Approximate Computation of Exact Association Rules*

Saurabh Bansal$^{1[0000-0003-1297-2184]}$, Sriram Kailasam$^{1[0000-0002-2218-8660]}$, and Sergei Obiedkov$^{2[0000-0003-1497-4001]}$

[1] IIT Mandi, India
`saurabh18213@gmail.com, sriramk@iitmandi.ac.in`
[2] HSE University, Russia
`sergei.obj@gmail.com`

Abstract. We adapt a polynomial-time approximation algorithm for computing the canonical basis of implications to approximately compute frequent implications, also known as exact association rules. To this end, we define a suitable notion of approximation that takes into account the frequency of attribute subsets and show that our algorithm achieves a desired approximation with high probability. We experimentally evaluate the proposed algorithm on several artificial and real-world data sets.

Keywords: Association rule · Implication · PAC learning.

1 Introduction

Formal concept analysis offers several approaches to computing implication bases of a formal context. Most popular ones are based on Next Closure, a general algorithm for computing closed sets of a closure operator [11]. This algorithm is suitable for building the canonical basis of a formal context, since its premises, taken together with closed attribute sets of the context, form a closure system. The attribute-incremental algorithm from [19] is often faster than approaches based on Next Closure, but its memory requirements limit its applicability to contexts with a moderate number of closed sets.

Recently, probably approximately correct (PAC) algorithms for computing the implication basis have been considered [8,9]. In this paper, we continue this line of research by proposing a new notion of approximation that focuses on capturing the frequent (in the sense of frequent itemset mining) part of the implication theory behind the context. We adapt a previously proposed PAC-algorithm to compute this frequency-aware approximation and show that it usually produces more accurate approximations than the original PAC-algorithm and thus can be useful even if one is not interested specifically in frequent implications. We also show that the algorithm runs much faster than exact Next Closure-based algorithms on dense formal contexts, at least if the size their canonical

* Supported by SPARC, a Government of India Initiative under grant no. SPARC/2018-2019/P682/SL.

basis is significantly smaller than the number of closed sets (which is often the case, as suggested by our experiments in Section 5).

The paper is organized as follows. In Section 2, we present the necessary definitions from formal concept analysis. Section 3 recalls two notions of Horn approximations and a PAC-algorithm for finding them. In Section 4, we introduce frequency-aware approximations and adapt the algorithm from the previous section to compute them. Section 5 describes the results of empirical evaluation of the proposed algorithm.

2 Main Definitions

Recall that a *formal context* $\mathbb{K}$ is a triple (G, M, I), where G is a set of objects, M is a set of attributes, and $I \subseteq G \times M$ is an incidence relation specifying which objects have which attributes [13]. For $X \subseteq G$ and $Y \subseteq M$, the following *derivation operators* are defined:

$$X' = \{m \in M \mid \forall g \in X\colon (g, m) \in I\} \qquad Y' = \{g \in G \mid \forall m \in Y\colon (g, m) \in I\}$$

Two closure operators are defined by subsequent application of the two derivation operators; X'' and Y'' are said to be *closed* in $\mathbb{K}$. Closed subsets of M are called *(concept) intents* of $\mathbb{K}$. The set of all intents of $\mathbb{K}$ is denoted by $\operatorname{Int}\mathbb{K}$.

An *implication* is an expression of the form $A \to B$, where $A \subseteq M$ is called the *premise* and $B \subseteq M$ the *conclusion* of the implication. A subset $C \subseteq M$ is a *model* of $A \to B$ if $A \nsubseteq C$ or $B \subseteq C$. The implication $A \to B$ *holds* or *is valid* in $\mathbb{K}$ if, for every $g \in G$, the set $\{g\}'$ is its model. If $\{g\}'$ is not a model of an implication, then g is a *counterexample* to it.

The *support* of an attribute set $A \subseteq M$ is $|A'|$, the number of objects that have all attributes from A. The *relative support* of A is $|A'|/|G|$. The (*relative*) *support* or *frequency* of an implication $A \to B$ is the (relative) support of $A \cup B$. An attribute set or an implication is called *frequent* if its support is above a certain specified threshold. The support parameter is important in association rule mining, where the goal is to find implications with high support that may still have a small number of counterexamples in the context [1]. There, valid implications are known as *exact association rules*.

A set $C \subseteq M$ is a *model* of a set $\mathcal{L}$ of implications over M if it is a model of every implication from $\mathcal{L}$. The set of all models of $\mathcal{L}$ is denoted by $\operatorname{Mod}\mathcal{L}$.

An implication set $\mathcal{L}$ over M defines a closure operator that maps $C \subseteq M$ to the smallest subset $\mathcal{L}(C) \in \operatorname{Mod}\mathcal{L}$ such that $C \subseteq \mathcal{L}(C)$. If C is a model of $\mathcal{L}$, then $C = \mathcal{L}(C)$ and we say that C *is closed* under $\mathcal{L}$.

An implication $A \to B$ *follows* from an implication set $\mathcal{L}$ if every model of $\mathcal{L}$ is a model of $A \to B$ or, equivalently, $B \subseteq \mathcal{L}(A)$. An implication set $\mathcal{L}$ is *non-redundant* if no implication $A \to B \in \mathcal{L}$ follows from $\mathcal{L} \setminus \{A \to B\}$.

A non-redundant set $\mathcal{L}$ of implications valid in $\mathbb{K}$ is called a *basis* of $\mathbb{K}$ if every implication valid in $\mathbb{K}$ follows from $\mathcal{L}$. A formal context can have several bases of different sizes. The *canonical* or *Duquenne–Guigues basis* of $\mathbb{K}$ is known to contain the smallest number of implications among all bases of $\mathbb{K}$ [14]. It is defined as $\{P \to P'' \mid P$ is a pseudo-intent of $\mathbb{K}\}$, where $P \subseteq M$ is called a *pseudo-intent* if $P \neq P''$ and, for every $Q \subsetneq P$, we have $Q'' \subseteq P$ whenever Q is a pseudo-intent.

