

One-shot Learners Using Negative Counterexamples and Nearest Positive Examples

Sanjay Jain ^{a,1} and Efim Kinber ^b

^a *School of Computing, National University of Singapore, Singapore 117590.*

Email: sanjay@comp.nus.edu.sg

^b *Department of Computer Science, Sacred Heart University, Fairfield, CT 06432-1000, U.S.A. Email: kinbere@sacredheart.edu*

Abstract

As some cognitive research suggests, in the process of learning languages, in addition to *overt* explicit negative evidence, a child often receives *covert* explicit evidence in form of corrected or rephrased sentences. In this paper, we suggest one approach to formalization of overt and covert evidence within the framework of *one-shot* learners via subset and membership queries to a teacher (oracle). We compare and explore general capabilities of our models, as well as complexity advantages of learnability models of one type over models of other types, where complexity is measured in terms of number of queries. In particular, we establish that “correcting” positive examples are sometimes more helpful to a learner than just negative (counter)examples and access to full positive data.

1 Introduction

There are two major formal models that have been used over the years to address various aspects of human learning: the Gold’s model [Gol67] of *identification in the limit*, that treats learning as a limiting process of creating and modifying hypotheses about the target concept, and the Angluin’s model [Ang88] of learning via queries that views learning as a finite (rather than an infinite limiting) process, however, allowing interaction between a learner and a teacher (formally, an oracle) in form of questions and answers. Unlike the

¹ Supported in part by NUS grant number R252-000-212-112 and R252-000-308-112.

Gold’s model, a learner in the latter model cannot change its mind: it can ask a finite number of questions, but, ultimately, it must produce a sole right conjecture. Such learners have been named *one-shot* in [LZ04]. There has been a good deal of research on one shot-learners using primarily *superset* queries (when a learner asks if a particular language is a superset of the target concept) and *disjointness* queries (when a learner asks if a particular language is disjoint with the target concept) ([LZ05, JLZ07]). In this paper, we study and compare one-shot learners that receive different types of answers to *subset* and *membership* queries. Learners making subset queries (testing if a particular language is a subset of the target concept) are concerned with a possibility of *overgeneralizing* — that is, including into conjecture data not belonging to the target concept. Membership queries test if a particular datum belongs to the target concept — it is, perhaps, the most natural type of a question to the teacher. We refer to these models as **SubQ** and, respectively, **MemQ**.

For this paper, we consider languages to be a subset of N , the set of natural numbers. Thus, terms like *least* counterexample (used below) are well defined. We define the nearest element in L to a number y as the $x \in L$ which minimizes $|y - x|$; in the case of two x ’s in L minimizing $|y - x|$, we take the smaller of the two x ’s to be the nearest. Thus, the order of nearest elements to y is $y, y - 1, y + 1, y - 2, y + 2, \dots, 0, 2y, 2y + 1, 2y + 2, \dots$

When considering languages over other domains, such as strings over a finite alphabet Σ , one can consider a recursive bijection between natural numbers and Σ^* to define least/nearest. Alternatively, some other possibilities are discussed in the conclusion.

While for a membership query, a natural answer would be just *yes* or *no*, a learner making a subset query can also receive a *negative counterexample* showing where the learner errs. In her original model [Ang88], D. Angluin suggested that a learner could receive an *arbitrary* negative counterexample for a subset query, when several negative counterexamples were possible. In addition to this, traditional, type of answers to subset queries (considered in several variants of models using subset queries, e.g., in [JK08, JK06]), we also consider the following types of answers:

— a learner receives the *least* negative counterexample (this type of counterexamples was considered, in particular, in [JK08, JK06]); we refer to this model as **LSubQ**.

— in addition to a negative counterexample, a learner receives a “correction”, the positive example *nearest* to the negative one; this approach stems from the following observation discussed, in particular, in [RP99]: while learning a language, in addition to *overt* explicit negative evidence (when a parent points out that a certain statement by a child is grammatically incorrect), a child

often receives also *covert* explicit evidence in form of corrected or rephrased utterances. As languages in our learning models are represented by subsets of the set of natural numbers, our concept of the nearest positive example seems to be appropriate in the given context. We apply the same idea to membership queries: we consider a model where a learner receives the nearest positive example if it gets the answer ‘no’ to a membership query. We refer to these two models as **NPSubQ** and, respectively, **NPMemQ**. A similar approach to “correction” queries was suggested in [BBBD05,BBDT06,TK07a,TK07b]: a learner, in response to a membership query, receives the least (in the lexicographic order) *extension* of the queried datum belonging to the target language. One can argue, however, that a (rephrased) correcting sentence, while obviously being close to the queried wrong one, is not necessarily an extension of it. Thus, in our model, we require the “correction” to be just close to a wrong datum.

— in the above approach, a teacher may have difficulties providing the nearest (correcting) positive example, as it can still be too complex — far larger than the negative example. Therefore, we consider also a variant of learning via queries, where the nearest positive example not exceeding the size of the negative example is provided (if any). We refer to the variants of this model for subset and, respectively, membership queries as **BNPSubQ** and **BNPMemQ** (**B** here stands for *bounded*).

In our most general models, we assume that, in addition to the subset and/or membership queries, a one-shot learner also has access to potentially all positive examples in the target concept. It can be easily seen that, when a learner can make indefinite number of subset or membership queries, this positive data presented to a one-shot learner becomes essentially irrelevant. However, we will also study the cases when the number of queries will be uniformly bounded, and in this context access to additional positive data may be important.

We restrict our attention to recursively enumerable classes of formal languages. More specifically, we concentrate primarily on indexable classes of recursive languages [Ang80,ZL95,LZZ08]; an example of an indexable class is the class of all regular languages (for the sake of comparison, we also give an example showing how our results can be extended if recursively enumerable classes of recursively enumerable languages are considered). In this context, it is natural to require a learner to output a conjecture that is an index of the target concept in the given numbering. It is also natural to require that subset queries are made about languages L_i from the given indexed family \mathcal{L} (as defined in the original Angluin’s model) — these languages can be viewed as potential conjectures. Additionally, we also require that membership queries are made only for elements which belong to some language in the class being learnt (the learner, having access to the numbering representing the target class of

recursive languages, can be assumed to have certain “innate” knowledge about the type of elements in the languages of the given class and about descriptions of those languages).

Note that our criteria of learnability are not closed under a subset, in the sense that \mathcal{L} may be learnable but some subclass $\mathcal{L}' \subset \mathcal{L}$ may not be, due to the requirement of asking queries only within the class.

Our primary goal is to compare variants of one-shot learners receiving variants of answers to subset and membership queries discussed above. First, we compare capabilities of these learners, establishing where learners in one model can learn classes of languages not learnable within the framework of another model. Secondly, we study when and how learners of one type can learn the same classes of languages more efficiently than the learners of the other type, where efficiency is measured in terms of the number of queries made during the learning process.

The paper is structured as follows. In Sections 2 and 3 we provide necessary notation and define our models of one-shot learners via subset and membership queries. In Section 4 we compare learning powers of different models defined in Section 3. First we show that if the number of queries is not uniformly bounded, then access to a text for a language does not enhance capabilities of a learner in any of our models. Thus, in cases where we compare the learning power of different models, we can assume that the learner does not receive a text of a language. Specifically, we establish that

(a) *least* counterexamples provided in response to subset queries can help to learn classes of languages that cannot be learned even if the teacher, in addition to *arbitrary* negative counterexample, provides the nearest positive example to a negative counterexample too;

(b) learners receiving arbitrary nearest positive and bounded nearest positive examples for subset or membership queries and corresponding counterexamples or, respectively, answers ‘no’ are incomparable, and can sometimes learn more than the respective learners receiving least counterexamples but no nearest positive data.

(c) learners using membership queries can sometimes learn more than the ones using subset queries getting least counterexamples and the nearest positive examples; conversely, learners using subset queries and getting the weakest type of feedback can sometimes learn more than the ones using membership queries and getting the strongest type of feedback in the framework of our models.

In Section 5, we give an example, showing how the results in Section 4 (which considered indexable classes of recursive languages) can be translated to recur-

sively enumerable classes of recursively enumerable languages; this example (and other corresponding results) requires a somewhat more complex technique than those in Section 4.

In Section 6 we consider when a uniformly bounded number of queries of type QA gives advantages (for learning) over queries of type QB . Our main results can be summarized as follows:

(a) one subset query providing the least counterexample can sometimes help a learner more than any number of subset queries returning arbitrary counterexamples, even if the nearest positive examples and access to full positive data are provided;

(b) one membership query and either the nearest positive example (for the answer ‘no’), or access to full positive data can sometimes help a learner more than any number of subset queries returning least counterexamples and the *bounded* nearest positive examples, or returning arbitrary counterexamples and the nearest positive examples — even in the presence of full positive data; for showing advantage over least counterexamples and the nearest positive examples, we need either one membership query and access to full text of the target language or two membership queries and the nearest positive examples in the case of ‘no’ answer.

(c) on the other hand, a finite number of subset queries returning least counterexamples and the nearest positive examples can be used to learn any class of languages learnable using only *one* membership query and the nearest positive example; if no nearest positive examples to membership queries are provided, then $2^r - 1$ subset queries are enough to learn any class learnable using r membership queries; if the bounded nearest positive examples to membership queries are provided, then a finite number of subset queries is enough to learn any class learnable using a bounded number of membership queries;

(d) still, one membership query returning a *bounded* nearest positive example can sometimes be more helpful to a learner than a prefixed bounded number of subset queries returning least negative counterexamples and the nearest positive examples — even in the presence of full positive data.

In this section, we also demonstrate that $k + 1$ membership or subset queries can do more than k queries of the same type — even when the strongest additional information (within the framework of our models) is provided. On the other hand, for both membership and subset queries, it is shown that no bounded number of membership or subset queries with the strongest additional information can reach the power of learners using unlimited number of queries of respective types.

In Section 7 we study the following problem: when a class \mathcal{L} is learnable

using query type QB , can one speed up the learning process (in terms of the number of queries used) by using query type QA ? We address questions such as when classes which are learnable using small number of queries of a type QA , require arbitrarily large number of queries of a type QB . For example, we address questions about existence of classes which can be learned using 1 query of a type QA , or a finite number of queries of a type QB ($k + 1$ queries of the type QB), but cannot be learned using a bounded number of queries of the type QB (k queries of the type QB).

Overall, we hope that our results and multitude of different examples of classes witnessing separations will give the reader a good insight on how one-shot learners using membership and subset queries operate. Section 8, Conclusion, is devoted to discussion of our results and possible directions for future research.

2 Notation and Preliminaries

Any unexplained recursion theoretic notation is from [Rog67]. The symbol N denotes the set of natural numbers, $\{0, 1, 2, 3, \dots\}$. Symbols \emptyset , \subseteq and \subset , denote empty set, subset and proper subset, respectively. Cardinality of a set S is denoted by $\text{card}(S)$. The maximum and minimum of a set are denoted by $\max(\cdot)$, $\min(\cdot)$, respectively, where $\max(\emptyset) = 0$ and $\min(\emptyset) = \infty$.

We let $\langle \cdot, \cdot \rangle$ stand for an arbitrary, computable, bijective mapping from $N \times N$ onto N [Rog67]. We assume without loss of generality that $\langle \cdot, \cdot \rangle$ is monotonically increasing in both of its arguments.

