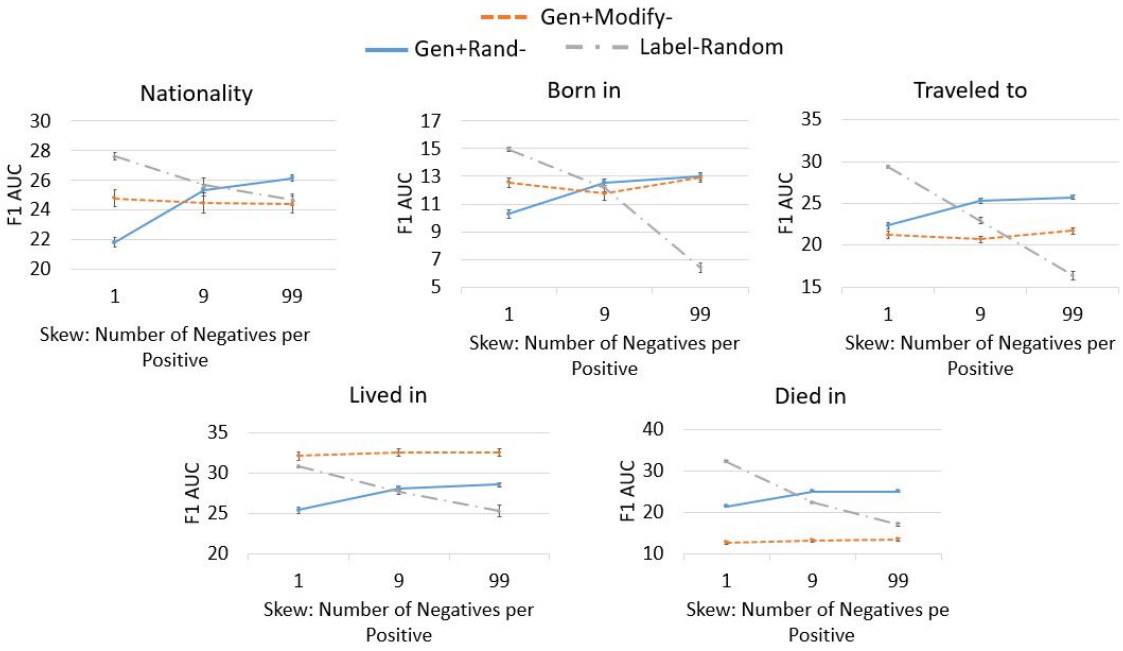

Figure 1: At low skews LABEL-RANDOM dominates, while at high skews `Gen+Rand-` (which combines GENERATE-POSITIVE and ADD-RANDOM-NEG) is best. We conclude that GENERATE-NEGATE is rarely useful because `Gen+Modify-` does not perform the best at any skew.