3 Probably Approximately Correct Computation of Implications

Being able to compute the canonical basis $\mathcal{L}$ of a formal context $\mathbb{K}$ in total polynomial time, i.e., time polynomial in the sizes of $\mathbb{K}$ and $\mathcal{L}$, is a major open problem. Known algorithms that compute $\mathcal{L}$ directly also compute $\operatorname{Int}\mathbb{K}$ as a side product [11,19,5,15]. However, $|\operatorname{Int}\mathbb{K}|$ can be exponentially larger than $|\mathcal{L}|$, see Example 1 from Section 5.3.

This motivates approximation algorithm design for finding the canonical basis. Probably approximately correct computation of the canonical basis of a formal context has been considered in various settings in [8,20,9,21]. The settings differ in whether the context is available directly or via a particular set of queries, as in the query learning framework [2] or in attribute exploration [12].

The notions of approximation that we will use here are more general than in these works. Assuming a probability distribution $\mathcal{D}$ over attribute subsets of M, we define the *Horn $\mathcal{D}$-distance* between an implication set $\mathcal{L}$ over M and a context $\mathbb{K} = (G, M, I)$ as the probability of obtaining a subset closed under $\mathcal{L}$ but not in $\mathbb{K}$ or vice versa when choosing it according to $\mathcal{D}$:

$$\operatorname{dist}^{\mathcal{D}}(\mathcal{L}, \mathbb{K}) := \Pr_{\mathcal{D}}(A \in \operatorname{Mod}\mathcal{L} \triangle \operatorname{Int}\mathbb{K}).$$

Here, $A \triangle B$ is the symmetric difference between the sets A and B. An *ϵ-Horn $\mathcal{D}$-approximation* of $\mathbb{K}$, where $0 < \epsilon < 1$, is an implication set $\mathcal{L}$ over M such that $\operatorname{dist}^{\mathcal{D}}(\mathcal{L}, \mathbb{K}) \leq \epsilon$. If $\mathcal{D}$ is the uniform distribution, then ϵ-Horn $\mathcal{D}$-approximation of $\mathbb{K}$ is the ϵ-Horn approximation of $\mathbb{K}$ as defined in the papers referenced above.

These papers also use the notion of an ϵ-strong Horn approximation, which we generalize in a similar way. The *strong Horn $\mathcal{D}$-distance* between $\mathcal{L}$ and $\mathbb{K}$ is the probability of choosing a subset with different closures under $\mathcal{L}$ and in $\mathbb{K}$:

$$\operatorname{dist}^{\mathcal{D}}_{\text{STRONG}}(\mathcal{L}, \mathbb{K}) := \Pr_{\mathcal{D}}(\mathcal{L}(A) \neq A'').$$

$\mathcal{L}$ is an *ϵ-strong Horn $\mathcal{D}$-approximation* of $\mathbb{K}$ if $\operatorname{dist}^{\mathcal{D}}_{\text{STRONG}}(\mathcal{L}, \mathbb{K}) \leq \epsilon$. An ϵ-strong Horn $\mathcal{D}$-approximation is always an ϵ-Horn $\mathcal{D}$-approximation.

An approximation $\mathcal{L}$ is an *upper approximation* of $\mathbb{K}$ if all implications of $\mathcal{L}$ are valid in $\mathbb{K}$, or, equivalently, $\mathcal{L}(A) \subseteq A''$ for all $A \subseteq M$, i.e., if $\operatorname{Int}\mathbb{K} \subseteq \operatorname{Mod}\mathcal{L}$.

We are interested in algorithms that, given a formal context $\mathbb{K} = (G, M, I)$, a distribution $\mathcal{D}$ over subsets of M, and parameters $0 < \epsilon \leq 1$ and $0 < \delta \leq 1$, compute, with probability at least $1 - \delta$, an ϵ- or ϵ-strong Horn $\mathcal{D}$-approximation of $\mathbb{K}$. As discussed in [20], such algorithms exist for the uniform distribution and they work in time polynomial in $|G|$, $|M|$, the size of the canonical basis of $\mathbb{K}$, $1/\epsilon$, and $1/\delta$. Such an algorithm for an upper ϵ-Horn approximation has been presented (in a different setting) already in [16] based on the results for learning Horn formulas with membership and equivalence queries [3]. We present its generalization to an arbitrary distribution as Algorithm 1.

Algorithm 1 HornApproximation($\mathbb{K}$, $EX_{\mathcal{D}}$, ϵ, δ)

Input: A formal context $\mathbb{K} = (G, M, I)$, a sampling oracle $EX_{\mathcal{D}}$ that returns a subset of M according to distribution $\mathcal{D}$, $0 < \epsilon \leq 1$, and $0 < \delta \leq 1$.

Output: A set of implications $\mathcal{L}$ that, with probability at least $1 - \delta$, is an ϵ-Horn $\mathcal{D}$-approximation of $\mathbb{K}$.

```
1:  $\mathcal{L} := []$
2:  $i := 1$
3:  while IsApproximatelyEquivalent($\mathcal{L}$, $\mathbb{K}$, $EX_{\mathcal{D}}$, $q_i(\epsilon, \delta)$) returns $X$ do
4:      $found$ := false
5:      for all $A \to B \in \mathcal{L}$ do
6:          $C := A \cap X$
7:          if $A \neq C \neq C''$ then
8:              $found$ := true
9:              replace $A \to B$ by $C \to C''$ in $\mathcal{L}$
10:             exit for
11:     if not $found$ then
12:         add $X \to X''$ to the end of $\mathcal{L}$
13:     $i := i + 1$
14: return $\mathcal{L}$
```