By φ we denote an acceptable numbering of all partial recursive functions [Rog67] from N to N . φ_i denotes the partial recursive function computed by program i in the φ -system. A language is a subset of N . W_i denotes $\text{domain}(\varphi_i)$. W_i is thus the i -th recursively enumerable set (language), in some acceptable numbering of recursively enumerable (r.e.) sets. Symbol L , with or without decorations, ranges over recursively enumerable languages. Symbol \mathcal{L} , with or without decorations, ranges over the sets of recursively enumerable languages.

We let $K = \{i \mid i \in W_i\}$. Note that K is a recursively enumerable but not recursive set [Rog67].

\mathcal{L} is called an indexed family of recursive languages (abbreviated: indexed family) iff there exists an indexing $(L_i)_{i \in N}$ of languages such that:

- (i) $\{L_i \mid i \in N\} = \mathcal{L}$.

(ii) One can effectively determine, in i and x , whether $x \in L_i$.

We now present some concepts from the language learning theory. A *sequence* σ is a mapping from an initial segment of N into $(N \cup \{\#\})$. The *content* of a sequence σ , denoted $\text{content}(\sigma)$, is the set of natural numbers in the range of σ . The *length* of σ , denoted by $|\sigma|$, is the number of elements in σ .

Intuitively, $\#$'s represent pauses in the presentation of data. We let σ , with or without decorations, range over the finite sequences. We denote the sequence formed by the concatenation of σ' at the end of σ by $\sigma\sigma'$. Sometimes we abuse the notation and use σx to denote the concatenation of the sequence σ and the sequence of the length 1 which contains the element x .

Gold considered the following definition of presentation of data to a learner. A *text* T for a language L is a mapping from N into $(N \cup \{\#\})$ such that L is the set of natural numbers in the range of T . $T(i)$ represents the $(i + 1)$ -th element in the text. The *content* of a text T , denoted by $\text{content}(T)$, is the set of natural numbers in the range of T ; that is, the language which T is a text for. $T[n]$ denotes the finite initial sequence of T with length n .

There are several criteria for learning considered in the literature. We will be mainly concerned with finite learning [Gol67]. In this model, the learner gets a text for the language as input. After reading some initial portion of the text, the learner outputs a conjecture, i.e., a description of a language, and stops. If this conjecture is correct, i.e., if the description represents the target language, then we say that the learner **TxtFIN**-identifies the language from the given text. A learner **TxtFIN**-identifies a language if it **TxtFIN**-identifies the language from all texts for the language. A learner **TxtFIN**-identifies \mathcal{L} , if it **TxtFIN**-identifies each $L \in \mathcal{L}$. **TxtFIN** denotes the set of all classes \mathcal{L} such that some learner **TxtFIN**-identifies \mathcal{L} .

An issue in the above model is the hypothesis space that the conjecture of the learner comes from. For this paper, we are mainly concerned about learning indexed families of recursive languages and assume a class preserving hypothesis space. That is, we assume that the hypothesis space H_0, H_1, \dots , used for learning the class \mathcal{L} satisfies the following two properties:

(i) one can effectively, from i and x , determine whether $x \in H_i$;

(ii) $\mathcal{L} = \{H_i \mid i \in N\}$.

3 Definitions for Query Learning

In addition to possible access to texts representing full positive data, the learners in our model, following [Ang88], will also use two types of queries to the teacher (formally, the oracle): subset queries and membership queries.

We only consider queries in the context of class preserving learning. That is, for learning a class \mathcal{L} , all the hypotheses are assumed to be from a hypothesis space H_0, H_1, \dots , where H_0, H_1, \dots form an indexed family and $\{H_0, H_1, \dots\} = \mathcal{L}$.

The subset queries are now restricted to the form “ $H_i \subseteq \text{target language?}$ ”. Correspondingly, the membership queries are also assumed to come from the hypothesis space: the learner only asks membership queries of the form “ $x \in \text{target language?}$ ”, for some $x \in \bigcup_{i \in N} H_i$. Note that this approach is somewhat different from traditional membership queries, where any member of N may be queried. If a learner uses hypotheses from the indexed family, it is natural to require that it only tests if elements belonging to candidate conjectures belong to the target concept. Moreover, if one allows membership queries for any member of N , then the learner can obtain all positive and negative data (so-called *informant*) for the target language, and thus the model would collapse to learning from informant (making the nearest positive examples irrelevant, except for the case of learning the empty language \emptyset). For these reasons, for learning \mathcal{L} , we restrict our study to considering membership queries only for the elements of $\bigcup_{L \in \mathcal{L}} L$.

The ‘yes’/‘no’ answer provided to the learner is based on whether the answer to the query is true or false. For subset queries (about H_i), in case of ‘no’ answer (meaning that H_i is not a subset of the target language), the teacher also provides a negative counterexample, which is a member of H_i , but not a member of the target language. Here, we make distinction between two different cases: when the least counterexample is provided (we refer to such queries as **LSubQ**) and when an arbitrary counterexample is provided (we refer to such queries as **SubQ**).

In related work, [GS91,MPS92] consider queries formulated using first order logic. [FGJ⁺94,KS96] consider asking queries to an arbitrary oracle (such as the oracle for halting problem), where the queries may not be directly related to the target language. [GL08] consider answering queries in the limit (from text), rather than trying to infer programs (with or without queries).

In addition, for ‘no’ answers to subset queries, we often also consider providing the learner with the nearest positive example to the negative counterexample. In the context of membership queries, if the answer is ‘no’, the learner is then provided the positive example nearest to the queried element x . We will denote it by using the prefix **NP** to the query type. We also consider the

variant of providing the *bounded* nearest positive example, when the nearest positive example not exceeding the negative counterexample (or the negative element, in the context of membership queries) is provided (this is denoted by using the prefix **BNP** to the query type). In the above cases, if the (bounded) nearest positive example does not exist, then ‘none’ is given as the (bounded) nearest positive example.

The above will provide us with the the following criteria for one-shot learnability:

<i>Correction type</i>	<i>Query type</i>		
	\subseteq , any counterexample	\subseteq , least counterexample	membership
none	SubQ	LSubQ	MemQ
nearest +ve	NPSubQ	NPLSubQ	NPMemQ
bounded nearest +ve	BNPSubQ	BNPLSubQ	BNPMemQ

In addition, text may or may not be provided to a learner: this is denoted by using **Txt** in front of the criterion name (for example, **TxtNPSubQ**). Note that the learner in this case outputs its hypothesis after asking finitely many queries, and reading a finite portion of the text.

Below we formally give the definition of **NPLSubQ**. Other criteria can be defined similarly.

Definition 1 (a) M **NPLSubQ**-identifies \mathcal{L} iff for some class preserving hypothesis space H_0, H_1, \dots , for all target $L \in \mathcal{L}$, M asks a finite sequence of subset queries, and, then outputs an index i such that $H_i = L$. The answer provided by the teacher for each subset query ‘is H_i a subset of L ’ is as follows:

- (i) ‘yes’, if $H_i \subseteq L$;
- (ii) ‘no’, if $H_i \not\subseteq L$; in addition, the learner is provided with $x = \min(H_i - L)$ as a negative counterexample and a y such that y is the earliest element in the sequence $x - 1, x + 1, x - 2, x + 2, \dots, 0, 2x, 2x + 1, 2x + 2, \dots$ which belongs to L (if there is no such y , then the special answer ‘none’ is provided to the learner).

(b) **NPLSubQ** = $\{\mathcal{L} \mid \text{some computable learner } M \text{ NPLSubQ-identifies } \mathcal{L}\}$.

Note that later queries may depend on earlier answers (and, in the case of text being provided, on the elements of the text already read by the learner).

We sometimes consider limiting the number of queries made by the learner. For example, $\mathbf{TxtNPMemQ}^k$ denotes the criterion $\mathbf{TxtNPMemQ}$ where the number of queries made by the learner is limited to at most k .

Note that the power of teachers/oracles in our definitions is not limited. That is, sometimes oracles may be non-algorithmic.

4 Relationships Among Various Query Learning Criteria

4.1 Providing Text

We first show that when there is no bound on the number of queries, texts do not help: providing a text does not increase the learning capability of one-shot learners.

Theorem 2 *Suppose $\mathbf{Q} \in \{\mathbf{SubQ}, \mathbf{LSubQ}, \mathbf{MemQ}\}$. Then,*

$$\mathbf{Q} = \mathbf{TxtQ}.$$

$$\mathbf{NPQ} = \mathbf{TxtNPQ}.$$

$$\mathbf{BNPQ} = \mathbf{TxtBNPQ}.$$

PROOF. We only show $\mathbf{SubQ} = \mathbf{TxtSubQ}$. Other parts can be proved similarly.

Suppose M $\mathbf{TxtSubQ}$ -identifies \mathcal{L} . Then, define M' not having access to the positive data as follows. M' searches for a finite segment σ and a conjecture A made by M on σ such that $\text{content}(\sigma) \subseteq A \subseteq \text{target language}$, where the answers to the queries made by M are answered in the same way as received by M' for the same queries. If and when such σ and A are found, M' outputs the conjecture A . On any input text for L which starts with σ , M would behave as in the simulation, and thus output the conjecture A . Thus A must be the correct conjecture. \blacksquare

4.2 Variants of \mathbf{SubQ}

In this subsection, we explore relationships between different variants of \mathbf{SubQ} .

First, we show that getting least counterexamples can sometimes be more helpful to learners than getting arbitrary counterexamples along with the nearest

(bounded) positive examples to the counterexamples.

Theorem 3 $\mathbf{LSubQ} - (\mathbf{TxtNPSubQ} \cup \mathbf{TxtBNPSubQ}) \neq \emptyset$.

PROOF. Let $L = \{100i + x \mid i \in N, x \in \{1, 2, 3\}\}$.

Let $L_i = L - \{100i + 1, 100i + 3\}$.

Let $X_i = L_i \cup \{100i + 1\}$.

Let $\mathcal{L} = \{L\} \cup \{L_i \mid i \in N\} \cup \{X_i \mid i \in K\}$.

It is easy to verify that \mathcal{L} is an indexed family. To see this, let w_0, w_1, \dots , be a recursive enumeration of K . Let $Z_0 = L$, $Z_{2i+2} = L_i$, $Z_{2i+1} = X_{w_i}$. Now, Z_0, Z_1, Z_2, \dots gives an indexing of \mathcal{L} , where membership problem for Z_i can be decided effectively in i .

L is needed for the proof only in the positive part of the theorem (to get the appropriate counterexample needed for distinguishing between L_i and X_i). Note that we require the queries to be class preserving — the negative part of the proof relies on this.

To see that $\mathcal{L} \in \mathbf{LSubQ}$, first query whether L is a subset of the target language. If L is a subset of the target language, then the target language must be L . Otherwise, the least counterexample is either $100i + 1$ or $100i + 3$, for some i . In the former case, the target language must be L_i . In the latter case, the target language must be X_i .

Now suppose by way of contradiction that $\mathcal{L} \in \mathbf{TxtNPSubQ}$ ($\mathbf{TxtBNPSubQ}$) as witnessed by M . We then give the following algorithm to solve K .

On the input i

1. Simulate M on a text for L_i , where answers to the queries of M are as follows.

For the queries which contain $100i + 3$, answer ‘no’ along with the counterexample $100i + 3$, and the (bounded) nearest positive example being $100i + 2$.

For the queries which do not contain $100i + 3$, answer ‘yes’.

2. If in the simulation above M ever queries a language which contains $100i + 1$ but not $100i + 3$, then output $i \in K$.

