Data Min Knowl Disc (2013) 27:372–395
DOI 10.1007/s10618-013-0329-7

Growing a list

Benjamin Letham · Cynthia Rudin ·
Katherine A. Heller

Received: 16 December 2012 / Accepted: 19 June 2013 / Published online: 16 July 2013

Abstract It is easy to find expert knowledge on the Internet on almost any topic, but obtaining a complete overview of a given topic is not always easy: information can be scattered across many sources and must be aggregated to be useful. We introduce a method for intelligently growing a list of relevant items, starting from a small seed of examples. Our algorithm takes advantage of the wisdom of the crowd, in the sense that there are many experts who post lists of things on the Internet. We use a collection of simple machine learning components to find these experts and aggregate their lists to produce a single complete and meaningful list. We use experiments with gold standards and open-ended experiments without gold standards to show that our method significantly outperforms the state of the art. Our method uses the ranking algorithm Bayesian Sets even when its underlying independence assumption is violated, and we provide a theoretical generalization bound to motivate its use.

Responsible editor: Hendrik Blockeel, Kristian Kersting, Siegfried Nijssen, Filip Zelezny.

Electronic supplementary material The online version of this article (doi:10.1007/s10618-013-0329-7) contains supplementary material, which is available to authorized users.

B. Letham (✉)
Operations Research Center, Massachusetts Institute of Technology, Cambridge, MA, USA
e-mail: bletham@mit.edu

C. Rudin
MIT Sloan School of Management, Massachusetts Institute of Technology, Cambridge, MA, USA
e-mail: rudin@mit.edu

K.A. Heller
Center for Cognitive Neuroscience, Statistical Science, Duke University, Durham, NC, USA
e-mail: kheller@gmail.com

Keywords Set completion · Ranking · Internet data mining · Collective intelligence

1 Introduction

We aim to use the collective intelligence of the world's experts to grow a list of useful information on any given topic. To do this, we aggregate knowledge from many experts' online guides in order to create a central, authoritative source list. We focus on the task of open-ended list aggregation, inspired by the collective intelligence problem of finding all planned events in a city. There are many online "experts" that list Boston events, such as Boston.com or Yelp, however these lists are incomplete. As an example of the difficulties caused by information fragmentation, traffic in parts of greater Boston can be particularly bad when there is a large public event such as a street festival or fundraising walk. Even though these events are planned well in advance, the lack of a central list of events makes it hard to avoid traffic jams, and the number of online sources makes it difficult to compile a complete list manually.

As the amount of information on the Internet continues to grow, it becomes increasingly important to be able to compile information automatically in a fairly complete way, for any given domain. The development of general methods that automatically aggregate this kind of collective knowledge is a vital area of current research, with the potential to positively impact the spread of useful information to users across the Internet.

Our contribution in this paper is a real system for growing lists of relevant items from a small "seed" of examples by aggregating information across many internet experts. We provide an objective evaluation of our method to show that it performs well on a wide variety of list growing tasks, and significantly outperforms existing methods. We provide some theoretical motivation by giving bounds for the Bayesian Sets method used within our algorithm. None of the components of our method are particularly complicated; the value of our work lies in combining these simple ingredients in the right way to solve a real problem.

There are two existing, publicly available methods for growing a list of items related to a user-specified seed. The first was introduced ten years ago on a large scale by Google Sets, which is accessible via Google Spreadsheet. The second is a more recent online system called Boo!Wa! (http://boowa.com), which is similar in concept to Google Sets. In our experiments, we found that Boo!Wa! is a substantial advance above Google Sets, and the algorithm introduced here is a similarly sized leap in technology above Boo!Wa!. In a set of 50 experiments shown in Sect. 4, the lower 25th percentile of our performance was better than the median performance of both Google Sets and Boo!Wa! in Precision@10 and average precision. More generally, our work builds on "search" and other work in information retrieval. Search engines locate documents containing relevant information, but to produce a list one would generally need to look through the webpages and aggregate the information manually. We build on the speed of search, but do the aggregation automatically and in a much more complete way than a single search.

2 Algorithm

Algorithm 1 gives an outline of the list growing algorithm, which we now discuss in detail.

Algorithm 1 Outline of the list growing algorithm

```
Input: A list of seed items
Output: A ranked list of new items related to the seed items
for as many iterations as desired do
  for each pair of seed items do
    Source discovery: Find all sites containing both items
    for each source site do
      List extraction: Find all items on the site represented similarly to the seed items
    end for
  end for
  for each discovered item do
    Feature space: Using a search with the item as the query, construct a binary feature vector of domains
    where the item is found
    Ranking: Score the item according to the seed using Bayesian Sets
  end for
  Implicit feedback: Add the highest-ranked non-seed item to the seed
end for
```

Source discovery: We begin by using the seed items to locate sites on the Internet that serve as expert sources for other relevant items. We use a combinatorial search strategy that relies on the assumption that a site containing at least two of the seed items likely contains other items of interest. Specifically, for every pair of seed items, we search for all websites that contain both of the items; this step takes advantage of the speed of "search."

In some cases, seed items may appear together in several contexts. For example, suppose one were to grow a list of Category 5 Atlantic hurricanes with "Hurricane Katrina" and "Hurricane Emily" in the seed. In addition to being Category 5 hurricanes, both of these hurricanes were in the 2005 hurricane season, and are found together on lists of 2005 Atlantic hurricanes. A search for sites containing both of these seed items would then recover both relevant sources on Category 5 hurricanes, as well as irrelevant sources on 2005 hurricanes. When more than two seed items are available, the context can be constrained by requiring source sites to contain all seed items, as opposed to just pairs. While this can potentially reduce the number of incorrect items, it could also miss relevant items that are found only on fragmented lists with little overlap. With our pairwise search strategy, our primary goal is complete coverage of relevant items, and we use the later ranking step to push the incorrect items to the bottom of the list. In Sect. 4.2 we explore experimentally how additional seed items constrain the context when ranking, and experimental results in Sect. 4.3 provide additional motivation for source discovery with pairs.

List extraction: The output of the combinatorial search is a collection of source sites, each of which contains at least two seed items. We then extract all of the new items from each of these sites. Here our strategy relies on the assumption that human experts organize information on the Internet using HTML tags. For each site found with the combinatorial search, we look for HTML tags around the seed items. We then find the largest set of HTML tags that are common to both seed items (for this site) and extract all items on the page that use the same HTML tags. In some situations, this strategy can result in noisy lists because it allows any HTML tags, including generic ones like `<b>` and `<a>`. An alternative strategy is to limit item extraction to list-specific HTML tags like `<li>`, and we explore this and related strategies experimentally in Sect. 4.2. As before, our goal at this step is complete coverage of relevant items, so we allow all HTML tags and use the later ranking step to ensure that the noise is pushed to the bottom of the list.

Feature space: At this point the algorithm has discovered a collection of lists, each from a different source. We now combine these lists so that the most relevant information is at the top of the final, merged list. To determine which of the discovered items are relevant, we construct a feature space in which to compare them to the seed items. Specifically, for each discovered item x, we construct a binary feature vector where each feature j corresponds to an internet domain (like boston.com or mit.edu), and $x_j = 1$ if item x can be found on internet domain j. This set of internet domains is found using a search engine with the item as the query.

The assumption behind this strategy is that related items should be found on a set of mainly overlapping domains, so we determine relevance by looking for items that cluster well with the seed items in the feature space. Constructing this feature space requires an additional search query for each discovered item. An alternative strategy that does not require additional search queries is to construct the feature space using only the source sites, that is, sites containing at least two seed items. This strategy, however, does not provide a way to distinguish between items that are found often in general, both with or without seed items, and items that are found often specifically with seed items. We compare these two approaches empirically in Sect. 4.2.

Ranking: The Bayesian Sets algorithm (Ghahramani and Heller 2005) ranks items according to the likelihood that they form a cluster with the seed, based on a probabilistic model for the feature space. Specifically, we suppose that each feature (in general, x_j) is a Bernoulli random variable with probability θ_j of success: $x_j \sim \text{Bern}(\theta_j)$. Following the typical Bayesian practice, we assign a Beta prior to the probability of success: $\theta_j \sim \text{Beta}(\alpha_j, \beta_j)$. Bayesian Sets assigns a score $f(x)$ to each item x by comparing the likelihood that x and the seed $S = \{x^1, \dots, x^m\}$ were generated by the same distribution to the likelihood they are independent:

$$f(x) := \log \frac{p(x, S)}{p(x)p(S)}. \tag{1}$$

Suppose there are N features: $x \in \{0, 1\}^N$. Because of the Bernoulli-Beta conjugacy, Ghahramani and Heller (2005) show that (1) has an analytical form under the assumption of independent features. However, the score given in Ghahramani and Heller (2005) can be arbitrarily large as m (the number of seed examples) increases. We prefer a normalized score because it leads to guarantees that the results are stable, or not too sensitive to any one seed item, as we show in Sect. 3. We use the following scoring function which differs from that in Ghahramani and Heller (2005) only by constant factors and normalization:

$$f_S(x) := \frac{1}{Z(m)} \sum_{j=1}^{N} x_j \log \frac{\alpha_j + \sum_{s=1}^{m} x_j^s}{\alpha_j} + (1 - x_j) \log \frac{\beta_j + m - \sum_{s=1}^{m} x_j^s}{\beta_j}, \tag{2}$$

where

$$Z(m) := N \log \left(\frac{\gamma_{\min} + m}{\gamma_{\min}} \right)$$

and $\gamma_{\min} := \min_j \min\{\alpha_j, \beta_j\}$ is the weakest prior hyperparameter. It is easy to show that $f_S(x) \in [0, 1]$, as we do in Lemma S1 in the supplement. Given the seed and the prior, (2) is linear in x, and can be formulated as a single matrix multiplication. When items are scored and then ranked using Bayesian Sets, the items that were most likely to have been generated by the same distribution as the seed items are put high on the list.

As is typically the case in Bayesian analysis, there are several options for selecting the prior hyperparameters α_j and β_j, including the non-informative prior $\alpha_j = \beta_j = 1$. Heller and Ghahramani (2006) recommend using the empirical distribution. Given n items to score $x^{(1)}, \ldots, x^{(n)}$, we let

$$\alpha_j = \kappa_1 \left(\frac{1}{n} \sum_{i=1}^{n} x_j^{(i)} \right), \beta_j = \kappa_2 \left(1 - \frac{1}{n} \sum_{i=1}^{n} x_j^{(i)} \right). \tag{3}$$

The first term in the sum in (2) corresponds to the amount of score obtained by x for the co-occurrence of feature j with the seed, and the second term corresponds to the amount of score obtained for the non-occurrence of feature j with the seed. When $\alpha_j = \beta_j$, the amount of score obtained when x_j and the seed both occur is equivalent to the amount of score obtained when x_j and the seed both do not occur. Increasing β_j relative to α_j gives higher emphasis to co-occurring features. This is useful when the feature vectors are very sparse, as they are here; thus we take $\kappa_2 > \kappa_1$.

There are a number of alternative ranking algorithms that could be considered within this same framework, including non-probabilistic metrics. Stand-alone comparisons of Bayesian Sets and several alternatives done in Heller and Ghahramani (2006) provide empirical motivation for its use. The generalization bounds provided in Sect. 3 provide theoretical motivation for using Bayesian Sets in high-dimensional settings with correlated features.

Feedback: Once the lists have been combined, we continue the discovery process by expanding the seed. A natural, unsupervised way of expanding the seed is to add the highest ranked non-seed item into the seed. Though not done here, one could also use a domain expert or even crowdsourcing to quickly scan the top ranked items and manually expand the seed from the discovered items. Then the process starts again; we do a combinatorial search for websites containing all pairs with the new seed item(s), extract possible new items from the websites, etc. We continue this process for as many iterations as we desire. In practice, there is no need to repeat searches for pairs of seed items that have already been explored in previous iterations. Also, we track which sites have been visited during source discovery and do not revisit these sites in later iterations.

Further implementation details are given in Appendix A. All of the components of our approach scale well to work fast on large problems: Item discovery is done with a regular expression, ranking is done with a single matrix multiplication, and all of the remaining steps require simple web queries. We used Google as the search engine for our experiments, however, Google creates an artificial restriction on the number of queries one can make per minute. This, and the speed of downloading webpages for item extraction, are the only two slow steps in our method—with the webpages already downloaded, the whole process took on average 1.9 s in our experiments in Sect. 4.1 (on a laptop with a 2.6 GHz i5 processor). Both issues would be fixed if we had our own web index and search engine, for instance, if we had direct access to Google's resources. Google or another search engine could implement this method and it would work in real time, and would give a substantial advantage over the state-of-the-art. In the meantime, companies or researchers wanting to curate master lists on specific topics can implement our method using one of the available search engines, as we do in our experiments in Sect. 4.