Algorithm 1 receives a sampling oracle $EX_{\mathcal{D}}$ that takes no arguments and, when called, returns a subset of M according to distribution $\mathcal{D}$. Starting with an empty list $\mathcal{L}$ of implications, the algorithm repeatedly calls procedure IsApproximatelyEquivalent to check if $\mathcal{L}$ is an ϵ-Horn $\mathcal{D}$-approximation of $\mathbb{K}$. If not, this procedure is expected to return a model X of $\mathcal{L}$ such that $X \neq X''$, which means that there is an implication valid in $\mathbb{K}$ that does not follow from $\mathcal{L}$. This X is called a *negative counterexample* to $\mathcal{L}$, as opposed to a *positive counterexample* Y, which is such that $\mathcal{L}(Y) \neq Y = Y''$. The algorithm then either refines one of the implications of $\mathcal{L}$ or simply adds implication $X \to X''$ so as to ensure that $X \subsetneq \mathcal{L}(X) \subseteq X''$. This guarantees that $\mathcal{L}$ always contains only valid implications of $\mathbb{K}$ and thus no positive counterexamples to $\mathcal{L}$ are possible.

If the IsApproximatelyEquivalent procedure always returns a negative counterexample X when it exists, then Algorithm 1 computes the canonical basis of $\mathbb{K}$. This easily follows from the results presented in [4] regarding the original query-based algorithm from [3]. In this case, the IsApproximatelyEquivalent procedure implements an equivalence oracle in the sense of [2,3]. However, finding such an X given $\mathbb{K}$ and $\mathcal{L}$ is the problem referred to as CMI in [17], where it is shown that it is at least as hard as (the decision version of) the Hypergraph Transversal Problem; no polynomial-time algorithm is known to exist for it.

To achieve an ϵ-Horn approximation with probability at least $1-\delta$, we use Algorithm 2, which makes a certain number of attempts to generate such an X using $EX_{\mathcal{D}}$ and returns **true** if all attempts fail.

Algorithm 2 IsApproximatelyEquivalent($\mathcal{L}$, $\mathbb{K}$, $EX_{\mathcal{D}}$, k)

```
Input: A set $\mathcal{L}$ of implications valid in context $\mathbb{K} = (G, M, I)$, a sampling oracle $EX_{\mathcal{D}}$
    that returns a subset of $M$ according to distribution $\mathcal{D}$, and $k \in \mathbb{N}$.
Output: A set $X \subseteq M$ such that $\mathcal{L}(X) = X \neq X''$ if found; true, otherwise.
1: for $j := 1$ to $k$ do
2:     $X := EX_{\mathcal{D}}()$
3:     if $\mathcal{L}(X) = X \neq X''$ then
4:         return $X$
5: return true
```

How many attempts are needed depends on how far we are in computing a Horn approximation or, more precisely, how many counterexamples we have already produced. In Algorithm 1, we use the function $q_i(\epsilon, \delta)$ to determine the number of calls to $EX_{\mathcal{D}}$ needed to generate the ith X. It has been known that an algorithm using equivalence queries can be transformed into a PAC algorithm for the same learning problem by replacing the ith equivalence query by

$$\left\lceil \frac{1}{\epsilon}\left(\ln \frac{1}{\delta} + i \ln 2\right) \right\rceil \tag{1}$$

calls to the $EX_{\mathcal{D}}$ oracle and terminating if none of them returns a counterexample [2]. Here, ϵ is the desired approximation quality and δ is the upper bound on the probability of failing to achieve an ϵ-approximation.

The quantity (1) grows linearly with i; however, a logarithmic dependence on i is sufficient, as shown in [21]. Defining the function q_i as

$$q_i(\epsilon, \delta) = \left\lceil \log_{1-\epsilon} \frac{\delta}{i(i+1)} \right\rceil, \tag{2}$$

we guarantee that Algorithm 1 computes an ϵ-Horn $\mathcal{D}$-approximation of $\mathbb{K}$ with probability at least $1-\delta$ in time polynomial in $|G|$, $|M|$, $|\mathcal{L}|$, $1/\epsilon$, and $1/\delta$, where $\mathcal{L}$ is the set of implications upon the termination of the algorithm, whose size never exceeds the size of the canonical basis of $\mathbb{K}$. The total number of calls to $EX_{\mathcal{D}}$ is $O(|\mathcal{L}||M|(\log|\mathcal{L}||M| + \log 1/\delta)/\epsilon)$ when using (2) compared with $O(|\mathcal{L}||M|(|\mathcal{L}||M| + \log 1/\delta)/\epsilon)$ when using (1) [21].

To obtain an ϵ-strong Horn $\mathcal{D}$-approximation, we replace the call to IsApproximatelyEquivalent in Algorithm 1 to the call to the IsStronglyApproximatelyEquivalent procedure presented as Algorithm 3 [9].

4 Computing Frequency-Aware Approximations

For practical applications, it may be reasonable to assume that attribute subsets are distributed according to their frequency. For a context $\mathbb{K} = (G, M, I)$ with

Algorithm 3 IsStronglyApproximatelyEquivalent($\mathcal{L}$, $\mathbb{K}$, $EX_{\mathcal{D}}$, k)

Input: A set $\mathcal{L}$ of implications valid in context $\mathbb{K} = (G, M, I)$, a sampling oracle $EX_{\mathcal{D}}$ that returns a subset of M according to distribution $\mathcal{D}$, and $k \in \mathbb{N}$.
Output: A set $\mathcal{L}(X) \subseteq M$ such that $\mathcal{L}(X) \neq X''$ if found; **true**, otherwise.

```
1: for $j := 1$ to $k$ do
2:     $Y := \mathcal{L}(EX_{\mathcal{D}}())$
3:     if $Y \neq Y''$ then
4:         return $Y$
5: return true
```

finite G and $A \subseteq M$, this means

$$\Pr(A) = \frac{|A'|}{\sum_{B \subseteq M} |B'|}. \tag{3}$$

Plugging this probability into the definition of (strong) Horn distance, we obtain the notion of frequency-aware Horn approximation: an ϵ- (ϵ-strong) Horn $\mathcal{D}$-approximation $\mathcal{L}$ of $\mathbb{K}$ is a *frequency-aware ϵ- (ϵ-strong) Horn approximation* of $\mathbb{K}$ if $\mathcal{D}$ is the probability distribution defined by (3).