If the above simulation stops with a conjecture, without querying a language which contains $100i + 1$ but not $100i + 3$, then output $i \notin K$.

End

Now, as M **TxtNPSubQ**-identifies (**TxtBNPSubQ**-identifies) \mathcal{L} , M , for the target language L_i , must eventually output a conjecture. During the process, if M queries a language containing $100i + 1$, but not $100i + 3$, then we have $i \in K$, as M is allowed only to query languages within the class \mathcal{L} . On the other hand, if M outputs a conjecture without querying about X_i , then, since the answers given to the queries of M are consistent with the target language being L_i or X_i , we must have that $X_i \notin \mathcal{L}$ (otherwise, M cannot identify both L_i and X_i). Thus $i \notin K$. \blacksquare

Note that the negative part of the proof used the following idea:

Remark 4 *Suppose \mathcal{L} contains the languages L_i , for all i , and X_i for i such that $i \in K$. Suppose also that, for all i , $L_i \subset X_i$, and one can effectively determine when, during a one-shot learning process, the subset query made is for X_i . Suppose, for a learner, one can effectively provide answers to the subset queries of type Q , except for a query being about the language X_i itself, in such a way that the answers are consistent with target language being either L_i or X_i . Then the class is not finitely learnable (based on the corresponding query type Q).*

The similar result holds when one considers membership queries instead of subset queries, when $X_i - L_i = \{x_i\}$, where x_i belongs only to X_i and no other language in the class.

Based on the above remark, often for diagonalization, we will just indicate how the queries can be answered to solve the halting problem as above, rather than giving the full proof.

Our next result shows that learners getting arbitrary counterexamples and the unbounded positive examples nearest to them can sometimes learn more than the ones getting the bounded nearest positive example. This holds even if the latter ones receive the least counterexamples and have access to full positive data.

Theorem 5 **NPSubQ** – **TxtBNPLSubQ** $\neq \emptyset$.

PROOF. Let $L = \{100i + x \mid i \in N, x \in \{1, 3\}\}$.

Let $L_i = L - \{100i + 1\}$.

Let $X_i = L_i \cup \{100i + 2\}$.

Let $\mathcal{L} = \{L\} \cup \{L_i \mid i \in N\} \cup \{X_i \mid i \in K\}$.

It is easy to verify that \mathcal{L} is an indexed family. Now, we show that $\mathcal{L} \in$ **NPSubQ**. First, a learner can query L . If L is contained in the target lan-

guage, then the target language must be L . If there is a counterexample, it must be $100i + 1$, for some i . Now the target language is X_i if the nearest positive example was $100i + 2$. Otherwise, the target language must be L_i .

We now show that $\mathcal{L} \notin \mathbf{TxtBNPLSubQ}$. We use Remark 4, as one can answer queries (except for X_i) of a supposed learner M based on target language being L_i (these answers will be consistent with target language being X_i also): if the queried language A contains $100i + 1$, then the answer is ‘no’, with the negative counterexample being the least element in $A - L_i$ (which is $\leq 100i + 1$), and the bounded nearest positive example being the largest element in L_i which is $\leq \min(A - L_i)$, if any (this element would be $\leq 100(i - 1) + 3$ or ‘none’); if the queried language does not contain $100i + 1$, then the answer is ‘yes’. Thus, by using Remark 4, we have that $\mathcal{L} \notin \mathbf{TxtBNPLSubQ}$. ■

From the above result, we can immediately derive the following corollary.

Corollary 6 $\mathbf{SubQ} \subset \mathbf{NPSubQ}$.
 $\mathbf{LSubQ} \subset \mathbf{NPLSubQ}$.

Now we show that, in the above result, the learners getting the unbounded nearest positive examples can be replaced by the ones getting the bounded nearest positive examples, and vice versa.

Therefore, the learners via subset queries and getting counterexamples and the nearest positive data of these two types are incomparable.

Theorem 7 $\mathbf{BNPSubQ} - \mathbf{TxtNPLSubQ} \neq \emptyset$.

PROOF. Let $L = \{100i + x \mid i \in N, x \in \{3, 4\}\}$.

Let $L_i = L - \{100i + 3\}$.

Let $X_i = L_i \cup \{100i + 1\}$.

Let $\mathcal{L} = \{L\} \cup \{L_i \mid i \in N\} \cup \{X_i \mid i \in K\}$.

It is easy to verify that \mathcal{L} is an indexed family.

To see that $\mathcal{L} \in \mathbf{BNPSubQ}$, a learner can query L . If L is contained in the target language, then the target language must be L . If there is a counterexample, it must be $100i + 3$, for some $i \in N$. Now the target language is X_i , if the bounded nearest positive example is $100i + 1$. Otherwise the target language must be L_i .

We now show that $\mathcal{L} \notin \mathbf{TxtNPLSubQ}$. We use Remark 4, as one can answer queries (except for X_i) of a supposed learner M based on the target language

being L_i (these answers will be consistent with the target language being X_i too): if the queried language A ($\neq X_i$) contains $100i + 3$, then the answer is ‘no’, with the negative counterexample being $\min(A - L_i)$ (which would be $\leq 100i + 3$), and the nearest positive example being the nearest element in L_i to $\min(A - L_i)$ (note that this would also be the nearest element in X_i to $\min(A - L_i)$); if the queried language does not contain $100i + 3$, then the answer is ‘yes’.

Thus, by using Remark 4, we have that $\mathcal{L} \notin \mathbf{TxtNPLSubQ}$. ■

As in the case of learners getting the unbounded nearest positive examples, from the above theorem, we immediately derive the following corollary.

Corollary 8 $\mathbf{SubQ} \subset \mathbf{BNPSubQ}$.
 $\mathbf{LSubQ} \subset \mathbf{BNPLSubQ}$.

Now we show that learners getting the nearest positive examples, in addition to arbitrary counterexamples, can sometimes learn more than the ones getting the least counterexamples and full positive data, but no nearest positive examples.

Theorem 9 $\mathbf{NPSubQ} \cap \mathbf{BNPSubQ} - \mathbf{TxtLSubQ} \neq \emptyset$.

PROOF. Let $L = \{100i + x \mid i \in N, x \in \{2, 4\}\}$.

Let $L_i = L - \{100i + 2\}$.

Let $X_i = L_i \cup \{100i + 1\}$.

Let $\mathcal{L} = \{L\} \cup \{L_i \mid i \in N\} \cup \{X_i \mid i \in K\}$.

It is easy to verify that \mathcal{L} is an indexed family.

To see that $\mathcal{L} \in \mathbf{NPSubQ}$ ($\mathbf{BNPSubQ}$), a learner can query L . If L is contained in the target language, then the target language must be L . If there is a counterexample, it must be $100i + 2$, for some $i \in N$. Now the target language is X_i , if the (bounded) nearest positive example is $100i + 1$. Otherwise the target language must be L_i .

We now show that $\mathcal{L} \notin \mathbf{TxtLSubQ}$. We use Remark 4, as one can answer queries (except for X_i) of a supposed learner M based on the target language being L_i (the negative counterexample for query $A \notin \{L_i, X_i\}$ would be $\min(A - L_i) = \min(A - X_i)$). Thus, by using Remark 4, we have that $\mathcal{L} \notin \mathbf{TxtLSubQ}$. ■

4.3 MemQ vs SubQ

In this subsection, we compare learning capabilities of one-shot learners using membership and subset queries. We establish that, within the framework of our models, the weakest learners using one type of queries can sometimes do more than the strongest learners using the other type of queries.

First, we show that learners using membership queries can sometimes be stronger than the ones using subset queries.

Theorem 10 $\text{MemQ} - (\text{TxtNPLSubQ} \cup \text{TxtBNPLSubQ}) \neq \emptyset$.

PROOF. Let $L = \{0\} \cup \{100i + 2 \mid i \in N\}$.

Let $L_i = \{100i + 1, 100i + 5\}$.

Let $X_i = \{100i + 1, 100i + 2, 100i + 5\}$.

Let $\mathcal{L} = \{L\} \cup \{L_i \mid i \in N\} \cup \{X_i \mid i \in K\}$.

We first show that $\mathcal{L} \in \text{MemQ}$. First, a learner queries 0. If the answer is ‘yes’, then the target language must be L . Otherwise, the learner determines an i such that $100i + 1$ is in the target language. Then querying $100i + 2$, the learner determines whether the target language is L_i or X_i .

We now show that $\mathcal{L} \notin \text{TxtNPLSubQ} (\text{TxtBNPLSubQ})$. We use Remark 4, as one can answer queries (except for X_i) of a supposed learner M as follows:

If the query of M contains $100i + 1$, then answer ‘yes’. Otherwise, answer ‘no’ to the query, with the least element of the query being the least negative counterexample. In the case of the queried language being L , there is no bounded nearest positive example. In all other cases, the nearest (bounded) positive example will be $100i + 1$ or $100i + 5$ or ‘none’.

Thus, by using Remark 4, we have that $\mathcal{L} \notin \text{TxtNPLSubQ} (\text{TxtBNPLSubQ})$. ■

Our next result demonstrates the advantage of subset queries over membership queries.

Theorem 11 $\text{SubQ} - (\text{TxtNPMemQ} \cup \text{TxtBNPMemQ}) \neq \emptyset$.

PROOF. Let $\mathcal{L} = \{L \mid \text{card}(N - L) \leq 1\}$. It is easy to verify that $\mathcal{L} \in \text{SubQ}$ (a learner can query N — if the answer is ‘yes’, then target language must

be N ; otherwise, the target language is $N - \{x\}$, where x is the negative counterexample received).

On the other hand, suppose by way of contradiction that some learner **TxtNPMemQ** (**TxtBNPMemQ**)-identifies \mathcal{L} . Let σ be the initial segment on which the learner conjectures N , when all the membership queries are answered ‘yes’. Then, for any x such that $x \notin \text{content}(\sigma)$ and x has not been queried by the learner, we have that the learner does not identify the language $N - \{x\}$. ■

4.4 Different variants of **MemQ**

In this subsection, we compare different variants of learners using membership queries.

Our first two results show, as in the case of subset queries, the advantages of learners getting the unbounded or bounded nearest positive examples (in addition to answers ‘no’) compared to learners getting the nearest positive example of the other type.

Theorem 12 **NPMemQ** – **TxtBNPMemQ** $\neq \emptyset$.

PROOF. Let $L = \{0\} \cup \{100i + 2 \mid i \in N\}$.

Let $L_i = \{100j + 2 \mid j \geq i\}$.

Let $X_i = L_i \cup \{100i + 1\}$.

Let $\mathcal{L} = \{L\} \cup \{L_i \mid i \in N\} \cup \{X_i \mid i \in K\}$.

Now we show $\mathcal{L} \in \mathbf{NPMemQ}$. First query 0. If 0 is in the target language, then target language must be L . Otherwise, the nearest positive example is either $100i + 2$ or $100i + 1$ for some i . In the former case, the target language must be L_i , and in the latter case, the target language must be X_i .

We now show that $\mathcal{L} \notin \mathbf{TtxtBNPMemQ}$. We use Remark 4, as one can answer queries x (except for $100i + 1$) of a supposed learner M based on whether $x \in L_i$. For $x \notin L_i$, if the queried element is 0 or $100j + b$, for $b \in \{1, 2\}$ and some $j < i$, then the bounded nearest positive example is ‘none’. If the queried element is $100j + 1$, for some $j > i$, then the bounded nearest positive example is $100(j - 1) + 2$.