3 Theoretical results

The derivation for Bayesian Sets assumes independent features. In this application, features are internet domains, which are almost certainly correlated. Because Bayesian sets is the core of our method, we motivate its use in this application by showing that even in the presence of arbitrary dependence among features, prediction ability can be guaranteed as the sample size increases. We consider an arbitrary distribution from which the seed S is drawn, and prove that as long as there are a sufficient number of items, x will on expectation score highly if it is from the same distribution as the seed S. Specifically, we provide a lower bound for $\mathbb{E}_x\left[f_S(x)\right]$ that shows that the expected score of x is close to the score of S with high probability. We provide two results that use different proof techniques. This is a case where statistical learning theory provides theoretical backing for a Bayesian method.

Proposition 1 *Suppose $x^1, \ldots, x^m$ are sampled independently from the same distribution $\mathcal{D}$ over $\{0,1\}^N$. For all $m \geq 2$, with probability at least $1-\delta$ on the draw of the training set $S = \{x^1, \ldots, x^m\}$,*

$$\mathbb{E}_x\left[f_S(x)\right] \geq \frac{1}{m}\sum_{s=1}^{m} f_S(x^s) - \sqrt{\frac{1}{2m}\log\left(\frac{2N}{\delta}\right)}.$$

The proof of Proposition 1 is an application of Hoeffding's inequality and the union bound, and is given in the supplementary material.

Theorem 1 provides an additional result that has a weaker dependence on δ, but has no dependence on the number of features N, which in this application is the number of internet domains and is thus extremely large. Theorem 1 also gives insight into the dependence of generalization on the problem parameters.

Theorem 1 *Suppose $x^1, \dots, x^m$ are sampled independently from the same distribution $\mathcal{D}$. Define p_j to be the probability that feature j takes value 1. Let $p_{\min} = \min_j \min\{p_j, 1-p_j\}$ be the probability of the rarest feature. For all $p_{\min} > 0$, $\gamma_{\min} > 0$ and $m \geq 2$, with probability at least $1-\delta$ on the draw of the training set $S = \{x^1, \dots, x^m\}$,*

$$\mathbb{E}_{x\sim\mathcal{D}}\left[f_S(x)\right] \geq \frac{1}{m}\sum_{s=1}^{m} f_S(x^s) - \sqrt{\frac{1}{2m\delta} + \frac{6}{g(m)\delta} + O\left(\frac{1}{m^2 \log m}\right)},$$

where

$$g(m) := (\gamma_{\min} + (m-1)p_{\min}) \log\left(\frac{\gamma_{\min} + m - 1}{\gamma_{\min}}\right)$$

The proof of Theorem 1 involves showing that Bayesian Sets is a "stable" algorithm, in the sense of "pointwise hypothesis stability" (Bousquet and Elisseeff 2002). We show that the Bayesian Sets score is not too sensitive to perturbations in the seed set. Specifically, when an item is removed from the seed, the average change in score is bounded by a quantity that decays as $\frac{1}{m \log m}$. This stability allows us to apply a generalization bound from Bousquet and Elisseeff (2002). The proof of pointwise hypothesis stability is given in the supplementary material.

The two quantities with the most direct influence on the bound are $\gamma_{\min}$ and $p_{\min}$. We show in the supplementary material that for $p_{\min}$ small relative to $\gamma_{\min}$, the bound improves as $\gamma_{\min}$ increases (a stronger prior). This suggests that a strong prior improves stability when learning data with rare features. As $p_{\min}$ decreases, the bound becomes looser, suggesting that datasets with rare features will be harder to learn and will be more prone to errors.

The fact that the bound in Theorem 1 is independent of N provides motivation for using Bayesian Sets on very large scale problems, even when the feature independence assumption does not hold. It indicates that Bayesian Set's performance may not degrade when faced with high dimensional data. That is, Theorem 1 provides evidence that Bayesian Sets may not suffer from the curse of dimensionality.

The gap between the expected score of x and the (empirical) score of the seed goes to zero as $\frac{1}{\sqrt{m}}$. Thus when the seed is sufficiently large, regardless of the distribution over relevant items, we can be assured that the relevant items generally have high scores.

4 Experiments

We demonstrate and evaluate the algorithm with two sets of experiments. In the first set of experiments, we provide an objective comparison between our method, Google Sets, and Boo!Wa! using a randomly selected collection of list growing problems for which there exist gold standard lists. The true value of our work lies in the ability to construct lists for which there are not gold standards, so in a second set of experiments we demonstrate the algorithm's performance on more realistic, open-ended list growing problems. For all experiments, the steps and parameter settings of the algorithm were exactly the same and completely unsupervised other than specifying two seed items. Boo!Wa! and Google Sets are online tools and we used them as provided by http://boowa.com and Google Spreadsheet, respectively. The dates on which the online tools were accessed are given with the implementation details in Appendix A.

4.1 Wikipedia gold standard lists

Many of the experiments in past work on set completion, such as those used to develop the technology behind Boo!Wa! of Wang and Cohen (2008), involve manually constructed gold standards on arbitrarily selected topics. Manually constructed lists are inherently subjective, and experiments on a small set of arbitrarily selected topics do not demonstrate that the method will perform well in general. We thus use Wikipedia lists on randomly selected topics as gold standards, which is the same experimental design used by Sarmento et al. (2007) and Pantel et al. (2009).

The "List of . . ." articles on Wikipedia form a large corpus of potential gold standard lists that cover a wide variety of topics. We limited our experiments to the "featured lists," which are a collection of over 2,000 Wikipedia lists selected by Wikipedia editors due to their high quality. We required the lists used in our experiments to have at least 20 items, and excluded any lists of numbers (such as dates or sports scores). We created a random sample of list growing problems by randomly selecting 50 Wikipedia lists that met the above requirements. The selected lists covered a wide range of topics, including, for example, "storms in the 2005 Atlantic hurricane season," "current sovereign monarchs," "tallest buildings in New Orleans," "X-Men video games," and "Pittsburgh Steelers first-round draft picks." We treated the Wikipedia list as the gold standard for the associated list growing problem. We give the names of all of the selected lists in the supplementary material.

For each of the 50 list growing problems, we randomly selected two list items from the gold standard to form a seed. We used the seed as an input to our algorithm, and ran one iteration. We used the same seed as an input to Google Sets and Boo!Wa!. We compared the lists returned by our method, Google Sets, and Boo!Wa! to the gold standard list by computing two ranking measures: Precision@10 and average precision. Precision@10 measures the fraction of the top 10 items in the list that are found on the gold standard list:

$$\text{Precision@10} = \frac{1}{10}\sum_{i=1}^{10} \mathbb{1}_{[\text{item } i \text{ in ranked list is correct}]}. \tag{4}$$

Precision@10 is an intuitive measure that corresponds directly to the number of accurate results at the top of the list. Average precision is a commonly used measure that combines precision and recall, to ensure that lists are both accurate and complete. The average precision of a ranked list is defined as the average of the precision recall curve.

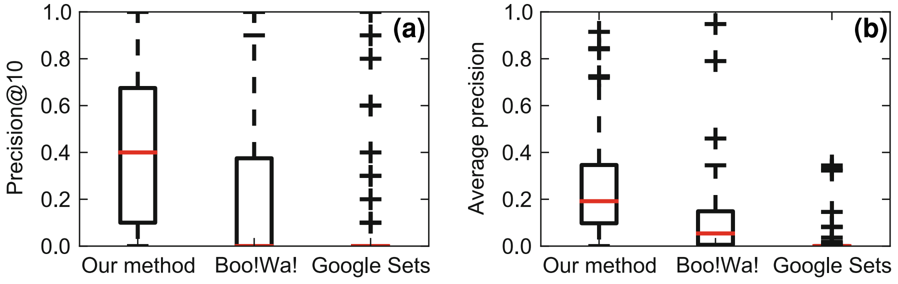

Fig. 1 **a** Precision@10 and **b** average precision across all 50 list growing problems sampled from Wikipedia. The median is indicated in *red* (Color figure online)

Before computing these measures, the seed items were removed from each list, as it is trivial to return the seed.

In Fig. 1 we show boxplots of the results across all 50 gold standard experiments. For both Google Sets and Boo!Wa!, the median precision at 10 was 0, indicating no correct results in the top 10. Our method performed significantly better ($p < 0.001$, signed-rank test), with median precision of 0.4, indicating 4 correct results in the top 10. Our method returned at least one relevant result in the top 10 for 80 % of the experiments, whereas Google Sets and Boo!Wa! returned at least one relevant result in the top 10 for only 24 % and 34 % of experiments, respectively. Our method performed well also in terms of recall, as measured in average precision, with a median average precision of 0.19, compared to 0 for Google Sets and 0.05 for Boo!Wa!. Boo!Wa! is a significant improvement over Google Sets, and our method is a large improvement over Boo!Wa!.

The supplementary material gives a list of the Precision@10 and average precision values for each of the Wikipedia gold standard experiments, as well as plots of Precision@5 and Precision@20.

There are some flaws with using Wikipedia lists as gold standards in these experiments. First, the gold standards are available online and could potentially be pulled directly without requiring any aggregation of experts across different sites. However, all three methods had access to the gold standards and the experiments did not favor any particular method, thus the comparison is informative. A more interesting experiment is one that necessitates aggregation of experts across different sites; these experiments are given in Sect. 4.3. Second, these results are only accurate insofar as the Wikipedia gold standard lists are complete. We limited our experiments to "featured lists" to have the best possible gold standards. A truly objective comparison of methods requires both randomly selected list problems and gold standards, and the Wikipedia lists, while imperfect, provide a useful evaluation.

4.2 Experimental analysis of algorithm steps

We performed several experiments modifying individual steps of the algorithm to explore the effect of design choices on performance, and to gather further insight into how each step contributes to the performance gain seen in Sect. 4.1 relative to

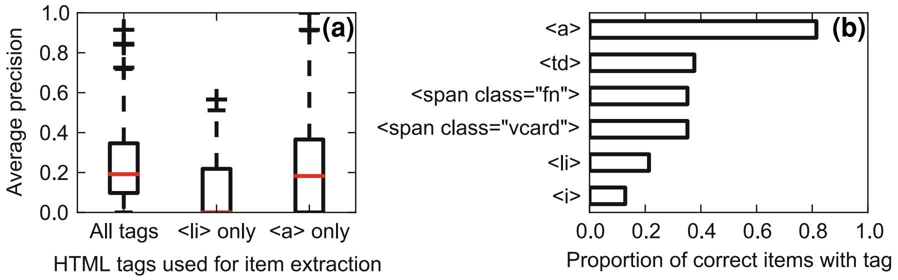

Fig. 2 **a** Average precision across the Wikipedia gold standard problems when extracting items using all tags (original implementation), `<li>` tags only, and `<a>` tags only. **b** The proportion of correct items extracted during the Wikipedia gold standard experiments that were found using a specific tag, for the six most commonly found tags

the baseline methods. We use the Wikipedia gold standard lists to explore experimentally the impact of which HTML tags are used for item extraction, the size of the seed when scoring, and the set of domains used in constructing the feature space.

In the item extraction step of our algorithm, we find the largest collection of HTML tags common to both seed items and extract all other items on the page that use that same collection of HTML tags. An alternative choice would be to look for a specific type of HTML tag, for example, list tags `<li>`, which could possibly reduce the number of incorrect items extracted. In Fig. 2a we repeated the Wikipedia gold standard experiments from Sect. 4.1, with a modified item extraction step in which we searched only for a specific type of tag: list tags `<li>` in one experiment, and hyperlink tags `<a>` in a second. Restricting to list tags significantly reduced average precision, while restricting to hyperlink tags produced results comparable to those obtained using all tags. Figure 2b provides insight into this difference by showing the proportion of all of the correct items extracted in the Wikipedia gold standard experiments that were extracted using a particular HTML tag, for the six most common tags. An item may be extracted using multiple HTML tags, either in a collection of tags or by discovering the same item on multiple pages, thus these proportions do not sum to 1. The value of 0.21 for `<li>` indicates that when extraction was limited to only `<li>` tags, we only obtained 21 % of the correct items that were obtained using all tags, which resulted in the significant performance drop seen in Fig. 2a. Limiting to `<a>` tags recovered 81 % of the correct items, which was enough to yield average precision comparable to that obtained using all tags. These results suggest that item extraction could be limited to link extraction, a problem for which many efficient software packages are widely available, without much loss.