The reason for using a frequency-aware approximation is two-fold. On the one hand, such approximation $\mathcal{L}$ ensures that most implications with high support follow from $\mathcal{L}$ and, in the case of the strong approximation, the closures of most frequent subsets under $\mathcal{L}$ coincide with their closures in the context. In the framework of association rule mining [1], such implications and such subsets are usually considered the most important. On the other hand, a frequency-aware approximation ignores attribute subsets that never occur in data. For real-world data sets, these may be the bulk of all the subsets, resulting in a misleadingly low value of $\text{dist}^{\mathcal{D}}(\mathcal{L}, \mathbb{K})$ when $\mathcal{D}$ is the uniform distribution. Using frequency-aware approximation allows one to capture the implications that describe dependencies inside attribute combinations that actually occur in data instead of focusing on implications that describe incompatibilities between attributes (which may also be important, but, in many cases, are a part of background knowledge).

To compute such frequency-aware approximations, we need to simulate a sampling oracle that samples attribute subsets according to (3). This oracle can be simulated by polynomial-time Algorithm 1 "Frequency-based Sampling" from [7], resulting in a total–polynomial time PAC algorithm for computing frequent implications. The algorithm uses the following probability distribution on objects $g \in G$ of context $\mathbb{K} = (G, M, I)$:

$$\Pr(g) = \frac{2^{|\{g\}'|}}{\sum_{h \in G} 2^{|\{h\}'|}}. \tag{4}$$

In other words, the probability of an object $g \in G$ is proportional to the number of subsets of its intent $\{g'\}$.

The algorithm consists of two steps. First, it selects an object g from G according to probability distribution (4), and then it selects a subset of $\{g\}'$

uniformly at random. It is shown in [7], that this algorithm generates an attribute subset according to probability distribution (3).

Therefore, a frequency-aware ϵ- or ϵ-strong Horn approximation can be computed by Algorithm 1 by passing the algorithm just described in place of $EX_{\mathcal{D}}$.

5 Experimental Evaluation

In this section, we study the performance of the randomized algorithm on real-life, as well as artificial data sets. We are primarily interested in two characteristics: the runtime and the quality of approximation.

5.1 Quality Factor

The randomized algorithms presented here are guaranteed to produce an upper ϵ- or ϵ-strong Horn approximation with the desired probability. In particular, if $\mathcal{L}$ is the implication set obtained from $\mathbb{K}$ when running the algorithm for ϵ-approximation with parameters ϵ and δ and a sampling oracle for probability distribution $\mathcal{D}$, then, with probability at least $1-\delta$, we have $\operatorname{dist}^{\mathcal{D}}(\mathcal{L},\mathbb{K}) \leq \epsilon$. Since $\mathcal{L}$ contains only implications valid in $\mathbb{K}$ and, consequently, $\operatorname{Int}\mathbb{K} \subseteq \operatorname{Mod}\mathcal{L}$, this means that

$$\frac{|\operatorname{Mod}\mathcal{L}| - |\operatorname{Int}\mathbb{K}|}{2^{|M|}} \leq \epsilon$$

when $\mathcal{D}$ is the uniform distribution. In other words, the difference between $|\operatorname{Mod}\mathcal{L}|$ and $|\operatorname{Int}\mathbb{K}|$ is small when considered on the scale of $2^{|M|}$: $\operatorname{Mod}\mathcal{L}$ contains at most $\epsilon 2^{|M|}$ extra subsets in addition to those in $\operatorname{Int}\mathbb{K}$.

However, if $\operatorname{Int}\mathbb{K}$ is small compared to $2^{|M|}$, this may still allow $\operatorname{Mod}\mathcal{L}$ to be several times larger than $\operatorname{Int}\mathbb{K}$. To see if this really happens in practice, we introduce the *quality factor* (QF) defined as follows:

$$QF(\mathcal{L},\mathbb{K},A) = \frac{|\operatorname{Int}\mathbb{K} \cap \mathfrak{P}(A)|}{|\operatorname{Mod}\mathcal{L} \cap \mathfrak{P}(A)|},$$

where $A \subseteq M$ and $\mathfrak{P}(A)$ is the power set of A. This measures the proportion of subsets of A closed in the context among those closed under the computed implications. When we report the quality factor in the experiments, we set A to be the set of roughly $\alpha|M|$ most frequent attributes of M, where α is $1/4$ for real-world data sets and $1/2$ for artificial data sets.

5.2 Testbed

The testbed consists of a server Intel Xeon E5-2650 v3 @ 2.30GHz with 20 cores and 40 threads.

5.3 Data Sets

The formal contexts used in the experiments are described in Table 1, where the last five columns correspond to the number of attributes, the number of objects, the size of the canonical basis, the number of intents, and the density, $|I|/|G||M|$, of the context (G, M, I) named in the first column.

Table 1. Contexts.

Context	Attributes	Objects	Canonical basis	Intents	Density
Census	122	48842	71787	248846	0.08
nom10shuttle	97	43500	810	2931	0.10
Mushroom	119	8124	2323	238710	0.19
Connect	114	7222	86583	50468988	0.38
inter10shuttle	178	43500	936	38199148	0.46
Chess	75	3196	73162	930851337	0.49
Example 1 ($n = 5$)	25	3125	5	28629152	0.80
Example 1 ($n = 6$)	36	46656	6	62523502210	0.83
Example 2 ($n = 10$)	21	30	1024	2038103	0.92
Example 2 ($n = 15$)	31	45	32768	2133134741	0.95

The first six data sets are real-world data sets, while the rest are synthetically generated. The real-world data sets have been derived from Census, Shuttle, Mushroom, Connect, and Chess data sets from the UCI machine learning repository [10]. They have been converted into formal contexts by using nominal scaling for categorical features (one attribute per category) and by scaling numerical features into multiple attributes using equidistant cut points. In the inter10shuttle data set, inter-ordinal rather than nominal scaling is used [15].