Thus, by using Remark 4, we have that $\mathcal{L} \notin \mathbf{TtxtBNPMemQ}$. ■

Theorem 13 $\text{BNPMemQ} - \text{TxtNPMemQ} \neq \emptyset$.

PROOF. Let $A_i = \{100i, 100i + 6, 100i + 7\}$. $L_i = \{100i, 100i + 7\}$. $X_i = \{100i, 100i + 4, 100i + 7\}$.

Let $\mathcal{L} = \{A_i, L_i \mid i \in N\} \cup \{X_i \mid i \in K\}$.

It is easy to verify that $\mathcal{L} \in \text{BNPMemQ}$. A learner first finds an i such that $100i$ belongs to the target language. Then it queries $100i+6$; if $100i+6$ belongs to the target language, then the target language must be A_i . Otherwise, if the bounded nearest positive example is $100i + 4$, then the target language is X_i . Otherwise the target language is L_i . Note that the nearest unbounded positive example is $100i + 7$ in both cases, which does not help to determine distinction between X_i and L_i .

We now show that $\mathcal{L} \notin \text{TxtNPMemQ}$. We use Remark 4, as one can answer queries x (except for $x = 100i + 4$) of a learner M based on membership relation with L_i . For queries $x = 100j + w$, for $j < i$, the nearest positive example would be $100i$. For queries $x = 100j + w$, for $j > i$, the nearest positive example would be $100i + 7$. For query $x = 100i + 6$, the nearest positive example would be $100i + 7$.

Thus, by using Remark 4, we have that $\mathcal{L} \notin \text{TxtBNPMemQ}$. ■

Now we show that getting the nearest positive examples (to the counterexamples) of either type can sometimes be more helpful to learners than getting access to the text (full positive data) for the target language.

Theorem 14 $\text{NPMemQ} \cap \text{BNPMemQ} - \text{TxtMemQ} \neq \emptyset$.

PROOF. Consider $L = \{100i + x \mid i \in N, x \in \{1, 3, 5\}\}$.

$L_i = \{0\} \cup \{100i + 1, 100i + 5\}$.

$X_i = L_i \cup \{100i + 2\}$.

$\mathcal{L} = \{L\} \cup \{L_i \mid i \in N\} \cup \{X_i \mid i \in K\}$.

$\mathcal{L} \in \text{NPMemQ} \cap \text{BNPMemQ}$, as one can first query 0 — if it does not belong to the target language, then the target language must be L . Otherwise, using membership queries, one finds an i such that $100i + 1$ belongs to the target language. Now one can query $100i + 3$ — then the (bounded) nearest positive example being $100i + 1$ or $100i + 2$, will determine that the target language is L_i or X_i .

We now show that $\mathcal{L} \notin \text{TxtMemQ}$. We use Remark 4, as one can answer

queries (except for $100i + 2$) of a supposed learner M based on membership relation with L_i . Thus, by using Remark 4, we have that $\mathcal{L} \notin \mathbf{TxtMemQ}$. ■

5 Recursively Enumerable Classes of Recursively Enumerable Languages

In this section, we demonstrate that one can generalize Theorem 5 to the case when the hypothesis space is a recursively enumerable numbering of recursively enumerable languages, and, thus, queries may be about r.e. indices for the language queried, rather than decision procedures as considered in the previous section. Similar ideas can be used for the rest of our theorems in Section 4. Note that the results for r.e. languages do not follow from the results in the previous section, as the hypothesis space is now an r.e. class of r.e. languages, rather than an indexed family. Thus, for example, Remark 4 does not hold — one cannot algorithmically determine if the queried language Q is for X_i or not, when Q is defined as follows: $Q = L_i$, if $i \notin K$; $Q = X_i$ otherwise. The proof of the following theorem illustrates how this problem can be overcome. Similar technique can be used for other diagonalizations in the previous section.

Below, in the name of learning criteria, we use the prefix **Re** to denote that we are considering learnability of r.e. classes of r.e. languages.

Theorem 15 ReNPSubQ – ReTxtBNPLSubQ $\neq \emptyset$.

PROOF. We will define a partial recursive function φ_e later. Recall that $\langle \cdot, \cdot \rangle$ is increasing in both its arguments.

Let $L = \{100\langle i, 0 \rangle \mid i \in N\}$.

Let $L_{i,1,0} = (L - \{100\langle i, 0 \rangle\}) \cup \{100\langle i, 0 \rangle + 5\} \cup \{100\langle j, 0 \rangle + 4 \mid j \neq i\}$.

Let $L_{i,2,j} = L_{i,1,0} \cup \{100\langle i, k \rangle + 5 \mid k \leq j\} \cup \{100\langle i, 0 \rangle + 4\}$.

Let $\mathcal{L} = \{L\} \cup \{L_{i,1,0} \mid i \in N\} \cup \{L_{i,2,j} \mid \varphi_e(i) \downarrow, i, j \in N\}$.

Given a fixed φ_e , the above is an indexed family of recursive languages. The positive side of the theorem holds even if we consider queries using the indexing of an indexed family.

$\mathcal{L} \in \mathbf{ReNPSubQ}$: One can first ask the question whether $L \subseteq$ target language. If so, the target language must be L . Otherwise, let $100\langle i, 0 \rangle$ be the (only possible) negative counterexample. If the nearest positive example is $100\langle i, 0 \rangle + 4$, then find a j such that $L_{i,2,j}$ is a subset of the target language,

but $L_{i,2,j+1}$ is not a subset of the target language — then the target language must be $L_{i,2,j}$. On the other hand, if the nearest positive example is $100\langle i, 0 \rangle + 5$, then the target language must be $L_{i,1,0}$.

We now show that $\mathcal{L} \notin \mathbf{ReTxtBNPLSubQ}$. For this, we define the partial recursive function φ_e as follows. Let M_0, M_1, \dots denote a recursive enumeration of all $\mathbf{ReTxtBNPLSubQ}$ learners.

$\varphi_e(i) \downarrow$ iff M_i , being fed a text for $L_{i,1,0}$, eventually outputs a conjecture when its questions “Is W_m a subset of the target language?” are answered as follows (for the first case which applies, based on some standard interleaved search):

- A. If W_m contains $100\langle i, 0 \rangle + 5$, then answer ‘yes’ to the subset query.
- B. If W_m contains $100\langle j, 0 \rangle$, for all $j \leq i$, then answer ‘no’ to the subset query, give the negative counterexample $100\langle i, 0 \rangle$, along with the bounded nearest positive example $100\langle i - 1, 0 \rangle + 4$ (if $i = 0$, then there is no bounded nearest positive example).
- C. If, for some $k < i$, W_m contains $100\langle k, 0 \rangle + 5$, and $100\langle j, 0 \rangle$, for $j < k$, then answer ‘no’ to the subset query, give the negative counterexample $100\langle k, 0 \rangle + 5$, along with the bounded nearest positive example $100\langle k, 0 \rangle + 4$.
- D. If none of the above cases hold, then M_i ’s question is not answered, and $\varphi_e(i)$ will diverge.

We now consider the following cases.

Case 1: M_i does not eventually output a conjecture when its questions are answered as above.

In this case, clearly, $\varphi_e(i)$ does not converge. Thus, either M_i asks a query outside the class \mathcal{L} , or the answers to M_i are consistent with the target language being $L_{i,1,0}$. However, M_i does not output any conjecture and, thus, does not $\mathbf{ReTxtBNPLSubQ}$ -identify $L_{i,1,0}$.

Case 2: M_i eventually outputs a conjecture when its questions are answered as above. In this case, let r be maximal such that M_i queries $L_{i,2,r}$ before it outputs its conjecture or makes a query for a language outside \mathcal{L} . Then, M_i $\mathbf{ReTxtBNPLSubQ}$ -identifies at most one of $L_{i,2,s}$, for $s \geq r$.

The theorem follows from the above cases. ■

6 Complexity Hierarchy

This section gives the relationship between different criteria of query learning considered in the paper: **MemQ**, **SubQ**, **LSubQ**, with or without the (bounded) nearest positive example, and with or without text, when the number of queries may be bounded.

Most of the following results hold only for indexed families (the r.e. class trick considered in Section 5 often does not work when we are having only constant number of queries).

We begin with the following useful propositions. Proposition 16 and Proposition 17 (a) were implicit in the work [Ang88]. Angluin considered these claims for membership queries explicitly in [Ang01]. If one allows queries from outside the class, then better results (for **SubQ**) than in Proposition 16 can be obtained.

Proposition 16 [Ang01] *Suppose $\text{card}(\mathcal{L}) \leq k + 1$. Then $\mathcal{L} \in \mathbf{MemQ}^k$ and $\mathcal{L} \in \mathbf{SubQ}^k$.*

PROOF. We give the proof for completeness. Suppose \mathcal{L} consists of only $k + 1$ languages. To show that $\mathcal{L} \in \mathbf{SubQ}^k$, one proceeds as follows. Initially all languages in \mathcal{L} are possible candidates for the target language. At any stage, one can ask the subset query about a maximal (in terms of set inclusion) remaining candidate A . Then one can either eliminate A as being the target language (if the answer to the query is ‘no’), or determine that the target language is A (if the subset answer is ‘yes’). After k questions, only one language remains as a possible candidate.

For membership queries, one can ask a query about $x \in A - B$, to eliminate either A or B as a possible candidate for the target language. Thus, k queries can eliminate k of the $k + 1$ languages as potential candidates for the target language. ■

Now we show that if the number of membership queries is uniformly bounded and the nearest positive examples are bounded, then learnable classes are finite.

Proposition 17 *Suppose $k \in \mathbb{N}$.*

(a) [Ang01] *If $\mathcal{L} \in \mathbf{MemQ}^k$, then $\text{card}(\mathcal{L}) \leq 2^k$.*

(b) *If $\mathcal{L} \in \mathbf{BNPMemQ}^k$, then \mathcal{L} must be finite.*

PROOF. (a) Immediate, as the answers to the k possible questions by **MemQ** ^{k}

learner determine the target language.

(b) As the answers to the k questions by $\mathbf{BNPMemQ}^k$ learner and the finite number of possibilities for the bounded nearest positive examples determine the target language, \mathcal{L} must be finite. ■

Our next two results show how one-shot learners using a uniformly bounded number of membership queries can simulate the ones using uniformly bounded number of subset queries when the target class is finite. First we show that, under the given conditions, one subset query can be simulated by one membership query, when a text for the target language is provided.

Theorem 18 *Suppose \mathcal{L} is finite and $\mathcal{L} \in \mathbf{SubQ}^1$. Then, $\mathcal{L} \in \mathbf{TxtMemQ}^1$.*

PROOF. Suppose \mathcal{L} is in \mathbf{SubQ}^1 . This means that for the first query A asked by a learner, either the target language is A , or each element of A is missing from at most one language in the class. Thus, for each language L in \mathcal{L} , except A , one can associate an element x_L , such that x_L is not in L , but in every other language in \mathcal{L} . Thus, since all such x_L (a finite number) are known to the learner, a text will eventually leave out at most one such x_L , for $L \in \mathcal{L} - \{A\}$. Now membership query about this x_L can determine if the target language is L or A . ■

If the target class is finite, then a simulation of k subset queries can be done using $2^k - 1$ membership queries, when a text for the target language is provided.