In the gold standard experiments in Sect. 4.1 we used a seed of 2 items, the smallest possible seed size. When seed items are related in multiple contexts, as discussed for the case of Atlantic hurricanes, two seed items may not be enough to produce an accurate ranked list. In Fig. 3a, for each Wikipedia gold standard experiment we randomly selected additional seed items from the gold standard and used the larger seed to compute a new ranked list. Increasing the seed size, which further constrained the context of the relation between the seed items, produced a modest increase in

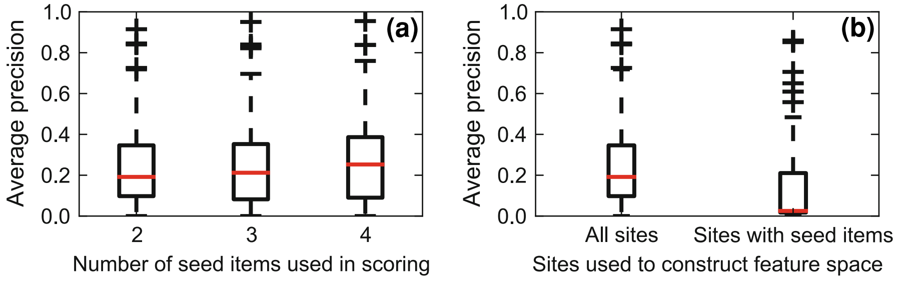

Fig. 3 Average precision across the Wikipedia gold standard problems when **a** expanding the number of seed items used in scoring, and **b** restricting the feature space construction to sites containing at least two seed items, that is, sites found in source discovery

performance. These results indicate that for many of the lists used in these experiments, two items were sufficient to specify the context. However, there is some gain to be had with a larger seed, and in general it is best for the user to specify as large a seed as possible.

We construct the binary feature space for each item using the domains of all of the sites where the item can be found. An alternative approach is to restrict the search to sites containing at least two seed items, that is, the sites found during the source discovery step. In Fig. 3b we repeated the Wikipedia gold standard experiments using this feature space strategy, and found that it significantly reduced average precision. In fact, Boo!Wa! uses a strategy similar to this one to construct a feature space, as we discuss in Sect. 5.

4.3 Open-ended experiments

In this set of experiments we demonstrate our method's performance on more realistic, open-ended list growing problems. For these problems gold standard lists are not available, and it is essential for the algorithm to aggregate results across many experts. We focus on four open-ended list growing problems: Boston events, Jewish foods, smartphone apps, and U.S. politicians. These are the sort of problems that our algorithm was designed to solve, and it performs very well, especially compared to the baselines.

4.3.1 Boston events

In this experiment, the seed items were two Boston events: "Boston arts festival" and "Boston harborfest." We ran the algorithm for 5 iterations, yielding 3,090 items. Table 1 shows the top 50 ranked items, together with the domain of the source site where they were discovered. There is no gold standard list to compare to directly, but the results are overwhelmingly actual Boston events. The events were aggregated across a variety of expert sources, including event sites, blogs, travel guides, and hotel pages. Table 2 shows the full set of results returned from Google Sets with the same two events as the seed, and the top 25 results returned by Boo!Wa!. Not only

Table 1 Items and the domain of their source sites from the top of the ranked list for the Boston events experiment

Item	**Source**
^{0}Boston arts festival	(original seed)
^{3}Cambridge river festival	bizbash.com
^{0}Boston harborfest	(original seed)
harborfest	
^{1}Boston chowderfest	celebrateboston.com
^{4}Berklee beantown jazz festival	pbase.com
the berklee beantown jazz festival,	
berklee bean town jazz festival	
^{2}Chinatown main street festival	blog.charlesgaterealty.com
www.chinatownmainstreet.org	
4th of july boston pops concert & fireworks display	travel2boston.us
boston 4th of july fireworks & concert	
Boston common frog pond	bostonmamas.com
ice skating on boston common frog pond	
First night boston	what-is-there-to-do.com
Boston dragon boat festival	pbase.com
hong kong dragon boat festival of boston	
dragon boat festival of boston	
Boston tea party re enactment	ef.com
Christopher columbus waterfront park	bostonmamas.com
Jimmy fund scooper bowl	bizbash.com
Opening our doors day	ef.com
Oktoberfest harvard square & harpoon brewery	sheratonbostonhotel.com
August moon festival	ef.com
Annual boston wine festival	worldtravelguide.net
Cambridge carnival	soulofamerica.com
Regattabar	berklee.edu
Arts on the arcade	berklee.edu
Franklin park zoo	hotels-rates.com
Faneuil hall annual tree lighting ceremony	ef.com
Annual oktoberfest and honk festival	ef.com
honk! festival	
Boston jazz week	telegraph.co.uk
Boston ballet	celebrateboston.com
Fourth of july reading of the declaration of independence	ef.com
Isabella stewart gardner museum	hotels-rates.com
Revere beach sand sculpting festival	bizbash.com
Shakespeare on the common	boston-discovery-guide.com
Boston bacon takedown	[...]bostonevents.blogspot.com
Jazz at the fort	berklee.edu
Cambridge dance party	[...]thrillsboston.blogspot.com
Boston celtic music festival	ef.com
Taste of the south end	bizbash.com
Greenway open market	travel2boston.us
Boston winter jubilee	ef.com
Urban ag fair	bostonmamas.com
Figment boston	festivaltrek.com
Boston kite festival	bostoneventsinsider.com
Chefs in shorts	bizbash.com
Old south meeting house	hotels-rates.com

Superscript numbers indicate the iteration at which the item was added to the seed via implicit feedback. [...] indicates the URL was truncated to fit in the figure. To improve readability, duplicate items were grouped and placed in italics

Table 2 Complete Google Sets results and top 25 Boo!Wa! results for the Boston events experiment (seed italicized)

Google Sets	Boo!Wa!
Boston arts festival	*Boston arts festival*
Boston harborfest	*Boston harborfest*
Whats going this month	Boston Fall Foliage
Interview with ann scott	Boston Wine Expo
Studio view with dannyo	Boston Flower and Garden Show
Tony savarino	Boston Vegetarian Food Festival
Artwalk 2011	The Boston Wine Expo
Greater boston convention visitors bureau	The Jazz Festival
Cambridge chamber of commerce	First Night Celebration
Boston tours	Thumbboston-vegetarian-food-festival
3 county fairground	Boston College Eagles
Boston massacre	Boston Vacation Rentals
	Best Boston Restaurants
	Boston Red Sox
	Boston Bruins
	Attractions in Boston:North End
	Attractions in Boston:Freedom Trail
	Attractions in Boston:Museum of Science
	Attractions in Boston:Prudential Center
	Attractions in Boston:New England Aquarium
	Attractions in Boston: Boston Public Garden
	Attractions in Boston:St. Patrick's Cathedral
	Attractions in Boston:South Beach - Ocean Drive
	Vacation-rentals-boston
	Boston-restaurants
	Parking-in-boston
	Shopping-boston

Google Sets and our implementation of our method return results all lower case, and in these tables we have capitalized the first letter for aesthetics. Boo!Wa! returns capitalized results, and we use here the capitalization that was returned

is the Google Sets list very short, but it does not contain any actual Boston events. Boo!Wa! was able to return some Boston events, but with a substantial amount of noise.

4.3.2 Jewish foods

In this experiment, the seed items were two Jewish foods: "Challah" and "Knishes." Although there are lists of foods that are typically found in Jewish cuisine, there is variety across lists and no authoritative definition of what is or is not a Jewish food. We completed 5 iterations of the algorithm, yielding 8,748 items. Table 3 shows the top 50 ranked items, together with their source domains. Almost all of the items are closely related to Jewish cuisine. The items on our list came from a wide variety of expert sources that include blogs, informational sites, bakery sites, recipe sites, dictionaries, and restaurant menus. In fact, the top 100 most highly ranked items came from a total of 52 unique sites. This diversity in source sites shows that the relevant items are found in many small lists, which provides motivation for using pairs of seed items for source discovery, as opposed to requiring all seed items to be on every source. In Table 4,

Table 3 Items and their source domains from the top of the ranked list for the Jewish foods experiment

Item	**Source**
^{0}Challah	(original seed)
braided challah	
^{3}Potato latkes	jewishveg.com
latkes; sweet potato latkes; potato latke	
^{1}Blintzes	jewfaq.org
cheese blintzes; blintz	
^{0}Knishes	(original seed)
potato knishes; knish	
^{2}Noodle kugel	pinterest.com
noodle kugel recipe; kugel; sweet noodle kugel	
^{4}Tzimmes	jewfaq.org
carrot tzimmes	
Matzo balls	jewishveg.com
matzo ball soup; matzo; matzoh balls	
Potato kugel	challahconnection.com
Passover recipes	lynnescountrykitchen.net
hanukkah recipes	
Gefilte fish	jewfaq.org
Honey cake	kveller.com
Soups, kugels & liver	allfreshkosher.com
Charoset	jewishveg.com
haroset	
Hamantaschen	butterfloureggs.com
Matzo meal	glattmart.net
Rugelach	pinterest.com
rugelach recipe	
Matzo brei	ilovekatzs.com
Cholent	jewfaq.org
Sufganiyot	kosheronabudget.com
Potato pancakes	jewishveg.com
Noodle pudding	epicurious.com
Kreplach	allmenus.com
Barley soup	ecampus.com
Mushroom barley	zagat.com
mushroom barley soup	
Chopped liver	ryedeli.com
Garlic mashed potatoes	tovascatering.com
Caponata	lynnescountrykitchen.net
Compote	kveller.com
Farfel & mushrooms	hungariankosher.com
farfel	
Kasha varnishkes	jinsider.com

we show the complete set of results returned from Google Sets for the same seed of Jewish foods, and the top 25 results returned by Boo!Wa!. Although the Google Sets results are foods, they are not closely related to Jewish cuisine. Boo!Wa! was able to return some Jewish foods, but also a lot of irrelevant results like “Shop Online,” “Lipkin’s Bakery,” and “Apple.”

Table 4 Complete Google Sets results and top 25 Boo!Wa! results for the Jewish foods experiment (seed italicized)

Google Sets	Boo!Wa!
Knishes	*Knishes*
Challah	*Challah*
Crackers	Applestrudel
Dinner rolls	Holishkes
Focaccie	Blintzes
Pains sucres	Gefilte
Pains plats	Apple
Biscotti integral de algarroba	Kasha
Souffle de zanahorias	Soup
Tarta de esparragos	Knishes.pdf
Leftover meat casserole	Knishes recipe PDF
Pan de canela	Shop Online
Focaccia	Hamantashen
Sweet hobz	Kamish Bread
Pranzu rolls	Apple Strudel
Focacce	Location
Chicken quesadillas	Danishes
Baked chicken chimichangas	Lipkin's Bakery
Honey mustard salad dressing	Flax Seed Bread
Dixxijiet hobz	Babka
Roast partridge	Pumpernickel Loaf
Fanny farmer brownies	Schnitzel
Pan pratos	Latke
Pan doce	Cole slaw
Cea rolls	Chopped Liver
Flat paes	Mini Potato Knish
Hobz dixxijiet	Oven Roasted Chicken

4.3.3 Smartphone apps

In this experiment, we began with two popular smartphone apps as the seed items: "Word lens" and "Aroundme." We ran the algorithm for 5 iterations, throughout which 7,630 items were extracted. Table 5 shows the top 50 most highly ranked items, together with the source domain where they were discovered. Not only are the results almost exclusively apps, but they come from a wide variety of sources including personal sites, review sites, blogs, and news sites. In Table 6, we show the lists returned by Google Sets and Boo!Wa! for the same seed, which are also predominantly apps.

4.3.4 U.S. Politicians

In this experiment, we began with two prominent U.S. politicians as the seed items: "Barack obama" and "Scott brown." We ran the algorithm for 5 iterations, yielding 8,384 items. Table 7 shows the top 50 most highly ranked items, together with the source site where they were discovered. All of the items in our list are names of politicians or politically influential individuals. In Table 8, we show the results returned from Google Sets and Boo!Wa! for the same seed. Google Sets managed to return only a few people related to politics. Boo!Wa! performed better than Google Sets, but the list still contains some noise, like "U.S. Senate 2014," "President 2016," and "Secret-service."