Example 1 is a context with n^n objects and n^2 attributes $M = M_1 \cup \cdots \cup M_n$ with $|M_i| = n, M_i \cap M_j = \varnothing$ for all $1 \le i < j \le n$, where the object intents $\{g\}'$ are all possible subsets of M such that $|\{g\}' \cap M_i| = n - 1$ for all $1 \le i \le n$ [12]. The canonical basis consists of only n implications of the type $M_i \to M$ for $1 \le i \le n$. The number of concept intents is $(2^n - 1)^n + 1$. This context is interesting, because the number of its closed attribute sets is exponential in $|M|$, while the size of the canonical basis is only linear in $|M|$. This is precisely the type of a context that should be hard for Next Closure–based algorithms, since they have to compute all closed sets as a side product, and much easier for our randomized algorithm. We ran experiments for $n = 5$ and $n = 6$.

Example 2 is a context with $3n$ objects $g_1, g_2, \ldots, g_{3n}$ and $2n + 1$ attributes $m_0, m_1, \ldots, m_{2n}$, where object g_i has attribute m_j if $i \le n$ and $j \notin \{0, i, i + n\}$, or if $i > n$ and $j \ne i - n$ [18]. There are exactly 2^n pseudo-intents of the form $\{m_{i_1}, m_{i_2}, \ldots, m_{i_n}\}$ where $i_j \in \{j, j + n\}$; thus, the size of the canonical basis is exponential in the context size. We ran experiments for $n = 10$ and $n = 15$.

5.4 Experiments

In all the experiments, a parallelized implementation of the randomized algorithm was used. We parallelized the search for a counterexample in Algorithms 2 and 3, as well as the search for an implication $A \to B$ to be refined in the main loop of Algorithm 1. Our implementation and the data sets used for the experiments are available at `https://github.com/saurabh18213/Implication-Basis`. Unless mentioned otherwise, forty threads were allocated to run the algorithm. The actual number of threads used at different points of the execution of the algorithm was determined using certain heuristics.

In Experiments 1, 3 and 4, we set $\epsilon = 0.1$ for real-world data sets, $\epsilon = 0.01$ for Example 1, $\epsilon = 0.01$ for Example 2 ($n = 10$), and $\epsilon = 0.001$ for Example 2 ($n = 15$). In Experiment 2, we vary the value of ϵ. All the reported results are for $\delta = 0.1$. No significant change in total time, the computed number of implications, and Quality Factor was observed when δ was varied. For real-world and artificial data sets, all the results are average of three and five measurements, respectively.

Experiment 1: Comparing approximations. In this experiment, we compute ϵ- and ϵ-strong Horn $\mathcal{D}$-approximations for different $\mathcal{D}$, varying the sampling oracle used in Algorithms 2 and 3. We use the *Uniform* oracle that generates subsets of M uniformly at random and the *Frequent* oracle that generates subsets according to the probability distribution specified by (3), as described in Section 4. In addition, we test the following combination of the two oracles. We first use the Uniform oracle. If, at some call to Algorithm 3, all k attempts to generate a counterexample with the Uniform oracle fail, instead of terminating the algorithm, we "redo" the k attempts, now with the Frequent oracle. If one of the attempts succeeds, we keep using the Frequent oracle for the remaining part of the computation; otherwise, the algorithm terminates. This approach is denoted by *Both* in the results below.

Table 2. Runtime in seconds for different types of approximation.

	ϵ-strong Horn approximation			ϵ-Horn approximation		
Data set	Uniform	Frequent	Both	Uniform	Frequent	Both
Census	0.18	1451.64	1184.10	0.16	5.02	0.21
nom10shuttle	0.15	0.73	0.71	0.14	0.43	0.44
Mushroom	0.11	1.89	1.95	0.06	0.16	0.14
Connect	0.14	307.51	307.10	0.07	0.08	0.07
inter10shuttle	0.59	6.77	6.47	0.58	0.60	0.60
Chess	0.07	167.96	169.77	0.04	0.04	0.03
Example 1 ($n = 5$)	0.03	0.03	0.04	0.03	0.03	0.04
Example 1 ($n = 6$)	0.31	0.27	0.36	0.31	0.29	0.37
Example 2 ($n = 10$)	0.27	0.17	0.27	0.21	0.19	0.26
Example 2 ($n = 15$)	96.72	74.64	108.77	83.31	75.12	115.81

Table 3. The number of implications computed for different types of approximation. The last column shows the number of implications in the entire canonical basis.

	ϵ-strong Horn approximation			ϵ-Horn approximation			Total
Data set	Uniform	Frequent	Both	Uniform	Frequent	Both	
Census	48	20882	19111	41	1210	71	71787
nom10shuttle	76	201	201	76	137	146	810
Mushroom	95	577	593	7	72	59	2323
Connect	120	10774	10730	7	9	9	86583
inter10shuttle	172	446	430	171	171	171	936
Chess	64	6514	6542	48	48	48	73162
Example 1 ($n = 5$)	5	0	5	5	0	5	5
Example 1 ($n = 6$)	6	0	6	6	0	6	6
Example 2 ($n = 10$)	357	269	340	321	262	347	1024
Example 2 ($n = 15$)	7993	6813	8375	7612	6970	8424	32768

Table 4. Quality Factor (QF) for different types of approximation.