Theorem 19 *Suppose \mathcal{L} is finite and $\mathcal{L} \in \mathbf{SubQ}^k$. Then $\mathcal{L} \in \mathbf{TxtMemQ}^{2^k-1}$.*

PROOF. Note that any class which can be learned using k subset queries has no subset chain of the size larger than 2^k . Suppose M \mathbf{SubQ}^k -identifies \mathcal{L} . Since the class is finite, there exists a finite subset of N , membership/non-membership of which in the target language determines the target. Thus, without loss of generality, we can assume that the rest of the elements of N do not matter. Thus, we are working with a finite set of languages over a finite domain, and thus non-effective learning is enough to show effective learning too.

We show the theorem by induction. For the base case, if $k = 1$, then, by Theorem 18, $\mathcal{L} \in \mathbf{TxtMemQ}^1$. For a larger k , suppose A is the first question that M asks. We divide the class \mathcal{L} into two parts: one consisting of the languages which contain A , and the other consisting of the languages which do not contain A . We will take at most 2^{k-1} queries to determine whether A is contained in the target language or not (along with a needed counterexample,

in the case of A not being a subset of the target language). Note that the largest subset chain among the languages (in \mathcal{L}) which are not a superset of A is of the size at most 2^{k-1} (otherwise M would not be able to identify \mathcal{L} using $k - 1$ further queries, after getting ‘no’ answer to the query A , along with a counterexample being an element which belongs to A but not to the largest language in a subset chain of the size $2^{k-1} + 1$).

Do the following until a counterexample to A is found, or all languages which are not supersets of A are eliminated from being candidates for the target language.

Possible = \mathcal{L} .

Neg = \emptyset .

Loop

1. Read more and more of input text T until we reach an initial segment σ of T such that for some $B \in \textit{Possible}$, $\text{content}(\sigma) \subseteq B$, and for all $C \in \textit{Possible}$ which contain $\text{content}(\sigma)$, $B \subseteq C$. (That is, there exists a unique minimal element in *Possible* which contains σ).
Eliminate from *Possible* all languages which do not contain $\text{content}(\sigma)$.
2. If $A \subseteq B$, then one can proceed with the simulation of M with answer ‘yes’ to the subset query A made by M . Note that by induction, we can simulate the rest of the queries of M by using at most $2^{k-1} - 1$ queries.
3. Else, pick $x \in A - B$ and do the membership query for x . If answer is ‘no’, then one can proceed with the simulation of M with the answer ‘no’ to the subset query A made by M (along with the negative counterexample x). Note that by induction, we can simulate the rest of the queries of M by using at most $2^{k-1} - 1$ queries.
4. Else, (answer is ‘yes’ to the membership query x): Eliminate from *Possible* all languages which do not contain x . Note that the size of the largest subset chain among remaining languages in *Possible* which do not contain A is reduced by at least 1 in the process (as B is eliminated from *Possible* but is contained in all the languages in the remainder of *Possible*).

End Loop

Since the set of all languages in \mathcal{L} that do not contain A is in \mathbf{SubQ}^{k-1} , it has no subset chain of the size larger than 2^{k-1} . By the comment at the end of step 4 above, note that each round of Loop uses one query and reduces the size of the largest subset chain among languages in *Possible* which do not contain A , by at least 1. Thus, there are at most 2^{k-1} queries, before the answer to the subset query for A can be determined. We are now done by induction. ■

Now we turn our attention to arbitrary (possibly infinite) target classes. First,

based on Theorem 11 from the previous section, we note that just one subset query sometimes helps a learner more than the strongest type of membership queries within the framework of our models — including access to a text of the target language.

Theorem 20 $\text{SubQ}^1 - (\text{TxtNPMemQ} \cup \text{TxtBNPMemQ}) \neq \emptyset$.

PROOF. Proof of Theorem 11 witnesses this too. ■

The next result demonstrates what advantages of a bounded number of membership queries over subset queries are possible. The following picture is quite complex. In particular, we show (Theorem 21(i)) that r membership queries can be simulated by $2^r - 1$ subset queries (we also show that this estimate is tight, Theorem 21(h)). However, if, in addition to membership queries, a learner gets either access to a text of the target language (Theorem 21(a)), or the nearest positive examples (Theorem 21 parts (b), (e), (f)), then, in most cases, just one membership query gives advantage over a learner using subset queries (and getting strongest feedback and having access to text of the target language). On the other hand, a learner using a finite number of subset queries and getting *least* counterexamples and the nearest positive examples can simulate any learner using just one membership query and getting the nearest positive examples (Theorem 21(d)); still, such a simulation is not always possible if a learner can use two such membership queries (Theorem 21(c)). Also, learners using a uniformly bounded number of membership queries and getting *bounded* positive examples can always be simulated by learners using subset queries if the number of queries of the latter type is not bounded (Theorem 21(g)).

Theorem 21 (a) $\text{TxtMemQ}^1 - (\text{TxtNPLSubQ} \cup \text{TxtBNPLSubQ}) \neq \emptyset$.

(b) $\text{NPMemQ}^1 - (\text{TxtNPLSubQ} \cup \text{TxtBNPLSubQ}) \neq \emptyset$.

(c) $\text{NPMemQ}^2 - (\text{TxtNPLSubQ} \cup \text{TxtBNPLSubQ}) \neq \emptyset$.

(d) $\text{NPMemQ}^1 \subseteq \bigcup_{r \in \mathbb{N}} \text{NPLSubQ}^r$.

(e) For all k , $\text{NPMemQ}^1 - (\text{TxtNPLSubQ}^k \cup \text{TxtBNPLSubQ}^k) \neq \emptyset$.

(f) For all k , $\text{BNPMemQ}^1 - (\text{TxtNPLSubQ}^k \cup \text{TxtBNPLSubQ}^k) \neq \emptyset$.

(g) For all k , $\text{BNPMemQ}^k \subseteq \bigcup_{r \in \mathbb{N}} \text{SubQ}^r$.

(h) For $r \geq 1$, $\text{MemQ}^r - (\text{TxtBNPLSubQ}^{2^r-2} \cup \text{TxtNPLSubQ}^{2^r-2}) \neq \emptyset$.

(i) For $r \geq 1$, $\text{MemQ}^r \subseteq \text{SubQ}^{2^r-1}$.

PROOF. (a) \mathcal{L} in the proof of Theorem 10 can be shown to be in **TextMemQ**¹, as one can first wait for either 0 or $100i + 1$, for some i , to appear in the input text. If 0 is in the input text, then L must be the target language. Otherwise querying $100i + 2$ determines whether the target language is L_i or X_i .

(b) Let $L = \{0\} \cup \{100i + x \mid i \in N, x \in \{0, 3\}\}$.

Let $L_i = \{100i + 2\} \cup \{100j + 3 \mid j > i\}$.

Let $X_i = L_i \cup \{100i + 1\}$.

Let $\mathcal{L} = \{L\} \cup \{L_i \mid i \in N\} \cup \{X_i \mid i \in K\}$.

Now, $\mathcal{L} \in \mathbf{NPMemQ}$ ¹ (Query 0; if the answer is ‘yes’, then the target language must be L ; otherwise, if the nearest positive example is $100i + 2$ for some i , then the target language must be L_i ; otherwise the nearest positive example would be $100i + 1$, for some i , and the target language is X_i).

We now show that $\mathcal{L} \notin \mathbf{TextBNPLSubQ}$. Suppose by way of contradiction that a learner **TextBNPLSubQ**-identifies \mathcal{L} . We proceed as in Remark 4. One can solve the halting problem K as follows. On input i , give text for L_i as input to the learner.

For queried language containing an element $< 100i + 1$, one can give the least element $< 100i + 1$ in the queried language as a counterexample (with no bounded nearest positive example). The query for a language with the least element $100i + 2$ is answered ‘yes’. Queries for a language with the least element being either $100j + 1$ or $100j + 2$, where $j > i$, are answered ‘no’, with respectively $100j + 1$ or $100j + 2$ being the negative counterexample; the bounded nearest positive example would be either $100(j - 1) + 3$ (if $j > i + 1$) or $100i + 2$ (if $j = i + 1$). Queries for a language which contains $100i + 1$, if made, determine that $i \in K$. If the conjecture is made before querying a language which contains $100i + 1$, then $i \notin K$ (as otherwise the answers to queries are consistent with both L_i and X_i , contradicting **TextBNPLSubQ**-learnability of \mathcal{L} by the learner).

Similarly, $\mathcal{L} \notin \mathbf{TextNPSubQ}$. Here the counterexamples given are $100i + 3$ for a language which contains $100i + 3$ (with the nearest positive example being $100i + 2$). If a query is made for a language which does not contain $100i + 3$, but contains $100i + 2$, then the answer is ‘yes’. When a query is made for a language with the minimal element being $> 100i + 3$, then the negative counterexample is the least element present in the queried language ($100j + 1$ or $100j + 2$ for some $j > i$), with the nearest positive example being $100j + 3$.

(c) \mathcal{L} in the proof of Theorem 10 can be shown to be in **NPMemQ**², as one can first query 0. If 0 is in the target language, then L must be the target

language. Otherwise the nearest positive example is $100i + 1$ for some i . Now querying $100i + 2$ determines whether the target language is L_i or X_i .

(d) Suppose M is a **NPMemQ**¹-learner for \mathcal{L} . Suppose the element queried by M is y . Let A_y be the language in \mathcal{L} which contains y (note that there must be unique such A_y , as M **NPMemQ**¹-learns \mathcal{L}). For $i \neq y$, let A_i denote the language L in \mathcal{L} , if any, such that $y \notin L$ and i is the nearest element (to y) in L . Note that there exists at most one such A_i in \mathcal{L} (otherwise, M cannot **NPMemQ**¹-identify \mathcal{L}). Furthermore, each member of \mathcal{L} is equal to A_i for some i . Also note that, for all $i \neq y$, for all j which are nearer to y than i , if A_i and A_j are both defined then $A_j \not\subseteq A_i$ (as $j \in A_j - A_i$).

Note that, if A_i (for $i \neq y$) is defined, then M , when its membership query y is answered ‘no’ with i given as the nearest positive example, outputs A_i as its conjecture (otherwise, M does not **NPMemQ**¹-identify \mathcal{L}).

Now the **NPLSubQ** learning algorithm for \mathcal{L} is given as follows:

1. For $i = y, y - 1, y + 1, y - 2, y + 2, \dots, 0, 2y$ do:
 - If A_i is defined and A_i is a subset of the target language, then output A_i as the conjecture.
 End For
2. If none of the above A_i are subset of the target language, then let z be the nearest positive example to the counterexample corresponding to the query A_y made in step 1. Conjecture A_z (if defined) (note that one can find A_z , if defined, using M).

Note that the number of queries made by the above **NPLSubQ** learner is bounded by $2y + 1$. Clearly, if A_y is the target language then step 1 correctly identifies A_y . If A_i is the target language where $i \leq 2y$ and $i \neq y$, then $A_j \not\subseteq A_i$ for any j which is nearer to y than i (see the discussion before the algorithm), and thus step 1 correctly identifies A_i .

On the other hand if $i > 2y$, then none of A_i , $i \leq 2y$, is a subset of A_i . Thus step 2 would correctly identify A_i . This completes the proof for part (d).

(e), (f), (h) are shown in Theorem 30.

(g), (i) follow from Propositions 16 and 17. ■

Now we will compare learners using queries of the same type. Our next result shows that one subset query providing the least counterexample can sometimes help learners more than subset queries returning arbitrary counterexamples,

even if the nearest positive examples are returned, and a text of the target language is accessible.