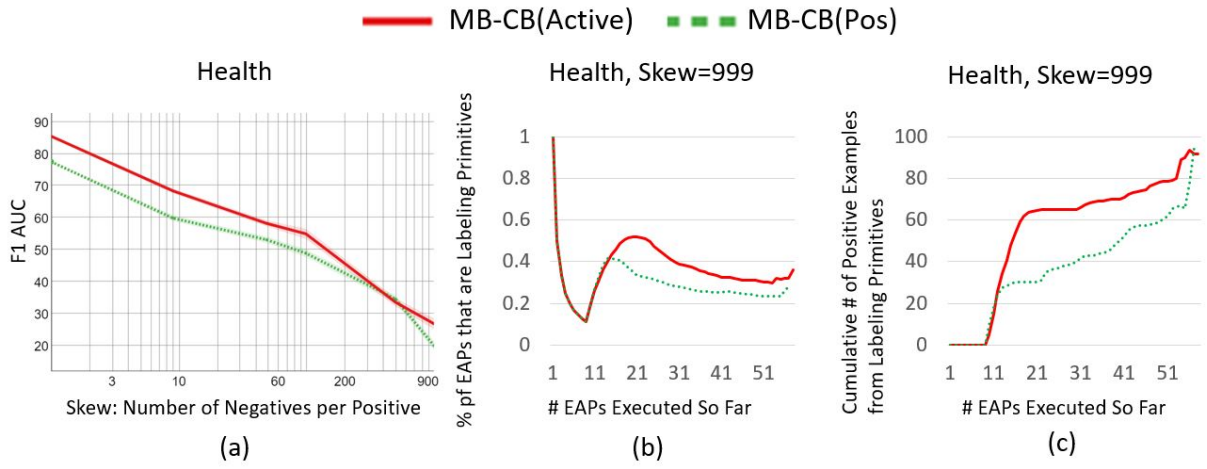

Figure 2: When training classifiers to identify "Health" headlines: (a) `MB-CB(Active)` outperforms `MB-CB(Pos)` in all skew settings; (b) when the skew is 999, `MB-CB(Active)` executes Labeling Primitives more often and Generation Primitives less often than `MB-CB(Pos)`; (c) when the skew is 999, `MB-CB(Active)` surprisingly obtains more positive examples from Labeling Primitives than does `MB-CB(Pos)`. We find similar results for the other three domains.