	ϵ-strong Horn approximation			ϵ-Horn approximation		
Data set	Uniform	Frequent	Both	Uniform	Frequent	Both
Census	0.0003	0.0184	0.0180	0.0003	0.0014	0.0004
nom10shuttle	0.0004	0.0695	0.0613	0.0004	0.0157	0.0208
Mushroom	0.0004	0.1454	0.1482	0.0001	0.0032	0.0014
Connect	0.9979	0.9979	0.9979	0.0001	0.0016	0.0016
inter10shuttle	0.4900	0.5533	0.5429	0.4900	0.4900	0.4900
Chess	0.6927	1.0000	0.9830	0.6927	0.6927	0.6927
Example 1 ($n = 5$)	1.0000	0.9692	1.0000	1.0000	0.9692	1.0000
Example 1 ($n = 6$)	1.0000	0.9844	1.0000	1.0000	0.9844	1.0000
Example 2 ($n = 10$)	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Example 2 ($n = 15$)	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

The time taken, the number of implications computed, and the value of the Quality Factor for each data set are shown in Tables 2, 3, and 4, respectively.

When computing an ϵ-strong Horn approximation, the runtime on the real-world data sets is significantly higher with the Frequent oracle than with the Uniform oracle, but so is the number of implications computed and, usually, the Quality Factor. An exception is Connect, for which QF almost does not change despite a sharp increase in the number of implications. With ϵ-Horn approximation, the situation is generally similar.

The runtime, the number of implications, and QF are lower for Example 1 when using the Frequent oracle than when using the Uniform oracle. This is because all non-trivial valid implications (i.e., implications $A \to B$ with $B \not\subseteq A$) have zero support and are ignored by frequency-aware approximations. The randomized algorithm computes no implications when using the Frequency oracle.

In Example 2, all non-trivial valid implications have non-zero support. In particular, all 2^n implications from the canonical basis have a rather high support n (while the total number of objects in the context is $3n$), and their premises are also rather large (n out of $2n + 1$ attributes). Because of this, all three metrics are almost the same for the Uniform and Frequent oracles in Example 2: chances to generate a negative counterexample to a current implication set $\mathcal{L}$ are similar for the two oracles.

As expected, for real-world data sets, most of the metrics are higher in the case of ϵ-strong approximation than in the case of ϵ-approximation. For the artificial data sets, there is not much difference due to the fact that the closure of all non-closed sets there is M and, therefore, Algorithm 1 adds to $\mathcal{L}$ only implications of the form $X \to M$ whether it computes ϵ- or ϵ-strong approximation.

When using both oracles, as described above, we usually obtain results similar to what we get with the Frequent oracle alone. An important exception is Example 1, where all implications have zero support. In general, using both oracles lets us capture such zero-support implications in addition to frequent implications.

Experiment 2: The quality of approximation. In this experiment, we vary the value of the ϵ parameter. The results in Tables 5–7 are for ϵ-strong Horn approximation with counterexamples generated following the approach labeled as “Both” in the description of Experiment 1. As expected, for most data sets, the run time, the number of implications computed, and the value of the quality factor increase as the value of ϵ is decreased. The exceptions are Chess and Connect, where a very good approximation (QF ≈ 1) is computed even at $\epsilon = 0.3$. The decrease in ϵ has no substantial effect on any of the metrics, even though the number of implications computed is several times smaller than the size of the canonical basis. It seems that the implication set $\mathcal{L}$ computed by the randomized algorithm forms the essential part of the implication theory behind the context, while valid implications that do not follow from $\mathcal{L}$ must have limited applicability due to large premises with low support.

Table 5. Time in seconds for different values of ϵ.

Data set	0.3	0.2	0.1	0.05	0.01
Census	0.19	37.63	1184.10	2345.26	2336.88
nom10shuttle	0.44	0.47	0.71	0.82	1.43
Mushroom	0.82	1.27	1.95	2.75	5.03
Connect	308.69	307.54	307.10	306.97	307.44
inter10shuttle	4.41	5.34	6.47	7.91	12.72
Chess	169.23	169.50	169.77	168.04	168.99
Example 1 ($n = 5$)	0.02	0.02	0.03	0.03	0.04
Example 1 ($n = 6$)	0.23	0.23	0.29	0.30	0.36
Example 2 ($n = 10$)	0.002	0.002	0.002	0.01	0.27
Example 2 ($n = 15$)	0.002	0.002	0.002	0.002	0.63

Table 6. The number of implications computed for different values of ϵ. The last column shows the number of implications in the entire canonical basis.

Data set	0.3	0.2	0.1	0.05	0.01	Total
Census	49	2865	19111	26257	26253	71787
nom10shuttle	136	149	201	231	303	810
Mushroom	349	440	593	749	1036	2323
Connect	10790	10746	10730	10735	10759	86583
inter10shuttle	356	383	430	479	582	936
Chess	6563	6572	6542	6537	6578	73162
Example 1 ($n = 5$)	3	4	5	5	5	5
Example 1 ($n = 6$)	1	2	6	6	6	6
Example 2 ($n = 10$)	1	2	4	28	340	1024
Example 2 ($n = 15$)	0	0	0	1	422	32768

Table 7. Quality Factor (QF) for different values of ϵ.