Theorem 22 $\text{LSubQ}^1 - (\text{TtxtNPSubQ} \cup \text{TtxtBNPSubQ}) \neq \emptyset$.

PROOF. Proof of Theorem 3 witnesses this too. ■

The next result establishes hierarchies on the number of queries. We show that, for both types of queries, $k + 1$ queries help learning more than k (even if a learner using k queries gets additional feedback and has access to full positive data).

Theorem 23 For all $k \in \mathbb{N}$,

(a) $\text{SubQ}^{k+1} - (\text{TtxtNPLSubQ}^k \cup \text{TtxtBNPLSubQ}^k) \neq \emptyset$.

(b) $\text{MemQ}^{k+1} - (\text{TtxtNPMemQ}^k \cup \text{TtxtBNPMemQ}^k) \neq \emptyset$.

PROOF. Let $L_0 = \{2x \mid x \in \mathbb{N}\}$.

For $i > 0$, let $L_i = L_0 \cup \{2i + 1\}$.

Let $\mathcal{L} = \{L_i \mid i \leq k + 1\}$.

$\mathcal{L} \in \text{SubQ}^{k+1}$ (MemQ^{k+1}) follows from Proposition 16.

Also, $\mathcal{L} \notin (\text{TtxtNPLSubQ}^k \cup \text{TtxtBNPLSubQ}^k)$, as one can give a text for the target language L_0 , and answer all subset queries for L_i , $1 \leq i \leq k + 1$, as ‘no’ (with the least negative counterexample as $2i + 1$, and the (bounded) nearest positive example as $2i$). If the learner conjectures a grammar for L_0 after asking at most k queries, then there exists an i , $1 \leq i \leq k + 1$, such that the learner did not query L_i . But then the answers are all consistent with the target language being L_i or L_0 , and the portion of the input text read could be extended to be a text for L_i . Thus, the learner fails to identify L_i . Similarly, $\mathcal{L} \notin (\text{TtxtNPMemQ}^k \cup \text{TtxtBNPMemQ}^k)$. ■

For both types of queries, uniformly bounded number of queries is not enough to achieve full learning capability.

Theorem 24 (a) $\text{SubQ} - \bigcup_{k \in \mathbb{N}} (\text{TtxtNPLSubQ}^k \cup \text{TtxtBNPLSubQ}^k) \neq \emptyset$.

(b) $\text{MemQ} - \bigcup_{k \in \mathbb{N}} (\text{TtxtNPMemQ}^k \cup \text{TtxtBNPMemQ}^k) \neq \emptyset$.

PROOF. Let $\text{INIT} = \{L \mid (\exists i)[L = \{x \mid x \leq i\}]\}$. $\text{INIT} \in \text{SubQ}$, as one could find the least i such that $\{x \mid x \leq i + 1\} \not\subseteq$ target language by using subset queries. Similarly, $\text{INIT} \in \text{MemQ}$, as one could find the least i such that

$i + 1 \notin$ the target language.

However, $\text{INIT} \notin (\mathbf{TxtNPLSubQ}^k \cup \mathbf{TxtBNPLSubQ}^k)$. Suppose by way of contradiction that some learner $M \in \mathbf{TxtNPLSubQ}^k$ or $\mathbf{TxtBNPLSubQ}^k$ identifies INIT .

Suppose we always answer the queries of M as ‘yes’. If M outputs a conjecture on some text for some language in \mathcal{L} , then let Q be the set of queries asked and let σ be the initial segment of the text which has been read by M by the time it outputs its conjecture. Otherwise, let T be a text for some language in \mathcal{L} on which M asks maximal number of questions, let Q be the set of questions asked on this text, and let σ be the initial segment of the text which has been read by M by the time it asks its last question. Suppose m is such that $m > \text{content}(\sigma)$, and $m >$ the largest element contained in any of the queries. Then, M would have the same behaviour for any of the target languages $\{x \mid x \leq i\}$, $i \geq m$, where the input text provided starts with σ . Thus, M can identify at most one language of the form $\{x \mid x \leq i\}$, $i \geq m$, even though \mathcal{L} contains infinitely many such languages.

Similarly, $\text{INIT} \notin \mathbf{TxtNPMemQ}^k \cup \mathbf{TxtBNPMemQ}^k$. ■

The next theorem compares the usefulness of the nearest and the bounded nearest positive examples for learners using the same type of queries. First we show that one subset query returning negative counterexample and the nearest positive example of either type can sometimes help a learner more than using subset queries, least counterexamples, nearest positive examples of the other type, and full positive data.

Theorem 25 $\mathbf{NPSubQ}^1 - \mathbf{TxtBNPLSubQ} \neq \emptyset$.

$\mathbf{BNPSubQ}^1 - \mathbf{TxtNPLSubQ} \neq \emptyset$.

PROOF. Proofs of Theorems 5 and 7 witness this too. ■

For membership queries, the picture is more complex. One *unbounded* nearest positive example can sometimes help a learner more than any number of bounded nearest positive examples — even in the presence of full positive data. However, the learners making membership queries and getting the *bounded* nearest positive examples can be simulated by learners using a finite number of just simple membership queries; still, no uniform bound on the number of queries in such a simulation is possible (even if the learner receives also the nearest positive examples and has access to full positive data) — moreover, if the learner using just one bounded positive example has also access to full positive data, then no above simulation is possible at all.

Theorem 26 (a) $\text{NPMemQ}^1 - \text{TxtBNPMemQ} \neq \emptyset$.

(b) For $k \in \mathbb{N}$, $\text{BNPMemQ}^k \subseteq \bigcup_{r \in \mathbb{N}} \text{MemQ}^r$.

(c) For $k \in \mathbb{N}$, $\text{BNPMemQ}^1 - \text{TxtNPMemQ}^k \neq \emptyset$.

(d) $\text{TxtBNPMemQ}^1 - \text{TxtNPMemQ} \neq \emptyset$.

PROOF. (a) Proof of Theorem 12 also witnesses this.

(b) Follows from Propositions 16 and 17.

(c) Let $d_0 = 1$, and $d_{i+1} = 2d_i + 1$. (Thus, $d_i = 2^{i+1} - 1$, for all $i \in \mathbb{N}$). Let $m = d_{2^{k+1}}$.

For $i \leq 2^{k+1}$, let $L_i = \{m - d_j \mid i \leq j \leq 2^{k+1}\} \cup \{m\}$.

Let $\mathcal{L} = \{L_i \mid i \leq 2^{k+1}\}$. Now, $\mathcal{L} \in \text{BNPMemQ}^1$, as one could query $m - d_0$. If $m - d_0$ is in the target language, then the target language must be L_0 . Otherwise, the bounded nearest positive example is $m - d_i$ for some i , and the target language must be L_i .

To see that $\mathcal{L} \notin \text{TxtNPMemQ}^k$, suppose by way of contradiction otherwise. Then, we can maintain an interval $[l_j, u_j]$, such that after the j -th query, for $l_j \leq i \leq u_j$, L_i is consistent with the answers given so far. Intuitively, $m - d_r$ is closer to m than to $m - d_{r+1}$ — thus one can answer a query for $m - d_r$ ‘yes’ (if r is closer to u_j) and ‘no’ (if r is closer to l_j , with the nearest positive example being m); this allows one to maintain at least half of the previous possibilities as consistent with the new answer.

Initially, let $l_0 = 0$ and $u_0 = 2^{k+1}$. After the j -th query (where $j < k$), extend the portion of the input text already read to contain all elements of L_{u_j} . If the $(j + 1)$ -th query (if any) is

— for an element $m - d_r$, where $r < l_j + (u_j - l_j)/2$, then the answer is ‘no’ with the nearest positive example being m ; $l_{j+1} = l_j + (u_j - l_j)/2$ and $u_{j+1} = u_j$.

— for an element $m - d_r$, where $r \geq l_j + (u_j - l_j)/2$, then the answer is ‘yes’; $l_{j+1} = l_j$ and $u_{j+1} = l_j + (u_j - l_j)/2$.

Each query halves the difference between l_j and u_j . Thus when the learner makes its conjecture, after making (at most) k queries, the difference between l_j and u_j is non-zero, and thus there are more than one possible target language consistent with the answers given and the portion of the text read. Thus the learner cannot TxtNPMemQ^k -identify \mathcal{L} .

(d) The class \mathcal{L} used in proof of Theorem 13 also belongs to TxtBNPMemQ^1

(as the input text allows one to get an i such that $100i$ is in the language; now querying $100i + 6$ allows one to determine if the target language is A_i , L_i or X_i). ■

Theorem 2 showed that $\mathbf{TxtFIN} \subseteq \mathbf{MemQ} \cap \mathbf{SubQ}$. Now we show that no uniformly bounded number of queries of either type suffices for a simulation of full positive data.

Theorem 27 $\mathbf{TxtFIN} - \bigcup_{k \in \mathbb{N}} (\mathbf{NPLSubQ}^k \cup \mathbf{BNPLSubQ}^k \cup \mathbf{NPMemQ}^k \cup \mathbf{BNPMemQ}^k) \neq \emptyset$.

PROOF. Let $\mathcal{L} = \{L \mid (\exists i > 0)(\exists D \mid \text{card}(D) = i)[L = \{0\} \cup \{\langle i, x \rangle + 1 \mid x \in D\}]\}$.

Clearly, $\mathcal{L} \in \mathbf{TxtFIN}$, and \mathcal{L} can be learned using finitely many queries of \mathbf{MemQ} or \mathbf{SubQ} type. However, it cannot be learned using at most k -queries of any type, when text is not provided to the learner. To see this, suppose by way of contradiction that M \mathbf{NPMemQ}^k -identifies \mathcal{L} (proof for other cases is similar). Then, answer all queries, except for 0, as ‘no’ with the nearest positive example being 0. Let Q be the set of questions asked in the above situation (before M makes a conjecture, if any). Let m be the maximal element in Q . Then M identifies at most one language of the form, $\{0\} \cup \{\langle i, x \rangle + 1 \mid x < i\}$, where $\langle i, 0 \rangle > 2m$, even though \mathcal{L} contains infinitely many such languages. ■

7 Complexity Speedup Advantages of One Type of Query over Another

In this section we consider when a class is learnable by using query type QB , but needs a high number of queries, whereas if we had used query type QA , then a small number of queries suffice.

Ideally, we would like theorems of the following types:

- (a) $QA^1 \cap QB$ diagonalizes against $\bigcup_{k \in \mathbb{N}} (\mathbf{TxtBNPQB}^k \cup \mathbf{TxtNPQB}^k)$.
- (b) $QA^1 \cap QB^{k+1}$ diagonalizes against $\mathbf{TxtBNPQB}^k \cup \mathbf{TxtNPQB}^k$.

That would give us a perfect set of speedup effects. However, this is not always possible, and we get as close to the above as possible (we do not have the best possible results for QB being \mathbf{MemQ} , and QA being \mathbf{SubQ}).

Note also that the results of type (c): QA^1 diagonalizes against $\mathbf{TxtBNPQB} \cup \mathbf{TxtNPQB}$ — have been obtained in the previous section, where possible: see

Theorems 20, 21, and 22.

7.1 *Txt vs Others*

In this subsection, we show that classes witnessing hierarchies on the number of queries made can be learnable by one-shot learners just from a text without any queries.