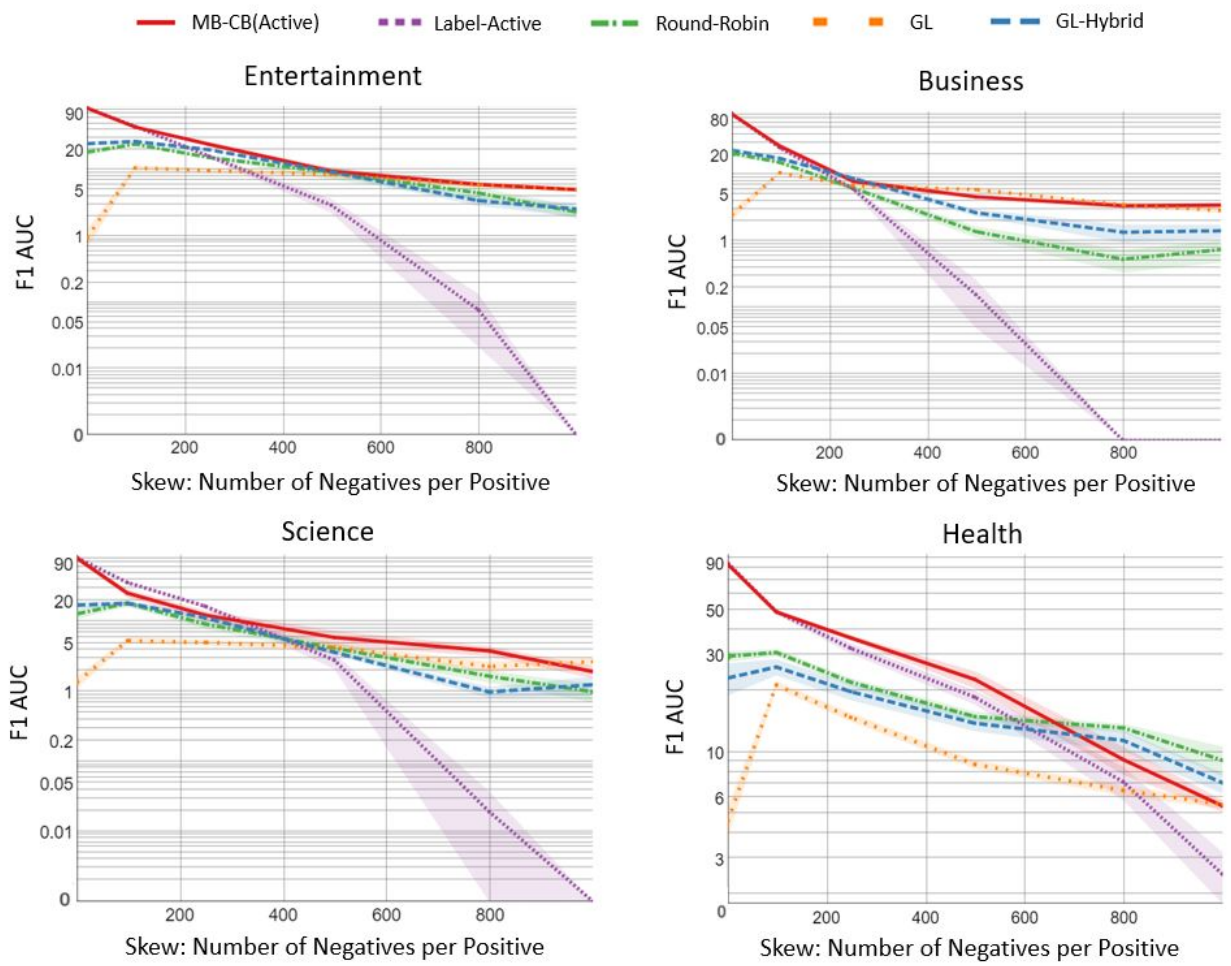

Figure 3: Comparison of `MB-CB(Active)`, `Round-Robin`, LABEL-ACTIVE, `GL`, and `GL-Hybrid` on 4 domains using real generated data. `MB-CB(Active)` trains better classifiers than the state-of-the-art baselines, `GL` and `GL-Hybrid`, across many skew settings.

believe that switching between labeling and generation is the critical technique for achieving robustness to skew. We now answer the third experimental question, "How does our algorithm for choosing EAPs, `MB-CB(Active)`, compare to other strategies?" For comparison, we choose two simple baselines, `Round-Robin` and LABEL-ACTIVE, and two state-of-the-art algorithms, `GL` and `GL-Hybrid` (Attenberg and Provost 2010).

Comparison Algorithms: `GL` (Guided Learning) tasks workers with generating examples at a specified class ratio. `GL` simply cycles between executing GENERATE-POSITIVE once and ADD-RANDOM-NEG 3 times in order to match the ratio that we use in `MB-CB(Active)`. `GL` does not execute LABEL-ACTIVE.

`Round-Robin` simply cycles between GENERATE-POSITIVE and LABEL-ACTIVE, where LABEL-ACTIVE only executes uncertainty sampling.

Finally, `GL-Hybrid` begins by executing `GL`. After every action, it estimates performance using cross-validation and constructs a learning curve. It then estimates future expected gain in performance by estimating the derivative of the learning curve at the last computed point. When the derivative drops below a threshold t, it switches to executing LABEL-ACTIVE and never executes `GL` again. We set $t = 0.00005$ as suggested by Attenberg *et al.* (2010).

Experimental Results With Real Generation Sets: We use the same dataset construction as the previous experiment to compare algorithms, with one exception. Instead of simulating generation by sampling positives from the NADS corpus, we use NADS-Generate (populated by AMT workers) as a *real* generation set. This setting is far more challenging, since algorithms may suffer losses from the distributional differences between the generated examples and the actual test examples.

Figure 3 shows the results for the "Entertainment," "Business," "Science", and "Health" domains. We use a log scale on the y-axis in order to more clearly show the differences. We first observe that, unsurprisingly, LABEL-ACTIVE per-

forms well in low-skew environments, but eventually is unable to learn anything at the highest skews. GL is a strong strategy only in high-skew domains; and Round-Robin achieves better results than GL at low-skew, but only outperforms LABEL-ACTIVE at high skew.

Next, we observe that GL-Hybrid does not clearly improve upon GL. The key weakness of GL-Hybrid lies in the difficulty of setting the threshold parameter. The estimations used to compute whether to switch to active learning can be wildly wrong, causing the algorithm to switch from guided learning to active learning either far too early or far too late if the threshold is not set correctly. For example, in a low-skew setting, GL-Hybrid may execute GENERATE-POSITIVE for an extremely long time if the performance of the classifier consistently rises.

Finally, we see that our algorithm, MB-CB(Active), is the most robust algorithm overall, and averages a 14.3 gain in F1 AUC over state-of-the-art GL-Hybrid across all skews and domains. This result underscores the importance of adaptive switching between the two primitives using a learning-based approach.

Code and data to reproduce our experiments can be found at: https://github.com/polarcoconut/thesis-skew.