Data set	0.3	0.2	0.1	0.05	0.01
Census	0.0004	0.0034	0.0180	0.0208	0.0208
nom10shuttle	0.0090	0.0140	0.0613	0.1017	0.1753
Mushroom	0.0382	0.0692	0.1482	0.2726	0.4504
Connect	0.9979	0.9979	0.9979	0.9979	0.9979
inter10shuttle	0.4956	0.5202	0.5429	0.6451	0.8910
Chess	0.9981	1.0000	0.9830	0.9963	1.0000
Example 1 ($n = 5$)	0.9692	0.9815	1.0000	1.0000	1.0000
Example 1 ($n = 6$)	0.9844	0.9875	0.9969	1.0000	1.0000
Example 2 ($n = 10$)	1.0000	1.0000	1.0000	1.0000	1.0000
Example 2 ($n = 15$)	1.0000	1.0000	1.0000	1.0000	1.0000

It should also be said that the Quality Factor as we compute it is not very relevant to Examples 1 and 2. Recall that, for artificial data sets, we select the $|M|/2$ most frequent attributes of M and then check how many subsets of these attributes are closed under the computed implications but not in the context. However, for Example 2, any selection of $|M|/2$ attributes contains at most one subset that is not closed in the context; so the value of the Quality Factor is bound to be high no matter what implications we compute. This explains why we have QF = 1 even for cases when no implications have been computed. The situation is similar for Example 1, though less dramatic. There, the number of implications computed is a better indicator of the approximation quality than QF. Tables 5 and 6 show that, for Example 1, we compute the basis exactly at $\epsilon \leq 0.05$ in a fraction of a second.

Experiment 3: The frequency of implications. In this experiment, we compute the support of implications in an ϵ-strong Horn approximation obtained with counterexamples generated using the *Frequent* oracle. The results are shown in Table 8. We show relative supports as percentages of $|G|$. In the second column, the average support of the implications is shown. For columns 3–6, if the value of P_x is y, then at least x% of the implications in the approximation have support greater than or equal to y. Example 1 is not shown because it has no implications with non-zero support.

Table 8. The relative support of the implications in the approximation.

Data set	Average	P_{10}	P_{50}	P_{90}	P_{99}	$\lvert G\rvert$
Census	0.0335	0.0409	0.0082	0.0020	0.0020	48842
nom10shuttle	5.4038	12.1456	1.2345	0.0023	0.0000	43500
Mushroom	11.7045	25.6360	7.9764	1.1489	0.5252	8124
Connect	0.9276	1.3385	0.2446	0.0415	0.0277	7222
inter10shuttle	35.7838	99.9831	11.7080	0.0092	0.0069	43500
Chess	6.7690	15.7697	4.0989	0.9387	0.5423	3196
Example 2 ($n = 10$)	32.2586	33.3333	33.3333	30.0000	26.6667	30
Example 2 ($n = 15$)	32.6944	33.3333	33.3333	31.1111	28.8889	45

Experiment 4: Runtime. In this experiment, we compare the performance of the randomized algorithm for computing an ϵ-strong approximation with that of two algorithms computing the canonical basis exactly: the optimized version of Next Closure from [5] (referred to as "6 + 1" there) and LinCbO from [15], which combines this optimized version with the LinClosure algorithm for computing the closures under implications [6] and introduces further optimizations. The randomized algorithm is run with the values of ϵ and δ specified in the beginning of Section 5.4. Counterexamples were generated following the approach labeled as "Both" in the description of Experiment 1.

We show the results in Table 9. To make comparison fair, we give the runtime of both the parallel version of the randomized algorithm with forty threads and the version with one thread only. The number of threads does not affect the value of the quality factor, which is therefore shown only once. The exact algorithms have taken excessive time on Example 1 ($n = 6$); so we had to terminate them before the basis was computed.

Between the two exact algorithms, LinCbO is consistently faster. The randomized algorithm, both with forty threads and one thread, runs faster than the exact algorithms on all contexts except the two sparse contexts (Census and nom10shuttle) and the medium-density context (Mushroom). For Mushroom context, the randomized algorithm runs faster than the exact algorithms, when forty threads are used. On Mushroom context, the value of the quality factor of the implication set obtained by the randomized algorithm is rather low, although higher than for the two sparse contexts. For the dense contexts, the value of the quality factor is close to 1 except for inter10shuttle, where it is around 0.54.

Table 9. Runtime in seconds (Experiment 4)

Data set	1 thread	40 threads	QF	Next Closure	LinCbO
Census	29 608	1184.10	0.0180	522	177
nom10shuttle	3.34	0.71	0.0613	1.25	0.44
Mushroom	25.92	1.95	0.1482	49	10.8
Connect	6239.75	307.10	0.9979	23 310	19 420
inter10shuttle	42.52	6.47	0.5429	19 223	16 698
Chess	1955.12	169.77	0.9830	325 076	234 309
Example 1 ($n = 5$)	0.05	0.04	1.0000	384	65
Example 1 ($n = 6$)	0.55	0.36	1.0000	–	–
Example 2 ($n = 10$)	0.22	0.27	1.0000	5.94	2.8
Example 2 ($n = 15$)	84.97	108.77	1.0000	203 477	29 710

This behavior is consistent with our expectations. In dense contexts, we usually have a large number of concept intents, which have to be enumerated as a side product by the exact algorithms. This slows them down considerably. The randomized algorithm does not have this weakness. In general, the randomized algorithm is preferable when the size of the basis is small with respect to the number of concept intents. If however the size of the canonical basis is comparable with the number of intents (as in the case of Census and nom10shuttle), the randomized algorithm tends to perform much worse, both in terms of runtime and quality. Context density can be a good (even if not always reliable) indicator for the applicability of the randomized algorithm.

Note also that the forty-thread version of the algorithm is up to twenty four times faster than the single-thread version on hard instances (such as Census and Connect).

6 Conclusion

Finding the canonical basis of a formal context is a computationally hard problem, which makes it reasonable to search for relatively efficient approximate solutions. To this end, an approach within the framework of probably approximately correct learning has been recently proposed [8,20] based on older works in machine learning and knowledge compilation [3,16]. The main contribution of this paper is two-fold. On the one hand, we extend the previously proposed approach by introducing frequency (or support) into approximation so as to shift the focus to approximating frequent implications. On the other hand, we present the first experimental evaluation of this approach in terms of its efficiency compared to the exact computation of the canonical basis.