Table 5 Items and their source domains from the top of the ranked list for the smartphone apps experiment

Item	Source
[0]Word lens	(original seed)
[2]Read it later *read later*	iapps.scenebeta.com
[0]Aroundme	(original seed)
[3]Instapaper *instapaper app*	time.com
[4]Evernote *evernote app*	crosswa.lk
[1]Flipboard	crosswa.lk
Dolphin browser	1mobile.com
Skitch	worldwidelearn.com
Facebook messenger	crosswa.lk
Zite	adriandavis.com
Tweetbot	duckduckgo.com
Google currents	secure.crosswa.lk
Springpad	time.com
Imessage	iphoneae.com
Retina display	twicpic.blogspot.com
Ibooks	crosswa.lk
Dropbox *dropbox (app); dropbox app*	mobileappreviews.craveonline.com
Marco arment	wired.com
Doubletwist	appolicious.com
Google latitude	iapps.scenebeta.com
Gowalla	mobileappreviews.craveonline.com
Skype for ipad	secure.crosswa.lk
Hulu plus	appadvice.com
Icloud	thetechcheck.com
Qik video *qik*	1mobile.com
Find my friends	oradba.ch
Skydrive	crosswa.lk
Google shopper	mobileappreviews.craveonline.com
Swype	techcrunch.com
Pulse news reader	techcrunch.com
Spotify	crosswa.lk
Readability	tips.flipboard.com
Apple app store	socialmediaclub.org
Tweetdeck	iapps.scenebeta.com
Angry birds space	appys.com
Smartwatch	theverge.com
Vlingo	mobileappreviews.craveonline.com
Rdio	techcrunch.com
Google goggles	sofialys.com
Xmarks	40tech.com
Ios 6	zomobo.net
Ibooks author	duckduckgo.com
Google drive	geekandgirliestuff.blogspot.com
Facetime	bgpublishers.com.au

Table 6 Complete Google Sets results and top 25 Boo!Wa! results for the smartphone apps experiment (seed italicized)

Google Sets	Boo!Wa!
Word lens	*Word lens*
Aroundme	*Aroundme*
Lifestyle	Plants v. Zombies
View in itunes	Amazon
Itunes	Bloom
Jcpenney weekly deals	AIM
Coolibah digital scrapbooking	Plants vs. Zombies
Epicurious recipes shopping list	Layar
170,000 recipes bigoven	Bjrk: Biophilia
Cf iviewer	Wikipedia Mobile
Txtcrypt	Web Source Viewer
Speak4it	WhatTheFont
Off remote free	The Weather Channel
Catholic calendar	EDITION29 STRUCTURES
Gucci	Dexigner
Board	Google
Ziprealty real estate	Zipcar
Allsaints spitalfields	Thrutu
Lancome make up	Google Earth
Pottery barn catalog viewer	Four-Square
Amazon mobile	Wikipedia
Gravity clock	Facebook
Dace	Kindle
Zara	Skype
Style com	Mint
Iridiumhd	Wi-Fi Finder App
Ebanner lite	Green Gas Saver

5 Related work

There is a substantial body of work in areas or tasks related to the one which we have presented, which we can only briefly review here. There are a number of papers on various aspects of "set expansion," often for completing lists of entities from structured lists, like those extracted from Wikipedia (Sarmento et al. 2007), using rules from natural language processing or topic models (Tran et al. 2010; Sadamitsu et al. 2011), or from opinion corpora (Zhang and Liu 2011). The task we explore here is *web-based set expansion* and methods developed for other set expansion tasks are not directly applicable. See, for example, Jindal and Roth (2011), for a review of different set expansion problems.

There is good deal of work in the machine learning community on aggregating ranked lists (*e.g.*, Dwork et al., 2001). These are lists that are typically already cleaned, fixed in scope, and ranked by individual experts, unlike our case. There is also a body of work on aggregated search (Lalmas 2011; Renda and Straccia 2003; Hsu and Taksa 2005; Beg and Ahmad 2003), which typically uses a text query to aggregate results from multiple search engines, or of multiple formats or domains (e.g. image and news), and returns links to the full source. Our goal is not to rank URLs but to scrape out and rank information gleaned from them. There are many resources for performing a search

Table 7 Items and their source domains from the top of the ranked list for the U.S. politicians experiment

Item	Source
^{0}Barack obama	(original seed)
obama	
^{0}Scott brown	(original seed)
^{1}John kerry	publicpolicypolling.com
^{3}Barney frank	masslive.com
^{4}John mccain	publicpolicypolling.com
mccain	
^{2}Nancy pelosi	theladypatriot.com
pelosi	
Mitch mcconnell	publicpolicypolling.com
Joe lieberman	publicpolicypolling.com
Mike huckabee	publicpolicypolling.com
Mitt romney	masslive.com
Bill clinton	mediaite.com
John boehner	audio.wrko.com
boehner	
Hillary clinton	blogs.wsj.com
Jon kyl	tpmdc.talkingpointsmemo.com
Joe biden	publicpolicypolling.com
Rudy giuliani	publicpolicypolling.com
Harry reid	theladypatriot.com
Olympia snowe	publicpolicypolling.com
Lindsey graham	politico.com
Newt gingrich	masspoliticsprofs.com
Jim demint	theladypatriot.com
Arlen specter	theladypatriot.com
Dick cheney	blogs.wsj.com
George w bush	wellgroomedmanscape.com
george w. bush	
Eric holder	disruptthenarrative.com
Dennis kucinich	publicpolicypolling.com
Timothy geithner	tpmdc.talkingpointsmemo.com
Barbara boxer	publicpolicypolling.com
Tom coburn	itmakessenseblog.com
Orrin hatch	publicpolicypolling.com
Michael bloomberg	masspoliticsprofs.com
Elena kagan	audio.wrko.com
Maxine waters	polination.wordpress.com
Al sharpton	porkbarrel.tv
Rick santorum	audio.wrko.com
Ted kennedy	newomenforchange.org
Janet napolitano	disruptthenarrative.com
Jeff sessions	tpmdc.talkingpointsmemo.com
Jon huntsman	publicpolicypolling.com
Michele bachmann	publicpolicypolling.com
Al gore	publicpolicypolling.com
Rick perry	publicpolicypolling.com
Eric cantor	publicpolicypolling.com
Ben nelson	publicpolicypolling.com
Karl rove	politico.com

Table 8 Complete Google Sets results and top 25 Boo!Wa! results for the U.S. politicians experiment (seed italicized)

Google Sets	Boo!Wa!
Barack obama	*Barack obama*
Scott brown	*Scott brown*
Our picks movies	William Galvin
Sex	Secret-service
Department of justice	Sheldon Whitehouse
Viral video	Debbie Stabenow
Africa	Dennis Kucinich
One persons trash	Susana Martinez
Donald trump	Stephen Colbert
New mom confessions	Martin O'Malley
Nonfiction	Claire McCaskill
Libya	U.S. Senate 2012
Sarah palin	Brian Schweitzer
Mtv	Michele Bachmann
Alan greenspan	Condoleezza Rice
Great recession	U.S. Senate 2014
Life stories	Lisa Murkowski
Jon hamm	Lindsey Graham
Islam	Maria Cantwell
The killing	Jeanne Shaheen
American idol	South Carolina
Middle east	North Carolina
Celebrity	Terry Branstad
Tea parties	President 2016
Budget showdown	Tommy Thompson
	Brian Sandoval
	Offshore drilling

or query by example. They often involve using a single example of a full document or image in order to retrieve more documents, structures within documents, or images (Chang and Lui 2001; Liu et al. 2003; Wang and Lochovsky 2003; Zhai and Liu 2005).

Methods such as that of Gupta and Sarawagi (2009) and Pantel et al. (2009) learn semantic classes, which could be used to grow a list, but require preprocessing which crawls the web and creates an index of HTML lists in an unsupervised manner. Kozareva et al. (2008) present a method for using a semantic class name and a seed of example instances to discover other instances from the same class on the web, using search queries. They limit the search to instances that match a very specific pattern of words ("*class name* such as *seed item* and *"), thus requiring the semantic class to have enough instances and web coverage that all instances match the pattern somewhere on the Internet. We found that this was not the case for more realistic open-ended problems, like Boston events and the others in Sect. 4.3. Paşca (2007a,b) also discovers semantic class attributes and instances, but using web query logs rather than actual internet sites.

Systems for learning categories and relations of entities on the web, like the Never-Ending Language Learner (NELL) system (Carlson et al. 2010a,b; Verma and Hruschka 2012), or KnowItAll (Etzioni et al. 2005) can be used to construct lists but require extensive preprocessing. We do not preprocess, instead we perform information extraction online, deterministically, and virtually instantaneously given access to a search engine. There is no restriction to HTML list structures or need for more time consuming learning methods (Freitag 1998; Soderland et al. 1999). We also do not require human-labeled web pages like wrapper induction methods (Kushmerick 1997).

The Set Expander for Any Language (SEAL) of Wang and Cohen (2007, 2008), implemented in Boo!Wa!, at first appears similar to our work but differs in significant ways. Wang and Cohen (2008) describe four strategies for source discovery, of which "unsupervised increasing seed size (ISS)" is most similar to ours. Unsupervised ISS begins with two seed items and iteratively expands the seed in the same way as our implicit feedback, by adding the most highly ranked non-seed item to the seed at each iteration. Within each iteration, unsupervised ISS uses only a subset of the seed items to try to further expand the set. Specifically, it uses the most recently added seed item together with three additional randomly-selected seed items, and searches for source sites containing all four of these items. Our source discovery differs in two major ways. First, our combinatorial search strategy uses all seed items in every iteration, rather than a randomly-selected subset of four seed items. Second, we use only pairs of seed items to find source sites, rather than requiring the source sites to contain four seed items. With this strategy we find all of the sites discovered by ISS, as well as additional sites that have less than four seed items. Once the source sites have been found, SEAL extracts new items by learning a character-based wrapper that finds patterns of characters that are common to the seed items. This is similar in concept to the way that we extract new items, although SEAL allows arbitrary patterns of characters whereas we look specifically for patterns in the HTML tree structure. Possibly the most significant differences between SEAL and our approach lie in ranking the extracted items. When the initial number of seed items is small, as in the list growing problems that we considered here, Wang and Cohen (2008) recommend a ranking algorithm that uses a random walk over a graph that contains nodes for the extracted items, the wrappers learned for each source site, and the source sites themselves. Wang and Cohen (2008) also considered using Bayesian Sets to rank, and in fact recommended its use when the number of initial seed items was large. However, the way in which SEAL constructs the feature space to be used by Bayesian Sets is critically different. SEAL uses two sets of features: the sources sites on which the extracted item was found during list extraction, and the wrappers that extracted it. We use the complete set of domains (not sites) where the extracted item can be found, and do not limit ourselves to the domains of source sites. Using all domains as features rather than just those containing seed items is very important for reducing the rank of items that happened to show up on the same site as a few seed items but in general are found on very different types of domains, as shown in our experiments in Sect. 4.2. Finding this full set of domains requires an additional set of queries, one for each item to be ranked, however these types of queries can be done efficiently when one has access to a web index. These differences between our method and Boo!Wa! translate into the order of magnitude improvement in the quality of the returned lists shown throughout Sect. 4.

6 Conclusions

The next generation of search engines should not simply retrieve URLs, but should aim at retrieving information. We designed a system that leads into this next generation, leveraging information from across the Internet to grow an authoritative list on almost any topic.

The gold standard experiments showed that our method performs well on a wide range of list growing problems, and provided insight into the effect of design choices on performance. There are several conclusions that can be drawn from the empirical results. First, we showed how increasing the number of seed items can improve performance by constraining the relationship between seed items, suggesting that users should be encouraged to provide as many seed items as possible. Second, even when a large seed is available, our results in Sect. 4.3 demonstrate the importance of using small groups of seed items for source site discovery (we used pairs). There we showed that in real problems, relevant items must be aggregated from many websites and are often only found together with a small collection of other relevant items.

Of all of the design choices, we found that the construction of the feature space for ranking discovered items had the largest impact on performance. Items that are likely to be correct, and should thus be highly ranked, are those that are found frequently on websites where seed items are found and, equally importantly, are not found frequently where seed items are not found. A feature space that considers only the sites on which seed items are found is not able to distinguish between items that are highly correlated with the seed and items that are just generally common. Our solution was to construct the feature space using an individual search query for each discovered item, allowing us to verify that the item was not frequently found without seed items. This led to substantially improved results compared to a feature space using only sites containing seed items, though at a cost of more search queries.

This feature space construction is a major source of improvement, but can be time consuming given the restrictions that Google and other search engines place on the number of queries per minute. Without this restriction, our results can be obtained in real-time on almost any computer. One major challenge that needs to be overcome to have a real-time implementation for public use is either to embed code like ours within a search engine infrastructure, or to find ways to use fewer search queries, chosen in an intelligent way, to construct a similar feature space that incorporates information about sites without seed items. Another challenge not handled here is to build in knowledge of language. Our results are not English-specific, but with some knowledge of natural language, higher quality results could potentially be obtained.

The Wikipedia gold-standard experiments provided a framework for quantifying performance on a range of list topics, but the open-ended experiments showed, qualitatively, the true strength of the developed method. For real problems for which complete lists were not available online, we found that the algorithm produced meaningful lists, with information extracted from a wide variety of sources. Moreover, the lists compared favorably with those from existing related technology.

In addition to these empirical results, we presented a theoretical bound that justifies the use of Bayesian Sets in a setting where its feature independence assumptions are not met. This bound will help to motivate its continued use in set expansion problems.

The list growing algorithm we presented was implemented on a laptop, with minimal heuristics and hand-tuning, and no language-specific processing or handling of special cases. Yet, the results are good enough to be directly useful to users in many cases. These encouraging results are an indication of the power of high-quality algorithms to gather crowdsourced information.

Acknowledgments Cynthia Rudin acknowledges funding from MIT Lincoln Laboratory and from NSF IIS-1053407. Katherine Heller acknowledges funding from a NSF postdoctoral fellowship and NIH P30 DA028803.

Appendix A implementation details

Source discovery: This step requires submitting the query *"term1" "term2"* to a search engine. In our experiments we used Google as the search engine, but any index would suffice. We retrieved the top 100 results.