Discussion

MB-CB is a first step towards an intelligent active learning approach that is robust to skew. Many different kinds of EAPs could be added into its repertoire, like distant supervision (Craven and Kumlien 1999a; Angeli et al. 2014) or feature labeling (Patterson and Hays 2016).

However, MB-CB has an important technical weakness. While it is very good at learning about the cost-effectiveness of an EAP for finding positive examples (which is especially valuable for initial training at high skews), it does not differentiate between the *qualities* of different positive (or negative) examples. For instance, its selection mechanism cannot prefer LABEL-ACTIVE over GENERATE-POSITIVE (which may produce positive examples from a completely different distribution), except when active learning is generating positive examples more cheaply. An alternative approach may be to model the problem using budget-limited multi-armed bandits (Tran-Thanh et al. 2010; 2012; Ding et al. 2013) that chooses the next EAP based on the expected gain in precision, recall or F1, though it may be challenging to robustly predict the expected gain.

Furthermore, we note that MB-CB's selection rule does not explicitly attempt to gather negative examples, which is rectified by the make-balanced heuristic. We hope that a future modification will make adding negatives an explicit part of the algorithm's selection policy, so that the number of negatives may also be chosen intelligently.

Related Work

High-Skew Active Learning

Various methods for active learning in high skew environments have been proposed, such as those based on nearest neighbors (He and Carbonell 2007; Doersch et al. 2012; Patterson et al. 2016), query by committee (Tomanek and Hahn 2009), and uncertainty sampling (Vijayanarasimhan and Grauman 2011). Other approaches use multiple classifiers to choose the next examples to label (Wallace et al. 2010; Li et al. 2012). Extensions of active learning algorithms for high skew scenarios include allowing the annotators to perform keyword search to generate examples (Vijayanarasimhan and Grauman 2011), labeling attributes instead of data points (Patterson and Hays 2016), and guided learning, which enables the annotators to generate training data points (Attenberg and Provost 2010). Our work builds upon guided learning. However, all these approaches are targeted at developing a *single* active learning strategy, whereas our work adaptively chooses among *various* strategies to achieve more efficient training. In some sense, our work can be understood as a *meta*-active learning approach.

Training algorithms often artificially reduce class imbalance by oversampling minority class examples (Zhu and Hovy 2007) or choosing skew-dependent misclassification costs (Bloodgood and Vijay-Shanker 2009). We choose the former strategy in our experiments.

Guided Learning and Example Generation

Attenberg *et al.* (2010) propose *guided learning* in which annotators generate or find positive examples. Guided learning has been useful in creating a variety of NLP datasets, including text classification datasets over tweets (Sadilek et al. 2013) and advertisements (Sculley et al. 2011); and paraphrase data for training dialog systems (Wang et al. 2012) and semantic parsers (Wang, Berant, and Liang 2015).

The original guided learning paper shows that in high-skew settings, guided learning is more effective than uncertainty sampling, and guided learning followed by uncertainty sampling is more effective than either of them in isolation. Our work builds upon this sequential hybrid, but allows the algorithm to dynamically choose between the two (and other EAPs). Our experiments show that this added power of interleaving performs substantially better than the user-defined switch point of the original paper.

Generation of near-miss examples has been used for training object detectors (Gurevich, Markovitch, and Rivlin 2006) and visual QA systems (Zhang et al. 2016).

Heuristics for Identifying Positive Examples

An alternative for generating balanced training sets in high-skew domains is to use a heuristic to noisily label examples. For example, distant supervision is frequently used for information extraction (Craven and Kumlien 1999b; Wu and Weld 2007). It labels as positive any sentence that contains two entities that are known to have a relation between them (in an external knowledge base). Unfortunately, the assumption that the target concept is in some external knowledge base is, in many cases, unrealistic.

Data programming (Ehrenberg et al. 2016) is a paradigm in which humans design domain-specific rules that can be programmatically used to label examples (e.g., (Hoffmann, Zettlemoyer, and Weld 2015)). Whenever feasible, data programming is a strategy for obtaining examples, and can be considered another EAP.