Loosely speaking, a frequency-aware Horn approximation of a formal context $\mathbb{K}$, as considered in this paper, is a subset $\mathcal{L}$ of implications valid in $\mathbb{K}$ from which most valid frequent implications of $\mathbb{K}$ follow. Somewhat more precisely, such $\mathcal{L}$ is biased towards ensuring, for $A \subseteq M$ with large support $|A'|$, that $A = \mathcal{L}(A)$ if and only if $A = A''$, or, in the case of strong approximation, that $\mathcal{L}(A) =$

A''. In many application settings, frequent implications, also known as exact association rules, are regarded as the most important. We present a polynomial-time algorithm to compute such an approximation with high probability. The resulting set $\mathcal{L}$ is polynomial in the size of the input context and the parameters controlling the quality of approximation and the probability of achieving it. For certain practical purposes, such $\mathcal{L}$ may be even more valuable than a full basis of frequent implications (whichever way it is defined), whose size can be exponential in the size of the input context.

A frequency-aware approximation can be relevant even if we are not interested specifically in frequent implications. If most attribute subsets are of zero support, which often happens in real-world data sets, then $\mathcal{L}$ would be regarded as a good Horn approximation provided that $\mathcal{L}(A) = M$ for all (or most) subsets A that never occur in data—no matter where $\mathcal{L}$ maps those relatively few subsets that do occur. Taking frequency into account solves this problem by making it hard to ignore such subsets. This results in a much more meaningful approximation.

Our experiments show that, if the size of the canonical basis is small compared to the number of concept intents, a high-quality approximation can be computed in significantly less time than it takes Next Closure–based algorithms to compute the basis exactly. The randomized algorithm that we propose for this purpose is very easy to parallelize, which can further decrease the total runtime. It remains to be seen how well, in terms of efficiency, the algorithm performs against algorithms that are not related to Next Closure.

Acknowledgments

We thank Aimene Belfodil for letting us know of the paper [7].

References

1. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In: Buneman, P., Jajodia, S. (eds.) Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, DC, USA, May 26-28, 1993. pp. 207–216. ACM Press (1993). https://doi.org/10.1145/170035.170072, `https://doi.org/10.1145/170035.170072`
2. Angluin, D.: Queries and concept learning. Machine learning **2**(4), 319–342 (1988)
3. Angluin, D., Frazier, M., Pitt, L.: Learning conjunctions of Horn clauses. Machine Learning **9**, 147–164 (1992)
4. Arias, M., Balcázar, J.L.: Construction and learnability of canonical Horn formulas. Machine Learning **85**(3), 273–297 (2011). https://doi.org/10.1007/s10994-011-5248-5, `http://dx.doi.org/10.1007/s10994-011-5248-5`
5. Bazhanov, K., Obiedkov, S.: Optimizations in computing the Duquenne–Guigues basis of implications. Annals of Mathematics and Artificial Intelligence **70**(1), 5–24 (2014)

6. Beeri, C., Bernstein, P.: Computational problems related to the design of normal form relational schemas. ACM TODS **4**(1), 30–59 (March 1979)
7. Boley, M., Lucchese, C., Paurat, D., Gärtner, T.: Direct local pattern sampling by efficient two-step random procedures. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 582–590. KDD'11, Association for Computing Machinery, New York, NY, USA (2011). https://doi.org/10.1145/2020408.2020500, `https://doi.org/10.1145/2020408.2020500`
8. Borchmann, D., Hanika, T., Obiedkov, S.: On the usability of probably approximately correct implication bases. In: Bertet, K., Borchmann, D., Cellier, P., Ferré, S. (eds.) Formal Concept Analysis. Proceedings ICFCA 2017. Lecture Notes in Computer Science, vol. 10308, pp. 72–88 (2017)
9. Borchmann, D., Hanika, T., Obiedkov, S.: Probably approximately correct learning of Horn envelopes from queries. Discrete Applied Mathematics **273**, 30 – 42 (2020). https://doi.org/https://doi.org/10.1016/j.dam.2019.02.036
10. Dua, D., Graff, C.: UCI Machine Learning Repository (2017)
11. Ganter, B.: Two basic algorithms in concept analysis. In: Proceedings of the 8th International Conference on Formal Concept Analysis. pp. 312–340. ICFCA'10, Springer-Verlag, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-11928-6_22
12. Ganter, B., Obiedkov, S.: Conceptual Exploration. Springer (2016)
13. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, Berlin/Heidelberg (1999)
14. Guigues, J.L., Duquenne, V.: Famille minimale d'implications informatives résultant d'un tableau de données binaires. Mathématiques et Sciences Humaines **24**(95), 5–18 (1986)
15. Janostik, R., Konecny, J., Krajča, P.: LinCbO: fast algorithm for computation of the Duquenne–Guigues basis (2021), `https://arxiv.org/abs/2011.04928`
16. Kautz, H.A., Kearns, M.J., Selman, B.: Horn approximations of empirical data. Artif. Intell. **74**(1), 129–145 (1995)
17. Khardon, R.: Translating between Horn representations and their characteristic models. J. Artif. Intell. Res. (JAIR) **3**, 349–372 (1995)
18. Kuznetsov, S.: On the intractability of computing the Duquenne–Guigues base. Journal of Universal Computer Science **10**(8), 927–933 (2004)
19. Obiedkov, S., Duquenne, V.: Attribute-incremental construction of the canonical implication basis. Annals of Mathematics and Artificial Intelligence **49**(1), 77–99 (2007)
20. Obiedkov, S.: Learning implications from data and from queries. In: Cristea, D., Le Ber, F., Sertkaya, B. (eds.) Formal Concept Analysis. pp. 32–44. Springer International Publishing, Cham (2019)
21. Yarullin, R., Obiedkov, S.: From Equivalence Queries to PAC Learning: The Case of Implication Theories. International Journal of Approximate Reasoning **127**, 1–16 (2020)