List extraction: For each site found with the combinatorial search, we look for HTML tags around the seed items. We use the following lines of HTML to illustrate:

```
<h2><b><a href="example1.com"> Boston Harborfest </a></b></h2>
<b><a href="example2.com"> Jimmy fund scooper bowl </a></b>
<b><a href ="example3.com"> the Boston Arts Festival 2012</a></b>
<h3><b><a href="example4.com"> Boston bacon takedown </a></b>< /h3>
<a href="example5.com"> Just a url </a>
```

For each of the two seed items used to discover this source, we search the HTML for the pattern:

< largest set of tags > (up to 5 words) seed item (up to 5 words) < matching end tags >.

In the above example, if the first seed item is "Boston arts festival," then it matches the pattern with the HTML tags: `<b><a>`. If the second seed item is "Boston harborfest," it matches the pattern with HTML tags: `<h2><b><a>`. We then find the largest set of HTML tags that are common to both seed items, for this site. In this example, "Boston arts festival" does not have the `<h2>` tag, so the largest set of common tags is: `<b><a>`. If there are no HTML tags common to both seed items, we discard the site. Otherwise, we extract all items on the page that use the same HTML tags. In this example, we extract everything with both a `<b>` and an `<a>` tag, which means "Jimmy fund scooper bowl" and "Boston bacon takedown," but not "Just a url."

In our experiments, to avoid search spam sites with extremely long lists of unrelated keywords, we reject sources that return more than 300 items. We additionally applied a basic filter rejecting items of more than 60 characters or items consisting of only numbers and punctuation. No other processing was done.

Feature space: We do separate Google searches for each item we have extracted to find the set of webpages containing it. We use quotes around the query term, discard results when Google's spelling correction system modifies the query, and retrieve the top 300 search results.

Ranking: In all of our experiments we took $\kappa_2 = 5$ and $\kappa_1 = 2$, similarly to that done in Heller and Ghahramani (2006).

Feedback: To avoid filling the seed with duplicate items like "Boston arts festival" and "The boston arts festival 2012," in our implicit feedback we do not add items to the seed if they are a sub- or super-string of a current seed item.

Access of online tools: The results for Boo!Wa! and Google Sets used in Sect.4.1 were retrieved from their respective online tools on November 14, 2012, and those used in Sect. 4.3 were retrieved on December 13, 2012. Results for our algorithm, which depend on Google search results and the content of the source webpages, were obtained in the period March 14–21, 2013 for the Wikipedia gold-standard experiments in Sects. 4.1 and 4.2, and in the period May 28–30, 2012 for the open-ended experiments in Sect. 4.3.

References

Beg MMS, Ahmad N (2003) Soft computing techniques for rank aggregation on the world wide web. World Wide Web 6(1):5–22

Bousquet O, Elisseeff A (2002) Stability and generalization. J Mach Learn Res 2:499–526

Carlson A, Betteridge J, Kisiel B, Settles B, Hruschka ER, Mitchell TM (2010a) Toward an architecture for never-ending language learning. In: Proceedings of the 24th conference on artificial intelligence, AAAI '10.

Carlson A, Betteridge J, Wang RC, Hruschka ER, Mitchell TM (2010b) Coupled semi-supervised learning for information extraction. In: Proceedings of the 3rd ACM international conference on web search and data mining, WSDM '10, pp 101–110.

Chang CH, Lui SC (2001) IEPAD: Information extraction based on pattern discovery. In: Proceedings of the 10th international conference on world wide web, WWW '01, pp 681–688.

Dwork C, Kumar R, Naor M, Sivakumar D (2001) Rank aggregation methods for the web. In: Proceedings of the 10th international conference on world wide web, WWW '01, pp 613–622.

Etzioni O, Cafarella M, Downey D, Popescu AM, Shaked T, Soderland S, Weld DS, Yates A (2005) Unsupervised named-entity extraction from the web: an experimental study. Artif Intell 165(1):91–134

Freitag D (1998) Information extraction from HTML: application of a general machine learning approach. In: Proceedings of the 15th national conference on artificial intelligence, AAAI '98, pp 517–523.

Ghahramani Z, Heller KA (2005) Bayesian sets. In: Advances in neural information processing systems 18, NIPS '05, pp 435–442.

Gupta R, Sarawagi S (2009) Answering table augmentation queries from unstructured lists on the web. Proceedings of the VLDB Endowment 2:289–300

Heller KA, Ghahramani Z (2006) A simple Bayesian framework for content-based image retrieval. In: Proceedings of the IEEE conference on computer vision and pattern recognition, CVPR '06, pp 2110–2117.

Hsu DF, Taksa I (2005) Comparing rank and score combination methods for data fusion in information retrieval. Inf Retr 8(3):449–480

Jindal P, Roth D (2011) Learning from negative examples in set-expansion. In: Proceedings of the 2011 11th IEEE international conference on data mining, ICDM '11, pp 1110–1115.

Kozareva Z, Riloff E, Hovy E (2008) Semantic class learning from the web with hyponym pattern linkage graphs. In: Proceedings of the 46th annual meeting of the association for computational linguistics: human language technologies, ACL '08, pp 1048–1056.

Kushmerick N (1997) Wrapper induction for information extraction. PhD thesis, University of Washington.

Lalmas M (2011) Aggregated search. In: Melucci M, Baeza-Yates R (eds) Advanced topics on information retrieval. Springer, Berlin

Liu B, Grossman R, Zhai Y (2003) Mining data records in web pages. In: Proceedings of the 9th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '03, pp 601–606.

Paşca M (2007a) Organizing and searching the world wide web of facts—step two: harnessing the wisdom of the crowds. In: Proceedings of the 16th international conference on world wide web, WWW '07, pp 101–110.

Paşca M (2007b) Weakly-supervised discovery of named entities using web search queries. In: Proceedings of the 16th ACM conference on information and knowledge management, CIKM '07, pp 683–690.

Pantel P, Crestan E, Borkovsky A, Popescu AM, Vyas V (2009) Web-scale distributional similarity and entity set expansion. In: Proceedings of the 2009 conference on empirical methods in natural language processing, EMNLP '09, pp 938–947.

Renda ME, Straccia U (2003) Web metasearch: rank versus score based rank aggregation methods. In: Proceedings of the 2003 ACM symposium on applied computing, SAC '03, pp 841–846.
Sadamitsu K, Saito K, Imamura K, Kikui G (2011) Entity set expansion using topic information. In: Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies, ACL '11, vol 2, pp 726–731.
Sarmento L, Jijkoun V, de Rijke M, Oliveira E (2007) "More like these" : growing entity classes from seeds. In: Proceedings of the 16th ACM conference on information and knowledge management, CIKM '07, pp 959–962.
Soderland S, Cardie C, Mooney R (1999) Learning information extraction rules for semi-structured and free text. Mach Learn 34(1–3):233–272
Tran MV, Nguyen TT, Nguyen TS, Le HQ (2010) Automatic named entity set expansion using semantic rules and wrappers for unary relations. In: Proceedings of the 2010 international conference on Asian language processing, IALP '10, pp 170–173.
Verma S, Hruschka ER (2012) Coupled bayesian sets algorithm for semi-supervised learning and information extraction. In: Proceedings of the 2012 European conference on machine learning and knowledge discovery in databases, ECML PKDD '12, pp 307–322.
Wang J, Lochovsky FH (2003) Data extraction and label assignment for web databases. In: Proceedings of the 12th international conference on world wide web, WWW '03, pp 187–196.
Wang RC, Cohen WW (2007) Language-independent set expansion of named entities using the web. In: Proceedings of the 2007 7th IEEE international conference on data mining, ICDM '07, pp 342–350.
Wang RC, Cohen WW (2008) Iterative set expansion of named entities using the web. In: Proceedings of the 2008 8th IEEE international conference on data mining, ICDM '08, pp 1091–1096.
Zhai Y, Liu B (2005) Web data extraction based on partial tree alignment. In: Proceedings of the 14th international conference on world wide web, WWW '05, pp 76–85.
Zhang L, Liu B (2011) Entity set expansion in opinion documents. In: Proceedings of the 22nd ACM conference on hypertext and hypermedia, HT '11, pp 281–290.

Supplement to Growing a List

Benjamin Letham[*] Cynthia Rudin[†] Katherine A. Heller[‡]

This supplementary material expands on the experiments and theory given in the main text of Growing a List. In Section 1 we give further detail on the Wikipedia gold standard experiments. In Section 2 we give the proofs of our main theoretical results, Proposition 1 and Theorem 1.

1 Wikipedia Gold Standard Experiments

In Table S1 we give a complete enumeration of the results from the Wikipedia gold standard experiments. For each list growing problem, we provide the Precision@10 and average precision (AveP) for all three methods (our method, Google Sets, and Boo!Wa!). This table illustrates both the diversity of the sampled list growing problems and the substantially improved performance of our method compared to the others. We focused on Precision@10 because 10 is the typical number of search results returned by a search engine. We supplement these results further with Precision@5 and Precision@20 in Figure S1.

2 Proofs

In this section, we provide the proofs of Proposition 1 and Theorem 1, comments on the effect of the prior ($\gamma_{\min}$) on generalization, and an example showing that Bayesian Sets does not satisfy the requirements for "uniform stability" defined by Bousquet and Elisseeff (2002).

Recall the definition of the scoring function:

$$f_S(x) := \frac{1}{Z(m)} \sum_{j=1}^{N} x_j \log \frac{\alpha_j + \sum_{s=1}^{m} x_j^s}{\alpha_j} + (1 - x_j) \log \frac{\beta_j + m - \sum_{s=1}^{m} x_j^s}{\beta_j}, \tag{S1}$$

where

$$Z(m) := N \log \left(\frac{\gamma_{\min} + m}{\gamma_{\min}} \right)$$

[*]Operations Research Center, MIT
[†]MIT Sloan School of Management
[‡]Center for Cognitive Neuroscience, Statistical Science, Duke

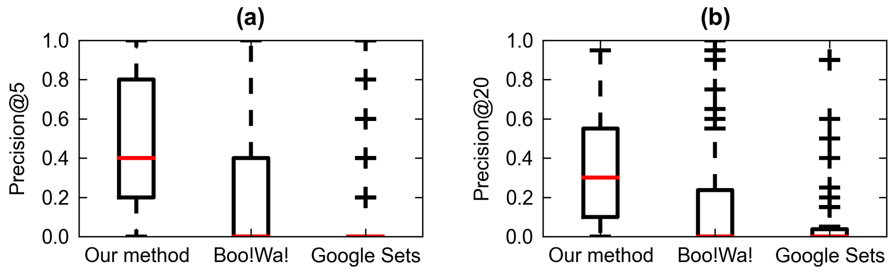

Figure S1: **(a)** Precision@5 and **(b)** Precision@20 across all 50 list growing problems sampled from Wikipedia. The median is indicated in red.

Table S1: Results for all 50 experiments with Wikipedia gold standards. "Us" indicates our method, "BW" indicates Boo!Wa!, and "GS" indicates Google Sets. "List of" has been removed from the title of each Wikipedia article, for brevity.

	Precision@10			**AveP**		
Wikipedia gold standard list	**Us**	**BW**	**GS**	**Us**	**BW**	**GS**
Awards and nominations received by Chris Brown	1	1	0	0.66	0.34	0
Medal of Honor recipients educated at the United States Military Academy	0.2	0	0	0.28	0.01	0
Nine Inch Nails concert tours	0.8	0	0	0.51	0	0
Bleach episodes (season 4)	0	0	0	0	0	0
Storms in the 2005 Atlantic hurricane season	0.1	0	0	0.13	0.11	0
Houses and associated buildings by John Douglas	0.6	0.6	0	0.26	0.32	0
Kansas Jayhawks head football coaches	0.9	0.8	0	0.91	0.79	0
Kraft Nabisco Championship champions	0	0	0	0.05	0.05	0
Washington state symbols	0	0	0	0	0	0
World Heritage Sites of the United Kingdom	0.3	0	0	0.19	0.08	0
Philadelphia Eagles head coaches	0	0	0	0.09	0	0
Los Angeles Dodgers first-round draft picks	0.6	0	0	0.19	0.28	0.00
New York Rangers head coaches	0.3	0.8	0	0.16	0.46	0
African-American Medal of Honor recipients	1	0	0	0.73	0.06	0
Current sovereign monarchs	0.5	0	0	0.15	0	0
Brotherhood episodes	0.9	0.2	0	0.72	0.06	0
Knight's Cross of the Iron Cross with Oak Leaves recipients (1945)	0	0	0	0	0.01	0.00
Pittsburgh Steelers first-round draft picks	0.1	0	0	0.38	0.00	0
Tallest buildings in New Orleans	0.6	0	0.6	0.45	0	0.08
Asian XI ODI cricketers	0.2	0	0.4	0.18	0.01	0.08
East Carolina Pirates head football coaches	0.1	0	0	0.05	0.01	0
Former championships in WWE	0.4	0	0.4	0.31	0.09	0.15
Space telescopes	0	0	0	0	0	0
Churches preserved by the Churches Conservation Trust in Northern England	0	0	0	0	0	0
Canadian Idol finalists	0.4	0	0.2	0.27	0.14	0.02
Wilfrid Laurier University people	0.3	0	0	0.34	0.11	0
Wario video games	0.1	0.6	0.8	0.12	0.22	0.34
Governors of Washington	0.5	0	0	0.42	0.13	0
Buffalo Sabres players	0.1	0	0	0.03	0	0
Australia Twenty20 International cricketers	0.6	0	1	0.24	0.01	0.32
Awards and nominations received by Madonna	0.9	1	0.2	0.70	0.13	0.00
Yukon Quest competitors	0.7	0.4	0.2	0.02	0.35	0.00
Arsenal F.C. players	0.8	0	0	0.85	0.18	0
Victoria Cross recipients of the Royal Navy	0.4	0	0	0.12	0.01	0
Formula One drivers	0	0.6	1	0	0.15	0.01
Washington & Jefferson College buildings	0	0	0	0	0	0
X-Men video games	0.4	0.2	0	0.27	0.05	0
Governors of Florida	0.4	0	0	0.25	0.04	0
The Simpsons video games	0.1	0	0	0.18	0	0
Governors of New Jersey	0.7	0	0	0.34	0.07	0
Uncharted characters	0.4	0	0.6	0.27	0.01	0.33
Miami Marlins first-round draft picks	0.4	1	0	0.16	0.27	0
Tallest buildings in Dallas	0.7	0.2	0	0.34	0.14	0
Cities and towns in California	0.8	0.6	1	0.35	0.04	0.04
Olympic medalists in badminton	0.3	0	0	0.13	0.05	0
Delegates to the Millennium Summit	0.9	0.4	0	0.51	0.01	0
Honorary Fellows of Jesus College, Oxford	0	0.4	0	0.03	0.34	0
Highlander: The Raven episodes	0.2	1	0	0.14	0.95	0
Voice actors in the Grand Theft Auto series	0.4	0	0	0.16	0.18	0
Medal of Honor recipients for the Vietnam War	0.9	0.8	0	0.84	0.08	0