Theorem 28 *Suppose $k \in \mathbb{N}$.*

$$(a) \text{ \textbf{TxtFIN} } \cap \text{ \textbf{MemQ} }^{k+1} - (\text{ \textbf{NPMemQ} }^k \cup \text{ \textbf{BNPMemQ} }^k) \neq \emptyset.$$

$$(b) \text{ \textbf{TxtFIN} } \cap \text{ \textbf{SubQ} }^{k+1} - (\text{ \textbf{NPLSubQ} }^k \cup \text{ \textbf{BNPLSubQ} }^k) \neq \emptyset.$$

PROOF. (a) Let $L_i = \{2x \mid x \leq k+1\} \cup \{2i+1\}$. Let $\mathcal{L} = \{L_i \mid i \leq k+1\}$. Then, clearly, $\mathcal{L} \in \text{ \textbf{TxtFIN} }$. $\mathcal{L} \in \text{ \textbf{MemQ} }^{k+1}$ follows from Proposition 16. Also, it is easy to verify that $\mathcal{L} \notin (\text{ \textbf{NPMemQ} }^k \cup \text{ \textbf{BNPMemQ} }^k)$, as the membership queries for even numbers $\leq 2(k+1)$ can be answered ‘yes’ and for odd numbers can be answered ‘no’ (the nearest positive examples can be given as $2x$ for the query $2x+1$). After k queries, there are still two languages which would be consistent with the answers.

(b) Proof similar to that in part (a) can be used to show that $\mathcal{L} \in (\text{ \textbf{TxtFIN} } \cap \text{ \textbf{SubQ} }^{k+1}) - (\text{ \textbf{NPLSubQ} }^k \cup \text{ \textbf{BNPLSubQ} }^k)$. ■

7.2 $QA = \text{ \textbf{LSubQ} }$ and $QB = \text{ \textbf{SubQ} }$

In this subsection, we establish speedup advantages of least counterexamples over arbitrary ones, for subset queries.

Theorem 29 (a) $(\text{ \textbf{LSubQ} }^1 \cap \text{ \textbf{SubQ} }^{k+1}) - (\text{ \textbf{TxtNPSubQ} }^k \cup \text{ \textbf{TxtBNPSubQ} }^k) \neq \emptyset.$

$$(b) (\text{ \textbf{LSubQ} }^1 \cap \text{ \textbf{SubQ} }) - \bigcup_{k \in \mathbb{N}} (\text{ \textbf{TxtNPSubQ} }^k \cup \text{ \textbf{TxtBNPSubQ} }^k) \neq \emptyset.$$

PROOF. (a)

$$\text{Let } \text{ \textbf{INITA} }_i = \{2j \mid j \in \mathbb{N}\} \cup \{2j+1 \mid j \leq i\}.$$

$$\text{Let } \mathcal{L} = \{N\} \cup \{\text{ \textbf{INITA} }_i \mid i < 2^k\}.$$

Clearly, $\mathcal{L} \in \text{ \textbf{LSubQ} }^1$ (by querying N ; if N is a subset of the target language, then the target language must be N ; otherwise if the least counterexample is

$2i + 1$, then the target language is $INITA_{i-1}$).

\mathcal{L} can be learned using $k+1$ **SubQ** queries (by doing binary search on $INITA_i$, $i = 0$ to 2^k — where $INITA_{2^k}$ is treated as N) to find the maximal i such that $2i + 1$ is in the target language.

To show that \mathcal{L} cannot be learned using **TxtNPSubQ** ^{k} or **TxtBNPSubQ** ^{k} , one can use a technique similar to that used in the proof of Theorem 26(c).

Suppose by way of contradiction that M **TxtNPSubQ** ^{k} -identifies (**TxtBNPSubQ** ^{k} -identifies) \mathcal{L} . Initially let $l_0 = 0, u_0 = 2^k$ (here we treat N as $INITA_{2^k}$). Intuitively, after the j -th query of the learner, all languages $INITA_i$, $l_j \leq i \leq u_j$ are consistent with the answers given so far.

Do the following, until the learner makes its conjecture.

Loop: for $j = 0$ to $k - 1$

Extend the initial portion of the text read by the learner so far, to make it a text for $INITA_{l_j}$, and provide this text to the learner.

Suppose the $(j + 1)$ -th query (if any) of the learner is for $INITA_i$.

If $i \leq (l_j + u_j)/2$, then answer ‘yes’, and let $l_{j+1} = (l_j + u_j)/2$, and $u_{j+1} = u_j$.

Otherwise, answer ‘no’, with $2i + 1$ as the negative counterexample, give $2i$ as the (bounded) nearest positive example, and let $l_{j+1} = l_j$ and $u_{j+1} = (l_j + u_j)/2$.

End Loop

Note that $u_j - l_j \geq 2^{k-j}$, as the difference gets halved after each query. Now, at the time the learner makes its conjecture, if any, it has asked $j \leq k$ queries. Thus, $l_j < u_j$. Hence, the answers given, and the initial portion of the input text read by the learner are consistent with at least two possible target languages $INITA_{l_j}$ and $INITA_{u_j}$. Thus, the learner does not identify at least one of them.

(b) Similar to part (a), except that we use $\mathcal{L} = \{N\} \cup \{INITA_i \mid i \in N\}$. ■

7.3 $QA = \text{MemQ}$ and $QB = (\text{L})\text{SubQ}$

In this section we study speedup advantages of membership queries of various types over subset queries. Our first result shows when a bounded number of membership queries and $k + 1$ subset queries have advantage over k subset queries. In particular, we show that if simple membership queries are used, then r such membership queries, for $2^r \geq k + 2$, are needed to get such an advantage (and this bound cannot be lowered, see Theorem 21(i)). If, in

addition to membership queries, a learner gets more feedback, or has access to a text of the target language, then just one such query can sometimes show speedup advantages over subset queries.

Theorem 30 *Suppose $r, k \geq 0$.*

(a) *Suppose $2^r \geq k + 2$.*

$$(\mathbf{MemQ}^r \cap \mathbf{SubQ}^{k+1}) - (\mathbf{TxtNPLSubQ}^k \cup \mathbf{TxtBNPLSubQ}^k) \neq \emptyset.$$

$$(b) (\mathbf{BNPMemQ}^1 \cap \mathbf{SubQ}^{k+1}) - (\mathbf{TxtNPLSubQ}^k \cup \mathbf{TxtBNPLSubQ}^k) \neq \emptyset.$$

$$(c) (\mathbf{NPMemQ}^1 \cap \mathbf{SubQ}^{k+1}) - (\mathbf{TxtNPLSubQ}^k \cup \mathbf{TxtBNPLSubQ}^k) \neq \emptyset.$$

$$(d) (\mathbf{TxtMemQ}^1 \cap \mathbf{SubQ}^{k+1}) - (\mathbf{TxtNPLSubQ}^k \cup \mathbf{TxtBNPLSubQ}^k) \neq \emptyset.$$

PROOF. (a) Consider $k + 2$ languages L_0, L_1, \dots, L_{k+1} , defined as follows.

Let $x_i = 2i + 1$, for $1 \leq i \leq k + 1$.

Let $y_j = 2(k + 2) + 2j + 1$, for $1 \leq j \leq r$.

Let $X = \{x_i \mid 1 \leq i \leq k + 1\}$.

Let $Y = \{y_i \mid 1 \leq i \leq r\}$.

Let $L_0 = \{x \mid x < y_r, x \notin X \cup Y\}$.

For $1 \leq i \leq k + 1$, let $L_i = L_0 \cup \{x_i\} \cup \{y_j \mid b_j = 1, \text{ where } b_1 \dots b_r \text{ is the binary representation of } i\}$.

Let $\mathcal{L} = \{L_i \mid i \leq k + 1\}$.

It is easy to verify that $\mathcal{L} \in \mathbf{MemQ}^r$ (a learner can ask questions about $y_1 \dots y_r$ to determine the target language). $\mathcal{L} \in \mathbf{SubQ}^{k+1}$ follows from Proposition 16.

However, k **LSubQ** queries, even in the presence of a text are not enough. To see this, consider giving a learner a text for L_0 , and answering subset queries based on the target language being L_0 (where negative counterexample would be x_i , for subset queries about L_i , $i > 0$, along with the (bounded) nearest positive example being $x_i - 1$). After at most k questions, the learner has to conjecture a grammar for L_0 . Let i be such that the learner does not ask query for L_i , $i > 0$. Then, the portion of the input text read (before the learner makes its conjecture) can be extended to a text for L_i , and the answers given are consistent with the target language being L_i . Thus, the learner fails to **TxtNPLSubQ** ^{k} -identify (**TxtBNPLSubQ** ^{k} -identify) L_i .

(b) Can be proved similarly to (a), except that in this case we use $y_j = 2(k+2) + j$, and let $L_i = L_0 \cup \{x_i, y_i\}$, for $1 \leq i \leq k+1$.

BNPMemQ¹-identification can be done by querying y_{k+1} (which would give the nearest y_i , if any, which is a member of the target language, and thus allowing one to determine the target language). Proof of $\mathcal{L} \in \mathbf{SubQ}^{k+1}$ and $\mathcal{L} \notin (\mathbf{TxtNPLSubQ}^k \cup \mathbf{TxtBNPLSubQ}^k)$, can be done as in part (a).

(c) The class considered in part(b) belongs to **NPMemQ**¹ too.

(d) Let $L = \{2x \mid x \in N\}$.

For $i \leq k$, let $L_i = L \cup \{2i+1, 2(k+1)+1\}$.

Let $\mathcal{L} = \{L\} \cup \{L_i \mid i \leq k\}$.

$\mathcal{L} \in \mathbf{TxtMemQ}^1$ as one can query $2(k+1)+1$, to find out whether the target language is L or one of L_i . In the latter case, one can just **TxtFIN** identify L_i from text.

Also $\mathcal{L} \in \mathbf{SubQ}^{k+1}$, as \mathcal{L} contains only $k+2$ languages (Proposition 16).

$\mathcal{L} \notin (\mathbf{TxtNPLSubQ}^k \cup \mathbf{TxtBNPLSubQ}^k)$, as the learner must conjecture on an input text being for L , and answers being given based on the target language being L . If L_i is not queried before the conjecture is made (there exists such an L_i , where $i \leq k$), then the answers are consistent with the target language being L_i , and the portion of the input text read by the learner (before it makes its conjecture) can be extended to a text for L_i . ■

The next result shows when, for classes learnable via subset queries, a finite number of membership queries can do more than any uniformly bounded number of subset queries. While one simple membership query does not usually give this sort of speedup advantage, adding access to text, or one extra query and the nearest positive examples provides advantage over the learners using subset queries and least counterexamples (if subset queries return arbitrary counterexamples, rather than the least ones, then just one membership query returning the nearest positive example suffices).

Theorem 31 (a) $\mathbf{TxtMemQ}^1 \cap \mathbf{SubQ} - \bigcup_{k \in N} (\mathbf{TxtNPLSubQ}^k \cup \mathbf{TxtBNPLSubQ}^k) \neq \emptyset$.

(b) $\mathbf{NPMemQ}^1 \cap \mathbf{SubQ} - \bigcup_{k \in N} (\mathbf{TxtNPLSubQ}^k \cup \mathbf{TxtBNPLSubQ}^k) \neq \emptyset$.

(c) $\mathbf{NPMemQ}^2 \cap \mathbf{SubQ} - \bigcup_{k \in N} (\mathbf{TxtNPLSubQ}^k \cup \mathbf{TxtBNPLSubQ}^k) \neq \emptyset$.