Conclusion

Active learning systems can use many different query types to acquire labeled training data; we present a novel solution aimed at maximizing classifier performance for a given annotation budget. After listing several existing EAPs and proposing a new one, we introduce a bandit algorithm for the problem of selecting EAPs. MB−CB works by computing the expected cost of obtaining a single positive example from each method and then picking the cheapest EAP. Because these costs can only be learned through execution of the EAPs, MB−CB adapts from the multi-armed bandit literature to make the tradeoff between exploiting the EAP it believes to be cheapest and exploring the costs of other EAPs.

We perform experiments on real and synthetic datasets to explore the behavior of the basic primitives and our control algorithm. First, we show that asking the crowd to generate 'near-miss' negative examples is not cost-effective compared to either traditional labeling (at low skew) or blindly labeling a random example to be negative (at high skew). Second, we demonstrate that, surprisingly, trying to label predicted positive examples actually results in finding *fewer* positive examples than active learning during the early stages of training. As a result, using LABEL-ACTIVE creates a better classifier. Finally, we show that MB−CB has the ability to adapt to domains of varying skew and outperforms state-of-the-art baselines, yielding a 14.3 point gain on average in F1 AUC over 24 environments (6 domains × 4 skews) compared to Attenberg *et al.*'s (2010) best algorithm, GL−Hybrid.

Acknowledgements

We are very grateful to Josh Attenberg who dug up an old laptop to provide us with parameters used for GL−Hybrid. We thank James Ferguson, Yifei Song, Mandar Joshi, Jonathan Bragg, and the anonymous reviewers for helpful comments and discussions. This work was supported by NSF grant IIS-1420667, ONR grant N00014-12-1-0211, the WRF/Cable Professorship, support from Google, a Bloomberg award, an IBM SUR award, a Microsoft Azure sponsorship, and a Visvesvaraya faculty award by the Government of India to the second author.

References

Angeli, G.; Tibshirani, J.; Wu, J. Y.; and Manning, C. D. 2014. Combining distant and partial supervision for relation extraction. In *EMNLP*.

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

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine Learning* 47(2):235–256.

Bache, K., and Lichman, M. 2013. UCI machine learning repository.

Bloodgood, M., and Vijay-Shanker, K. 2009. Taking into account the differences between actively and passively acquired data: The case of active learning with support vector machines for imbalanced datasets. In *NAACL*.

Chawla, N. V.; Bowyer, K. W.; Hall, L. O.; and Kegelmeyer, W. P. 2002. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research* 16:321–357.

Cortes, C.; DeSalvo, G.; Kuznetsov, V.; Mohri, M.; and Yang, S. 2017. Discrepancy-Based Algorithms for Non-Stationary Rested Bandits. *ArXiv e-prints*.

Craven, M., and Kumlien, J. 1999a. Constructing biological knowledge bases by extracting information from text sources. In Lengauer, T.; Schneider, R.; Bork, P.; Brutlag, D. L.; Glasgow, J. I.; Mewes, H.-W.; and Zimmer, R., eds., *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology, August 6-10, 1999, Heidelberg, Germany*, 77–86. AAAI.

Craven, M., and Kumlien, J. 1999b. Constructing biological knowledge bases by extracting information from text sources. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*.

Ding, W.; Qin, T.; Zhang, X.-D.; and Liu, T.-Y. 2013. Multi-armed bandit with budget constraint and variable costs. In *AAAI*.

Doersch, C.; Singh, S.; Gupta, A.; Sivic, J.; and Efros, A. A. 2012. What makes paris look like paris? *ACM Transactions on Graphics (SIGGRAPH)* 31(4):101:1–101:9.

Ehrenberg, H. R.; Shin, J.; Ratner, A. J.; Fries, J. A.; and Ré, C. 2016. Data programming with DDLite: Putting humans in a different part of the loop. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*, HILDA '16, 13:1–13:6. New York, NY, USA: ACM.

Garivier, A., and Moulines, E. 2011. On upper-confidence bound policies for switching bandit problems. In Kivinen, J.; Szepesvári, C.; Ukkonen, E.; and Zeugmann, T., eds., *Algorithmic Learning Theory*, 174–188. Berlin, Heidelberg: Springer Berlin Heidelberg.