and $\gamma_{\min} := \min_j \min\{\alpha_j, \beta_j\}$. We begin by showing that the normalized score $f_S(x)$ in (S1) takes values only on $[0, 1]$.

Lemma S1. *$0 \le f_S(x) \le 1$.*

Proof. It is easy to see that $f_S(x) \ge 0$. To see that $f_S(x) \le 1$,

$$\begin{aligned}
\max_{S,x} f_S(x) &= \frac{1}{Z(m)} \max_{S,x} \sum_{j=1}^{N} \left(x_j \log \frac{\alpha_j + \sum_{s=1}^m x_j^s}{\alpha_j} + (1 - x_j) \log \frac{\beta_j + m - \sum_{s=1}^m x_j^s}{\beta_j} \right) \\
&\le \frac{1}{Z(m)} \sum_{j=1}^{N} \max_{x_j, x_j^1, \ldots, x_j^m} \left(x_j \log \frac{\alpha_j + \sum_{s=1}^m x_j^s}{\alpha_j} + (1 - x_j) \log \frac{\beta_j + m - \sum_{s=1}^m x_j^s}{\beta_j} \right) \\
&= \frac{1}{Z(m)} \sum_{j=1}^{N} \max \left\{ \max_{x_j^1, \ldots, x_j^m} \log \frac{\alpha_j + \sum_{s=1}^m x_j^s}{\alpha_j}, \max_{x_j^1, \ldots, x_j^m} \log \frac{\beta_j + m - \sum_{s=1}^m x_j^s}{\beta_j} \right\} \\
&= \frac{1}{Z(m)} \sum_{j=1}^{N} \max \left\{ \log \frac{\alpha_j + m}{\alpha_j}, \log \frac{\beta_j + m}{\beta_j} \right\} \\
&= \frac{1}{Z(m)} \sum_{j=1}^{N} \log \frac{\min\{\alpha_j, \beta_j\} + m}{\min\{\alpha_j, \beta_j\}} \\
&\le \frac{1}{Z(m)} \sum_{j=1}^{N} \log \frac{\gamma_{\min} + m}{\gamma_{\min}} \\
&= 1.
\end{aligned}$$

□

Now we provide the proof to Proposition 1.

Proof of Proposition 1. For convenience, denote the seed sample average as $\mu_j := \frac{1}{m} \sum_{s=1}^m x_j^s$, and the probability that $x_j = 1$ as $p_j := \mathbb{E}_x[x_j]$. Then,

$$\begin{aligned}
&\frac{1}{m} \sum_{s=1}^{m} f_S(x^s) - \mathbb{E}_x \left[f_S(x) \right] \\
&\qquad = \frac{1}{N \log \left(\frac{\gamma_{\min} + m}{\gamma_{\min}} \right)} \sum_{j=1}^{N} \left((\mu_j - p_j) \log \frac{\alpha_j + m\mu_j}{\alpha_j} + (p_j - \mu_j) \log \frac{\beta_j + m(1 - \mu_j)}{\beta_j} \right) \\
&\qquad \le \frac{1}{N} \sum_{j=1}^{N} |\mu_j - p_j|. \qquad \text{(S2)}
\end{aligned}$$

For any particular feature j, Hoeffding's inequality (Hoeffding, 1963) bounds the difference between the empirical average and the expected value:

$$\mathbb{P}(|\mu_j - p_j| > \epsilon) \le 2 \exp\left(-2m\epsilon^2\right). \qquad \text{(S3)}$$

We then apply the union bound to bound the average over features:

$$\begin{aligned}
\mathbb{P}\left(\frac{1}{N} \sum_{j=1}^{N} |\mu_j - p_j| > \epsilon \right) &\le \mathbb{P}\left(\bigcup_{j=1}^{N} \{ |\mu_j - p_j| > \epsilon \} \right) \\
&\le \sum_{j=1}^{N} \mathbb{P}\left(|\mu_j - p_j| > \epsilon \right) \\
&\le 2N \exp\left(-2m\epsilon^2\right). \qquad \text{(S4)}
\end{aligned}$$

Thus,

$$\mathbb{P}\left(\frac{1}{m}\sum_{s=1}^{m} f_S(x^s) - \mathbb{E}_x\left[f_S(x)\right] > \epsilon\right) \leq 2N\exp\left(-2m\epsilon^2\right), \tag{S5}$$

and the proposition follows directly. □

The bound in Proposition 1 has a tighter dependence on δ than the bound in Theorem 1, however it depends inversely on N, the number of features.

We now present the proof of Theorem 1. The result uses the algorithmic stability bounds of Bousquet and Elisseeff (2002), specifically the bound for pointwise hypothesis stability. We begin by defining an appropriate loss function. Suppose x and S were drawn from the same distribution $\mathcal{D}$. Then, we wish for $f_S(x)$ to be as large as possible. Because $f_S(x) \in [0,1]$, an appropriate metric for the loss in using f_S to score x is:

$$\ell(f_S, x) = 1 - f_S(x). \tag{S6}$$

Further, $\ell(f_S, x) \in [0,1]$.

For algorithmic stability analysis, we will consider how the algorithm's performance changes when an element is removed from the training set. We define a modified training set in which the i'th element has been removed: $S^{\setminus i} := \{x^1, \ldots, x^{i-1}, x^{i+1}, \ldots, x^m\}$. We then define the score of x according to the modified training set:

$$f_{S^{\setminus i}}(x) = \frac{1}{Z(m-1)}\sum_{j=1}^{N} x_j \log\frac{\alpha_j + \sum_{s\neq i} x_j^s}{\alpha_j} + (1-x_j)\log\frac{\beta_j + (m-1) - \sum_{s\neq i} x_j^s}{\beta_j}, \tag{S7}$$

where

$$Z(m-1) = N\log\left(\frac{\gamma_{\min} + m - 1}{\gamma_{\min}}\right). \tag{S8}$$

We further define the loss using the modified training set:

$$\ell(f_{S^{\setminus i}}, x) = 1 - f_{S^{\setminus i}}(x). \tag{S9}$$

The general idea of algorithmic stability is that if the results of an algorithm do not depend too heavily on any one element of the training set, the algorithm will be able to generalize. One way to quantify the dependence of an algorithm on the training set is to examine how the results change when the training set is perturbed, for example by removing an element from the training set. The following definition of pointwise hypothesis stability, taken from Bousquet and Elisseeff (2002), states that an algorithm has pointwise hypothesis stability if, on expectation, the results of the algorithm do not change too much when an element of the training set is removed.

Definition S1 (Bousquet and Elisseeff, 2002)**.** *An algorithm has pointwise hypothesis stability η with respect to the loss function ℓ if the following holds*

$$\forall i \in \{1, \ldots, m\}, \quad \mathbb{E}_S\left[|\ell(f_S, x^i) - \ell(f_{S^{\setminus i}}, x^i)|\right] \leq \eta. \tag{S10}$$

The algorithm is said to be stable if η scales with $\frac{1}{m}$.

In our theorem, we suppose that all of the data belong to the same class of "relevant" items. The framework of Bousquet and Elisseeff (2002) can easily be adapted to the single-class setting, for example by framing it as a regression problem where all of the data points have the identical "true" output value 1. The following theorem comes from Bousquet and Elisseeff (2002), with the notation adapted to our setting.

Theorem S1 (Bousquet and Elisseeff, 2002)**.** *If an algorithm has pointwise hypothesis stability η with respect to a loss function ℓ such that $0 \leq \ell(\cdot,\cdot) \leq 1$, we have with probability at least $1-\delta$,*

$$\mathbb{E}_x\left[\ell(f_S, x)\right] \leq \frac{1}{m}\sum_{i=1}^{m}\ell(f_S, x^i) + \sqrt{\frac{1+12m\eta}{2m\delta}}. \tag{S11}$$

We now show that Bayesian Sets satisfies the conditions of Definition S1, and determine the corresponding η. The proof of Theorem 1 comes from inserting our findings for η into Theorem S1. We begin with a lemma providing a bound on the central moments of a Binomial random variable.

Lemma S2. *Let $t \sim$ Binomial(m,p) and let $\mu_k = \mathbb{E}\left[(t-\mathbb{E}[t])^k\right]$ be the k^{th} central moment. For integer $k \geq 1$, μ_{2k} and μ_{2k+1} are $O\left(m^k\right)$.*

Proof. We will use induction. For $k=1$, the central moments are well known (*e.g.*, Johnson et al, 2005): $\mu_2 = mp(1-p)$ and $\mu_3 = mp(1-p)(1-2p)$, which are both $O(m)$. We rely on the following recursion formula (Johnson et al, 2005; Romanovsky, 1923):

$$\mu_{s+1} = p(1-p)\left(\frac{d\mu_s}{dp} + ms\mu_{s-1}\right). \tag{S12}$$

Because μ_2 and μ_3 are polynomials in p, their derivatives will also be polynomials in p. This recursion makes it clear that for all s, μ_s is a polynomial in p whose coefficients include terms involving m.

For the inductive step, suppose that the result holds for $k=s$. That is, μ_{2s} and μ_{2s+1} are $O(m^s)$. Then, by (S12),

$$\mu_{2(s+1)} = p(1-p)\left(\frac{d\mu_{2s+1}}{dp} + (2s+1)m\mu_{2s}\right). \tag{S13}$$

Differentiating μ_{2s+1} with respect to p yields a term that is $O(m^s)$. The term $(2s+1)m\mu_{2s}$ is $O(m^{s+1})$, and thus $\mu_{2(s+1)}$ is $O(m^{s+1})$. Also,

$$\mu_{2(s+1)+1} = p(1-p)\left(\frac{d\mu_{2(s+1)}}{dp} + 2(s+1)m\mu_{2s+1}\right). \tag{S14}$$

Here $\frac{d\mu_{2(s+1)}}{dp}$ is $O(m^{s+1})$ and $2(s+1)m\mu_{2s+1}$ is $O(m^{s+1})$, and thus $\mu_{2(s+1)+1}$ is $O(m^{s+1})$.

This shows that if the result holds for $k=s$ then it must also hold for $k=s+1$ which completes the proof. □

The next lemma provides a stable, $O\left(\frac{1}{m}\right)$, bound on the expected value of an important function of a binomial random variable.

Lemma S3. *For $t \sim$ Binomial(m,p) and $\alpha > 0$,*

$$\mathbb{E}\left[\frac{1}{\alpha+t}\right] = \frac{1}{\alpha+mp} + O\left(\frac{1}{m^2}\right). \tag{S15}$$

Proof. We expand $\frac{1}{\alpha+t}$ at $t = mp$:

$$\begin{aligned}
\mathbb{E}\left[\frac{1}{\alpha+t}\right] &= \mathbb{E}\left[\sum_{i=0}^{\infty}(-1)^i \frac{(t-mp)^i}{(\alpha+mp)^{i+1}}\right] \\
&= \sum_{i=0}^{\infty}(-1)^i \frac{\mathbb{E}\left[(t-mp)^i\right]}{(\alpha+mp)^{i+1}} \\
&= \frac{1}{\alpha+mp} + \sum_{i=2}^{\infty}(-1)^i \frac{\mu_i}{(\alpha+mp)^{i+1}}
\end{aligned} \tag{S16}$$

where μ_i is the i^{th} central moment and we recognize that $\mu_1 = 0$. By Lemma S2,

$$\frac{\mu_i}{(\alpha+mp)^{i+1}} = \frac{O\left(m^{\lfloor \frac{i}{2} \rfloor}\right)}{O\left(m^{i+1}\right)} = O\left(m^{\lfloor \frac{i}{2} \rfloor - i - 1}\right). \tag{S17}$$

The alternating sum in (S16) can be split into two sums:

$$\sum_{i=2}^{\infty}(-1)^i \frac{\mu_i}{(\alpha+mp)^{i+1}} = \sum_{i=2}^{\infty} O\left(m^{\lfloor \frac{i}{2} \rfloor - i - 1}\right) = \sum_{i=2}^{\infty} O\left(\frac{1}{m^i}\right) + \sum_{i=3}^{\infty} O\left(\frac{1}{m^i}\right). \tag{S18}$$

These are, for m large enough, bounded by a geometric series that converges to $O\left(\frac{1}{m^2}\right)$. □

The following three lemmas provide results that will be useful for proving the main lemma, Lemma S7.

Lemma S4. *For all $\alpha > 0$,*

$$g(\alpha, m) := \frac{\log\left(\frac{\alpha+m}{\alpha}\right)}{\log\left(\frac{\alpha+m-1}{\alpha}\right)} \tag{S19}$$

is monotonically non-decreasing in α for any fixed $m \geq 2$.

Proof. Define $a = \frac{m-1}{\alpha}$ and $b = \frac{m}{m-1}$. Observe that $a \geq 0$ and $b \geq 1$, and that for fixed m, a is inversely proportional to α. We reparameterize (S19) to

$$g(a, b) := \frac{\log(ab+1)}{\log(a+1)}. \tag{S20}$$

To prove the lemma, it is sufficient to show that $g(a, b)$ is monotonically non-increasing in a for any fixed $b \geq 1$. Well,

$$\frac{\partial g(a,b)}{\partial a} = \frac{\frac{b}{ab+1}\log(a+1) - \frac{1}{a+1}\log(ab+1)}{(\log(a+1))^2},$$

so $\frac{\partial g(a,b)}{\partial a} \leq 0$ if and only if

$$h(a, b) := (ab+1)\log(ab+1) - b(a+1)\log(a+1) \geq 0. \tag{S21}$$

$h(a, 1) = (a+1)\log(a+1) - (a+1)\log(a+1) = 0$, and,

$$\begin{aligned}\frac{\partial h(a,b)}{\partial b} &= a\log(ab+1) + a - (a+1)\log(a+1)\\ &= a(\log(ab+1) - \log(a+1)) + (a - \log(a+1))\\ &\geq 0 \quad \forall a \geq 0,\end{aligned}$$

because $b \geq 1$ and $a \geq \log(1+a)$ $\forall a \geq 0$. This shows that (S21) holds $\forall a \geq 0, b \geq 1$, which proves the lemma. □

Lemma S5. *For any $m \geq 2$, $t \in [0, m-1]$, $\alpha > 0$, and $\gamma_{\min} \in (0, \alpha]$,*

$$\frac{1}{Z(m)}\log\frac{\alpha+t+1}{\alpha} \geq \frac{1}{Z(m-1)}\log\frac{\alpha+t}{\alpha}. \tag{S22}$$

Proof. Denote

$$g(t; m, \alpha) := \frac{1}{Z(m)}\log\frac{\alpha+t+1}{\alpha} - \frac{1}{Z(m-1)}\log\frac{\alpha+t}{\alpha}. \tag{S23}$$

By Lemma S4 and $\gamma_{\min} \leq \alpha$, for any $\alpha > 0$ and for any $m \geq 2$,

$$\frac{\log\left(\frac{\alpha+m}{\alpha}\right)}{\log\left(\frac{\alpha+m-1}{\alpha}\right)} \geq \frac{\log\left(\frac{\gamma_{\min}+m}{\gamma_{\min}}\right)}{\log\left(\frac{\gamma_{\min}+m-1}{\gamma_{\min}}\right)} = \frac{Z(m)}{Z(m-1)}.$$

Thus,

$$\frac{\log\left(\frac{\alpha+m}{\alpha}\right)}{Z(m)} \geq \frac{\log\left(\frac{\alpha+m-1}{\alpha}\right)}{Z(m-1)}, \tag{S24}$$

which shows

$$g(m-1; m, \alpha) = \frac{1}{Z(m)}\log\frac{\alpha+m}{\alpha} - \frac{1}{Z(m-1)}\log\frac{\alpha+m-1}{\alpha} \geq 0. \tag{S25}$$

Furthermore, because $Z(m) > Z(m-1)$,

$$\frac{\partial g(t; m, \alpha)}{\partial t} = \frac{1}{Z(m)}\frac{1}{\alpha+t+1} - \frac{1}{Z(m-1)}\frac{1}{\alpha+t} < 0, \tag{S26}$$

for all $t \geq 0$. Equations S25 and S26 together show that $g(t; m, \alpha) \geq 0$ for all $t \in [0, m-1], m \geq 2$, proving the lemma. □

Lemma S6. *For any $m \geq 2$, $t \in [0, m-1]$, $\beta > 0$, and $\gamma_{\min} \in (0, \beta]$,*

$$\frac{1}{Z(m)} \log \frac{\beta + m - t}{\beta} \geq \frac{1}{Z(m-1)} \log \frac{\beta + m - 1 - t}{\beta}. \tag{S27}$$

Proof. Let $\tilde{t} = m - t - 1$. Then, $\tilde{t} \in [0, m-1]$ and by Lemma S5, replacing α with β,

$$\frac{1}{Z(m)} \log \frac{\beta + \tilde{t} + 1}{\beta} \geq \frac{1}{Z(m-1)} \log \frac{\beta + \tilde{t}}{\beta}. \tag{S28}$$

□

The next lemma is the key lemma that shows Bayesian Sets satisfies pointwise hypothesis stability, allowing us to apply Theorem S1.

Lemma S7. *The Bayesian Sets algorithm satisfies the conditions for pointwise hypothesis stability with*

$$\eta = \frac{1}{\log\left(\frac{\gamma_{\min}+m-1}{\gamma_{\min}}\right)(\gamma_{\min} + (m-1)p_{\min})} + O\left(\frac{1}{m^2 \log m}\right). \tag{S29}$$

Proof.

$$\begin{aligned}
&\mathbb{E}_S|\ell(f_S, x^i) - \ell(f_{S^{\setminus i}}, x^i)| \\
&\quad = \mathbb{E}_S \left| f_{S^{\setminus i}}(x^i) - f_S(x^i) \right| \\
&\quad = \mathbb{E}_S \left| \frac{1}{Z(m-1)} \sum_{j=1}^{N} \left[x_j^i \log \frac{\alpha_j + \sum_{s \neq i} x_j^s}{\alpha_j} + (1 - x_j^i) \log \frac{\beta_j + (m-1) - \sum_{s \neq i} x_j^s}{\beta_j} \right] \right. \\
&\qquad \left. - \frac{1}{Z(m)} \sum_{j=1}^{N} \left[x_j^i \log \frac{\alpha_j + \sum_{s=1}^m x_j^s}{\alpha_j} + (1 - x_j^i) \log \frac{\beta_j + m - \sum_{s=1}^m x_j^s}{\beta_j} \right] \right| \\
&\quad \leq \mathbb{E}_S \sum_{j=1}^{N} x_j^i \left| \frac{1}{Z(m-1)} \log \frac{\alpha_j + \sum_{s \neq i} x_j^s}{\alpha_j} - \frac{1}{Z(m)} \log \frac{\alpha_j + \sum_{s=1}^m x_j^s}{\alpha_j} \right| \\
&\qquad + (1 - x_j^i) \left| \frac{1}{Z(m-1)} \log \frac{\beta_j + (m-1) - \sum_{s \neq i} x_j^s}{\beta_j} - \frac{1}{Z(m)} \log \frac{\beta_j + m - \sum_{s=1}^m x_j^s}{\beta_j} \right| \qquad \text{(S30)} \\
&\quad := \mathbb{E}_S \sum_{j=1}^{N} x_j^i \text{term}_j^1 + (1 - x_j^i) \text{term}_j^2 \qquad \text{(S31)} \\
&\quad = \sum_{j=1}^{N} \mathbb{E}_{x_j^1, \ldots, x_j^m} \left[x_j^i \text{term}_j^1 + (1 - x_j^i) \text{term}_j^2 \right] \\
&\quad = \sum_{j=1}^{N} \mathbb{E}_{x_j^i} \left[\mathbb{E}_{x_j^{s \neq i} | x_j^i} \left[x_j^i \text{term}_j^1 \right] \right] + \mathbb{E}_{x_j^i} \left[\mathbb{E}_{x_j^{s \neq i} | x_j^i} \left[(1 - x_j^i) \text{term}_j^2 \right] \right] \\
&\quad = \sum_{j=1}^{N} \mathbb{E}_{x_j^i} \left[x_j^i \mathbb{E}_{x_j^{s \neq i} | x_j^i} \left[\text{term}_j^1 \right] \right] + \mathbb{E}_{x_j^i} \left[(1 - x_j^i) \mathbb{E}_{x_j^{s \neq i} | x_j^i} \left[\text{term}_j^2 \right] \right] \\
&\quad = \sum_{j=1}^{N} \mathbb{E}_{x_j^{s \neq i}} \left[\text{term}_j^1 | x_j^i = 1 \right] \mathbb{P}\left(x_j^i = 1 \right) + \mathbb{E}_{x_j^{s \neq i}} \left[\text{term}_j^2 | x_j^i = 0 \right] \mathbb{P}\left(x_j^i = 0 \right) \\
&\quad \leq \sum_{j=1}^{N} \max \left\{ \mathbb{E}_{x_j^{s \neq i}} \left[\text{term}_j^1 | x_j^i = 1 \right], \mathbb{E}_{x_j^{s \neq i}} \left[\text{term}_j^2 | x_j^i = 0 \right] \right\}, \qquad \text{(S32)}
\end{aligned}$$

where (S30) uses the triangle inequality, and in (S31) we define term_j^1 and term_j^2 for notational convenience. Now consider each term in (S32) separately,

$$\begin{aligned}\mathbb{E}_{x_j^{s\neq i}}\left[\text{term}_j^1|x_j^i=1\right] &= \mathbb{E}_{x_j^{s\neq i}}\left|\frac{1}{Z(m-1)}\log\frac{\alpha_j+\sum_{s\neq i}x_j^s}{\alpha_j}-\frac{1}{Z(m)}\log\frac{\alpha_j+\sum_{s\neq i}x_j^s+1}{\alpha_j}\right| \\ &= \mathbb{E}_{x_j^{s\neq i}}\left[\frac{1}{Z(m)}\log\frac{\alpha_j+\sum_{s\neq i}x_j^s+1}{\alpha_j}-\frac{1}{Z(m-1)}\log\frac{\alpha_j+\sum_{s\neq i}x_j^s}{\alpha_j}\right], \qquad \text{(S33)}\end{aligned}$$

where we have shown in Lemma S5 that this quantity is non-negative. Because $\{x^s\}$ are independent, $\{x_j^s\}$ are independent for fixed j. We can consider $\{x_j^s\}_{s\neq i}$ to be a collection of $m-1$ independent Bernoulli random variables with probability of success $p_j = \mathbb{P}_{x\sim\mathcal{D}}(x_j=1)$, the marginal distribution. Let $t=\sum_{s\neq i}x_j^s$, then $t\sim\text{Binomial}(m-1,p_j)$. Continuing (S33),

$$\begin{aligned}\mathbb{E}_{x_j^{s\neq i}}\left[\text{term}_j^1|x_j^i=1\right] &= \mathbb{E}_{t\sim\text{Bin}(m-1,p_j)}\left[\frac{1}{Z(m)}\log\frac{\alpha_j+t+1}{\alpha_j}-\frac{1}{Z(m-1)}\log\frac{\alpha_j+t}{\alpha_j}\right] \\ &\leq \frac{1}{Z(m-1)}\mathbb{E}_{t\sim\text{Bin}(m-1,p_j)}\left[\log\frac{\alpha_j+t+1}{\alpha_j+t}\right] \\ &= \frac{1}{Z(m-1)}\mathbb{E}_{t\sim\text{Bin}(m-1,p_j)}\left[\log\left(1+\frac{1}{\alpha_j+t}\right)\right] \\ &\leq \frac{1}{Z(m-1)}\log\left(1+\mathbb{E}_{t\sim\text{Bin}(m-1,p_j)}\left[\frac{1}{\alpha_j+t}\right]\right) \\ &= \frac{1}{Z(m-1)}\log\left(1+\frac{1}{\alpha_j+(m-1)p_j}+O\left(\frac{1}{m^2}\right)\right). \qquad \text{(S34)}\end{aligned}$$

The second line uses $Z(m)\geq Z(m-1)$, the fourth line uses Jensen's inequality, and the fifth line uses Lemma S3. Now we turn to the other term.