Gurevich, N.; Markovitch, S.; and Rivlin, E. 2006. Active learning with near misses. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, 362. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

Hasbro. 2000. Taboo™ hasbro.com/common/instruct/taboo(2000).pdf.

He, J., and Carbonell, J. 2007. Nearest-neighbor-based active learning for rarer category detection. In *NIPS*.

Hoffmann, R.; Zettlemoyer, L. S.; and Weld, D. S. 2015. Extreme extraction: Only one hour per relation. *CoRR* abs/1506.06418.

Lewis, D. D., and Catlett, J. 1994. Heterogeneous uncertainty sampling for supervised learning. In *ICML*.

Li, S.; Ju, S.; Zhou, G.; and Li, X. 2012. Active learning for imbalanced sentiment classification. In *EMNLP-CoNLL*.

Liu, A.; Soderland, S.; Bragg, J.; Lin, C. H.; Ling, X.; and Weld, D. S. 2016. Effective crowd annotation for relation extraction. In *NAACL*.

Mintz, M.; Bills, S.; Snow, R.; and Jurafsky, D. 2009. Distant supervision for relation extraction without labeled data. In *ACL*.

Patterson, G., and Hays, J. 2016. Coco attributes: Attributes for people, animals, and objects. *European Conference on Computer Vision*.

Patterson, G.; Horn, G. V.; Belongie, S.; Perona, P.; and Hays, J. 2016. Tropel: Crowdsourcing detectors with minimal training. In *HCOMP*.

Piskorski, J., and Yangarber, R. 2013. *Information Extraction: Past, Present and Future*. Berlin, Heidelberg: Springer Berlin Heidelberg. 23–49.

Sadilek, A.; Brennan, S.; Kautz, H.; and Silenzio, V. 2013. nemesis: Which restaurants should you avoid today? In *HCOMP*.

Sculley, D.; Otey, M. E.; Pohl, M.; Spitznagel, B.; Hainsworth, J.; and Zhou, Y. 2011. Detecting adversarial advertisements in the wild. In *KDD*.

Settles, B. 2012. *Active Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Thompson, W. R. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 25(3/4):285–294.

Tomanek, K., and Hahn, U. 2009. Reducing class imbalance during active learning for named entity annotation. In *K-CAP*.

Tran-Thanh, L.; Chapman, A.; Luna, J. E. M. D. C. F.; Rogers, A.; and Jennings, N. R. 2010. Epsilon-first policies for budget-limited multi-armed bandits. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 1211–1216. Event Dates: 11 - 15 July, 2010.

Tran-Thanh, L.; Chapman, A.; Rogers, A.; and Jennings, N. R. 2012. Knapsack based optimal policies for budget-limited multi-armed bandits. In *Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI-12)*, 1134–1140.

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

von Ahn, L., and Dabbish, L. 2004. Labeling images with a computer game. In *CHI*.

Wallace, B. C.; Small, K.; Brodley, C. E.; and Trikalinos, T. A. 2010. Active learning for biomedical citation screening. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, 173–182. New York, NY, USA: ACM.

Wang, Y.; Berant, J.; and Liang, P. 2015. Building a semantic parser overnight. In *ACL*.

Wang, W. Y.; Bohus, D.; Kamar, E.; and Horvitz, E. 2012. Crowdsourcing the acquisition of natural language corpora: Methods and observations. In *Spoken Language Technology Workshop*.

Weiss, G. M., and Provost, F. 2003. Learning when training data are costly: The effect of class distribution on tree induction. *Journal of Artificial Intelligence Research* 19:315–354.

Wu, F., and Weld, D. 2007. Autonomously semantifying Wikipedia. In *Proceedings of the ACM Sixteenth Conference on Information and Knowledge Management (CIKM-07)*.

Zhang, P.; Goyal, Y.; Summers-Stay, D.; Batra, D.; and Parikh, D. 2016. Yin and yang: Balancing and answering binary visual questions. In *CVPR*.

Zhu, J., and Hovy, E. 2007. Active learning for word sense disambiguation with methods for addressing the class imbalance problem. In *EMNLP*.