$$\begin{aligned}&\mathbb{E}_{x_j^{s\neq i}}\left[\text{term}_j^2|x_j^i=0\right] \\ &= \mathbb{E}_{x_j^{s\neq i}}\left|\frac{1}{Z(m-1)}\log\frac{\beta_j+(m-1)-\sum_{s\neq i}x_j^s}{\beta_j}-\frac{1}{Z(m)}\log\frac{\beta_j+m-\sum_{s\neq i}x_j^s}{\beta_j}\right| \\ &= \mathbb{E}_{x_j^{s\neq i}}\left[\frac{1}{Z(m)}\log\frac{\beta_j+m-\sum_{s\neq i}x_j^s}{\beta_j}-\frac{1}{Z(m-1)}\log\frac{\beta_j+(m-1)-\sum_{s\neq i}x_j^s}{\beta_j}\right]. \qquad \text{(S35)}\end{aligned}$$

We have shown in Lemma S6 that this quantity is non-negative. Let $q_j=1-p_j$. Let $t=m-1-\sum_{s\neq i}x_j^s$, then $t\sim\text{Binomial}(m-1,q_j)$. Continuing (S35):

$$\begin{aligned}\mathbb{E}_{x_j^{s\neq i}}\left[\text{term}_j^2|x_j^i=0\right] &\leq \frac{1}{Z(m-1)}\mathbb{E}_{t\sim\text{Bin}(m-1,q_j)}\left[\log\frac{\beta_j+t+1}{\beta_j+t}\right] \\ &\leq \frac{1}{Z(m-1)}\log\left(1+\frac{1}{\beta_j+(m-1)q_j}+O\left(\frac{1}{m^2}\right)\right). \qquad \text{(S36)}\end{aligned}$$

where the steps are as with (S34). We now take (S34) and (S36) and use them to continue (S32):

$$\begin{aligned}
&\mathbb{E}_S|\ell(f_S, x^i) - \ell(f_{S^{\setminus i}}, x^i)| \\
&\quad \le \sum_{j=1}^{N} \max\left\{ \frac{1}{Z(m-1)} \log\left(1 + \frac{1}{\alpha_j + (m-1)p_j} + O\left(\frac{1}{m^2}\right)\right), \right. \\
&\qquad\qquad \left. \frac{1}{Z(m-1)} \log\left(1 + \frac{1}{\beta_j + (m-1)q_j} + O\left(\frac{1}{m^2}\right)\right)\right\} \\
&\quad \le \sum_{j=1}^{N} \frac{1}{Z(m-1)} \log\left(1 + \frac{1}{\min\{\alpha_j, \beta_j\} + (m-1)\min\{p_j, q_j\}} + O\left(\frac{1}{m^2}\right)\right) \\
&\quad \le \frac{N}{Z(m-1)} \log\left(1 + \frac{1}{\gamma_{\min} + (m-1)p_{\min}} + O\left(\frac{1}{m^2}\right)\right) \\
&\quad := \eta. \qquad\qquad (S37)
\end{aligned}$$

Using the Taylor expansion of $\log(1+x)$,

$$\begin{aligned}
\eta &= \frac{N}{Z(m-1)} \left(\frac{1}{\gamma_{\min} + (m-1)p_{\min}} + O\left(\frac{1}{m^2}\right) - \frac{1}{2}\left(\frac{1}{\gamma_{\min} + (m-1)p_{\min}} + O\left(\frac{1}{m^2}\right)\right)^2 \right) \\
&= \frac{N}{Z(m-1)} \left(\frac{1}{\gamma_{\min} + (m-1)p_{\min}} + O\left(\frac{1}{m^2}\right)\right) \\
&= \frac{1}{\log\left(\frac{\gamma_{\min}+m-1}{\gamma_{\min}}\right)(\gamma_{\min} + (m-1)p_{\min})} + O\left(\frac{1}{m^2 \log m}\right). \qquad (S38)
\end{aligned}$$

□

The proof of Theorem 1 is now a straightforward application of Theorem S1 using the result of Lemma S7.

Proof of Theorem 1. By Lemma S7, we can apply Theorem S1 to see that with probability at least $1-\delta$ on the draw of S,

$$\begin{aligned}
\mathbb{E}_x\left[\ell(f_S, x)\right] &\le \frac{1}{m}\sum_{i=1}^{m} \ell(f_S, x^i) + \sqrt{\frac{1 + 12m\eta}{2m\delta}} \\
\mathbb{E}_x\left[1 - f_S(x)\right] &\le \frac{1}{m}\sum_{s=1}^{m} (1 - f_S(x^s)) + \sqrt{\frac{1 + 12m\eta}{2m\delta}} \\
\mathbb{E}_x\left[f_S(x)\right] &\ge \frac{1}{m}\sum_{s=1}^{m} f_S(x^s) - \sqrt{\frac{1 + 12m\eta}{2m\delta}} \\
&= \frac{1}{m}\sum_{s=1}^{m} f_S(x^s) \\
&\quad - \sqrt{\frac{1}{2m\delta} + \frac{6}{\delta \log\left(\frac{\gamma_{\min}+m-1}{\gamma_{\min}}\right)(\gamma_{\min} + (m-1)p_{\min})} + O\left(\frac{1}{\delta m^2 \log m}\right)}.
\end{aligned}$$

□

2.1 Comments on the effect of the prior on generalization.

The prior influences the generalization bound via the quantity

$$h(\gamma_{\min}, m, p_{\min}) := \log\left(\frac{\gamma_{\min} + m - 1}{\gamma_{\min}}\right)(\gamma_{\min} + (m-1)p_{\min}). \qquad (S39)$$

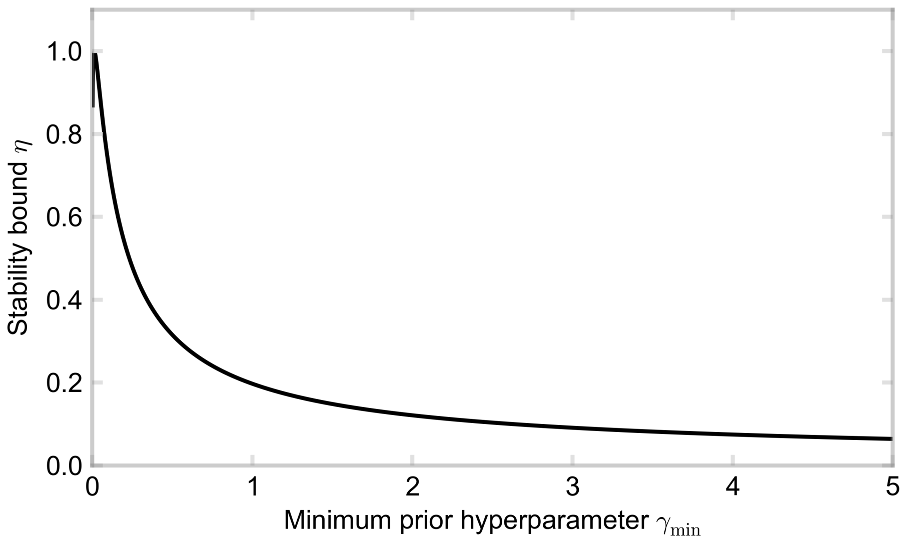

Figure S2: The stability bound η as a function of the prior $\gamma_{\min}$, for fixed $m = 100$ and $p_{\min} = 0.001$. For $\gamma_{\min}$ large enough relative to $p_{\min}$, stronger priors yield tighter bounds.

As this quantity increases, the bound becomes tighter. We can thus study the influence of the prior on generalization by studying the behavior of this quantity as $\gamma_{\min}$ varies. The second term, $(\gamma_{\min} + (m-1)p_{\min})$, is similar to many results from Bayesian analysis in which the prior plays the same role as additional data. This term is *increasing* with $\gamma_{\min}$, meaning it yields a tighter bound with a stronger prior. The first term, $\log\left(\frac{\gamma_{\min}+m-1}{\gamma_{\min}}\right)$, is inherited from the normalization $Z(m)$. This term is *decreasing* with $\gamma_{\min}$, that is, it gives a tighter bound with a weaker prior. The overall effect of $\gamma_{\min}$ on generalization depends on how these two terms balance each other, which in turn depends primarily on $p_{\min}$.

Exact analysis of the behavior of $h(\gamma_{\min}, m, p_{\min})$ as a function of $\gamma_{\min}$ does not yield interpretable results, however we gain some insight by considering the case where $\gamma_{\min}$ scales with m: $\gamma_{\min} := \tilde{\gamma}(m-1)$. Then we can consider (S39) as a function of $\tilde{\gamma}$ and $p_{\min}$ alone:

$$h(\tilde{\gamma}, p_{\min}) := \log\left(\frac{\tilde{\gamma}+1}{\tilde{\gamma}}\right)(\tilde{\gamma} + p_{\min}). \tag{S40}$$

The bound becomes tighter as $\tilde{\gamma}$ increases, as long as we have $\frac{\partial h(\tilde{\gamma}, p_{\min})}{\partial \tilde{\gamma}} > 0$. This is the case when

$$p_{\min} < \tilde{\gamma}(\tilde{\gamma}+1)\log\left(\frac{\tilde{\gamma}+1}{\tilde{\gamma}}\right) - \tilde{\gamma}. \tag{S41}$$

The quantity on the right-hand side is increasing with $\tilde{\gamma}$. Thus, for $p_{\min}$ small enough relative to $\tilde{\gamma}$, stronger priors lead to a tighter bound. To illustrate this behavior, in Figure S1 we plot the stability bound η (excluding $O\left(\frac{1}{m^2 \log m}\right)$ terms) as a function of $\gamma_{\min}$, for $m = 100$ and $p_{\min} = 0.001$. For $\gamma_{\min}$ larger than about 0.01, the bound tightens as the prior is increased.

2.2 Bayesian Sets and Uniform Stability.

In addition to pointwise hypothesis stability, Bousquet and Elisseeff (2002) define a stronger notion of stability called "uniform stability."

Definition S2 (Bousquet and Elisseeff, 2002)**.** *An algorithm has uniform stability κ with respect to the loss function ℓ if the following holds*

$$\forall S, \quad \forall i \in \{1, \dots, m\}, \quad ||\ell(f_S, \cdot) - \ell(f_{S^{\setminus i}}, \cdot)||_\infty \leq \kappa. \tag{S42}$$

The algorithm is said to be stable if κ scales with $\frac{1}{m}$.

Uniform stability requires a $O\left(\frac{1}{m}\right)$ bound for all training sets, rather than the average training set as with pointwise hypothesis stability. The bound must also hold for all possible test points, rather than testing on the perturbed point. Uniform stability is actually a very strong condition that is difficult to meet, since if (S42) can be violated by any possible combination of training set and test point, then uniform stability does not hold. Bayesian Sets does not have this form of stability, as we now show with an example.

Choose the training set of m data points to satisfy:

$$\begin{aligned} x_j^i &= 0 \quad \forall j, \quad i = 1, \dots, m-1 \\ x_j^m &= 1 \quad \forall j, \end{aligned}$$

and as a test point x, take $x_j = 1 \; \forall j$. Let x^m be the point removed from the training set. Then,

$$\begin{aligned} \kappa &= |\ell(f_S, x) - \ell(f_{S^{\setminus m}}, x)| \\ &= |f_{S^{\setminus m}}(x) - f_S(x)| \\ &= \left| \frac{1}{Z(m-1)} \sum_{j=1}^{N} x_j \log \frac{\alpha_j + \sum_{s=1}^m x_j^s - x_j^m}{\alpha_j} - \frac{1}{Z(m)} \sum_{j=1}^{N} x_j \log \frac{\alpha_j + \sum_{s=1}^m x_j^s}{\alpha_j} \right| \\ &= \left| \frac{1}{Z(m-1)} \sum_{j=1}^{N} \log \frac{\alpha_j}{\alpha_j} - \frac{1}{Z(m)} \sum_{j=1}^{N} \log \frac{\alpha_j + 1}{\alpha_j} \right| \\ &= \frac{1}{Z(m)} \sum_{j=1}^{N} \log \frac{\alpha_j + 1}{\alpha_j} \\ &\geq \frac{\log \frac{\max_j \alpha_j + 1}{\max_j \alpha_j}}{\log \left(\frac{\gamma_{\min} + m}{\gamma_{\min}} \right)}, \end{aligned} \tag{S43}$$

which scales with m as $\frac{1}{\log m}$, not the $\frac{1}{m}$ required for stability.

References

Bousquet O, Elisseeff A (2002) Stability and generalization. Journal of Machine Learning Research 2:499–526

Hoeffding W (1963) Probability inequalities for sums of bounded random variables. Journal of the American Statistical Association 58(301):13–30

Johnson NL, Kemp AW, Kotz S (2005) Univariate Discrete Distributions. John Wiley & Sons

Romanovsky V (1923) Note on the moments of a binomial $(p + q)^n$ about its mean. Biometrika 15:410–412