(d) $\mathbf{MemQ} \cap \mathbf{SubQ} - \bigcup_{k \in N} (\mathbf{TxtNPLSubQ}^k \cup \mathbf{TxtBNPLSubQ}^k) \neq \emptyset$.

PROOF. (a): P is a partial recursive function defined later. It will be the case that $1 \leq P(i) \leq i$.

Let F be a recursive function such that $F(0) = 1$, and $F(i + 1) = F(i) + 100(i + 1) + 1$.

Let $L = \{0\} \cup \{F(i) + 2 \mid i \in N\}$.

For $1 \leq j \leq i$, let $A_{i,j} = \{F(i) + 100j + 2\}$.

Let $L_i = \{F(i) + 1\} \cup \{F(i) + 100j \mid 1 \leq j \leq i + 1\}$.

Let $X_i = L_i \cup \{F(i) + 2, F(i) + 100P(i) + 2\}$, if $P(i)$ is defined.

$\mathcal{L} = \{L\} \cup \{A_{i,j} \mid 1 \leq j \leq i\} \cup \{L_i \mid i \in N\} \cup \{X_i \mid P(i) \downarrow\}$.

It is easy to verify that $\mathcal{L} \in \mathbf{TxtMemQ}^1$. First, wait for either 0 or $F(i) + 1$ or $F(i) + 100j + 2$ to appear in the text, for some $1 \leq j \leq i$. If 0 appears in the input text, then L is the target language; if $F(i) + 1$ appears in the input text, then query $F(i) + 2$: if the answer is ‘no’, then the target language is L_i , otherwise the target language is X_i ; if $F(i) + 100j + 2$ appears in the input text, for some $1 \leq j \leq i$, then query $F(i) + 1$ — if the answer is ‘yes’, then the target language is X_i , otherwise the target language is $A_{i,j}$.

For seeing that \mathcal{L} is in **SubQ**, note that one can first query L : if L is a subset of the target language then the target language is L . Otherwise, query L_i , and $A_{i,j}$ one by one until a subset (for parameter i) is found. Then one can query L_i , $A_{i,j}$, for $1 \leq j \leq i$. If L_i is not a subset of the target language, then the target language is one of $A_{i,j}$ (the parameter j can then be easily determined by querying $A_{i,j}$ for $1 \leq j \leq i$). If L_i is a subset of the target language, then if some $A_{i,j}$ is also a subset of the target language, where $1 \leq j \leq i$, then the target language is X_i ; otherwise the target language is L_i .

Let M_0, M_1, \dots denote a recursive enumeration of all **TextNPLSubQ** (respectively, **TextBNPLSubQ**) learners. We now define $P(i)$. Suppose $i = \langle r, s \rangle$. Consider the learner M_r using hypothesis space given by φ_s (that is $\varphi_s(k, x)$ determines whether x belongs to the k -th language in the hypothesis space). Consider M_r on a text for L_i . Note that one can detect from a recursive procedure for the query from the class, which query in the class is being asked. Answer ‘no’ to the query L or the query L_j, X_j , where $j \neq i$, or the query $A_{j,r}$ ’s, with the nearest positive example given based on the target language being L_i . Answer ‘yes’ to the subset query for L_i . If X_i is queried, then $P(i) \uparrow$. If M_r outputs a conjecture and there exists a j such that $A_{i,j}$ has not been queried, before making the conjecture, then let $P(i) = j$, for one such j . Otherwise $P(i)$ is not defined. It is easy to verify that M_r either makes at least i queries (for each of $A_{i,j}$) or fails to identify at least one of L_i or X_i .

Thus, for all $i = \langle r, s \rangle$, M_r fails to $\mathbf{TxtNPSubQ}^{i-1}$ ($\mathbf{TxtBNPSubQ}^{i-1}$)-identify \mathcal{L} using hypothesis space given by φ_s . As every machine has infinitely many copies, part (a) follows.

(b): Similar to part (a), except that we use

$$L = \{0\} \cup \{F(i) + x \mid i \in N, x \in \{0, 4\}\},$$

$$L_i = \{F(i) + 3\} \cup \{F(i) + 100j \mid 1 \leq j \leq i + 1\}, \text{ and}$$

$$X_i = L_i \cup \{F(i) + 2, F(i) + 100P(i) + 2\}, \text{ if } P(i) \text{ is defined.}$$

Now the class is in \mathbf{NPMemQ}^1 , as one could do a membership query for 0. Answer ‘yes’ would imply that the target language is L ; answer ‘no’, with the nearest positive example of the form $F(i) + 2$ would imply that the target language is X_i ; the nearest positive example of the form $F(i) + 3$ would imply that the target language is L_i ; the nearest positive example of the form $F(i) + 100j + 2$, for $1 \leq j \leq i$, would imply that the target language is $A_{i,j}$.

$\mathcal{L} \notin \mathbf{TxtBNPLSubQ}^k$ can be done as in part (a).

$\mathcal{L} \notin \mathbf{TxtNPSubQ}^k$ holds as, except for the query being L , all answers can be done as in part (a), by giving least negative counterexamples. For the query being L : the negative counterexample given is $F(i) + 4$, with the nearest positive example being $F(i) + 3$. The rest of the proof can proceed as in part (a).

(c) \mathcal{L} as defined in part (a) above is in \mathbf{NPMemQ}^2 , as one can first query 0. If 0 is in the target language, then the target language must be L . If 0 is not in the target language, then if the nearest positive example to 0 is $F(i) + 1$, then query $F(i) + 2$ to determine if the target language is L_i or X_i ; on the other hand, if the nearest positive example to 0 is $F(i) + 100j + 2$, then the target language must be $A_{i,j}$.

(d) Note that INIT (as defined in the proof of Theorem 24 (a)) belongs to \mathbf{MemQ} . Thus, part (d) follows from proof of Theorem 24 (a). \blacksquare

The above theorem is optimal, as $\bigcup_{r \in N} \mathbf{BNPMemQ}^r \subseteq \bigcup_{k \in N} \mathbf{SubQ}^k$, (see Theorem 21(g)) and $\mathbf{NPMemQ}^1 \subseteq \bigcup_{k \in N} \mathbf{NPLSubQ}^k$ (see Theorem 21(d)).

7.4 $QA = (\mathbf{L})\mathbf{SubQ}$ and $QB = \mathbf{MemQ}$

The results in this section do not give us the complete picture, and there are some open problems.

First, we show that one simple subset query gives advantage over any uniformly bounded number of membership queries of the strongest type.

Theorem 32 $\text{SubQ}^1 \cap \text{MemQ} - \bigcup_{k \in \mathbb{N}} (\text{TxtNPMemQ}^k \cup \text{TxtBNPMemQ}^k) \neq \emptyset$.

PROOF. Let $L = \{2x \mid x \in \mathbb{N}\}$. Let $L_i = L \cup \{1\} - \{2i\}$. Let $\mathcal{L} = \{L\} \cup \{L_i \mid i \in \mathbb{N}\}$.

$\mathcal{L} \in \text{SubQ}^1 \cap \text{MemQ}$: For the **SubQ** learner, the answer to the query for L gives away the language. For **MemQ** learner, the query of 1 determines whether the language is L or one of the L_i 's; in the latter case, one can sequentially query the even numbers to determine the unique i such that $2i \notin$ target language — which would give L_i as the target language.

We now show that $\mathcal{L} \notin \text{TxtNPMemQ}^k \cup \text{TxtBNPMemQ}^k$. Suppose a learner M is given. Suppose answers given to queries of M are ‘yes’ for all the questions in $L \cup \{1\}$.

If M outputs a conjecture on some text for some language in \mathcal{L} , then let Q be the set of questions asked by M on T , and let σ be the initial segment of the text which has been read by M by the time it outputs its conjecture. Otherwise, let T be a text (for some language in \mathcal{L}) on which M asks maximal number of questions, let Q be the set of questions asked by M on this text, and let σ be the initial segment of the text which has been read by M by the time it asks its last question. Suppose i is such that, $2i > \max(\text{content}(\sigma))$ and $2i >$ any of the queries in Q . Then, M would have the same behaviour (in terms of whether it outputs a conjecture or not, and which conjecture it outputs) for any target language $L_j \in \mathcal{L}$, $j > i$, where the input text provided starts with σ . Thus, M can identify at most one of the languages L_j , $j > i$, even though \mathcal{L} contains infinitely many such languages. ■

Unfortunately, the strongest possible speedup result for subset queries and $k + 1$ membership queries over k —

$\text{SubQ}^1 \cap \text{MemQ}^{k+1} - (\text{TxtNPMemQ}^k \cup \text{TxtBNPMemQ}^k) \neq \emptyset$,

does not hold (this follows from Proposition 17 and Theorem 18). The following theorem tries to obtain some closest possible results: we must either add some extra power to a subset query or learners using $k + 1$ membership queries, or not allow access to full positive data to learners using at most k membership queries.

Theorem 33 For all $k \in \mathbb{N}$,

(a) $\text{LSubQ}^1 \cap \text{MemQ}^{k+1} - (\text{TxtNPMemQ}^k \cup \text{TxtBNPMemQ}^k) \neq \emptyset$.

(b) $\mathbf{SubQ}^1 \cap \mathbf{TxtMemQ}^{k+1} - (\mathbf{TxtNPMemQ}^k \cup \mathbf{TxtBNPMemQ}^k) \neq \emptyset$.

(c) $\mathbf{SubQ}^1 \cap \mathbf{MemQ}^{k+1} - (\mathbf{NPMemQ}^k \cup \mathbf{BNPMemQ}^k) \neq \emptyset$.

PROOF. (a) \mathcal{L} given in Theorem 29(a) is in \mathbf{MemQ}^{k+1} (this can be done by finding the largest $i \leq 2^{k+1} - 1$ such that $2i + 1$ belongs to the target language, in a binary search manner). \mathcal{L} can be shown not to be in $(\mathbf{TxtNPMemQ}^k \cup \mathbf{TxtBNPMemQ}^k)$, using essentially the idea of the diagonalization proof in Theorem 29(a).

(b) Let $L = \{2x \mid x \in N\} \cup \{4x + 1 \mid x \in N\}$.

Let $L_i = L \cup \{4j + 3, 4(i + k + 1) + 3\} - \{4i + 1\}$, where $j = (i \bmod (k + 1))$.

Let $\mathcal{L} = \{L\} \cup \{L_i \mid i \in N\}$.

It is easy to verify that $\mathcal{L} \in \mathbf{SubQ}^1$ (query L ; if a subset, then the target language must be L ; otherwise the counterexample is of the form $4i + 1$, for some i , which implies that the target language is L_i).

Also $\mathcal{L} \in \mathbf{TxtMemQ}^{k+1}$, as one can query $4j + 3$, for $j \leq k$. If none of these are present, then the target language must be L . Otherwise, using the text provided, one can find an i such that $4(i + k + 1) + 3$ is in the target language. Then the target language must be L_i .

On the other hand, \mathcal{L} is not in $\mathbf{TxtNPMemQ}^k$ or $\mathbf{TxtBNPMemQ}^k$. To see this, note that one can give the text for L as input to the learner, and answer all queries based on the target language being L . Let σ be the initial portion of the input text that has been read by the learner at the time it makes its conjecture. Now, at the time the learner makes its conjecture, it has asked at most k queries. Thus, there is a $j < k + 1$, such that the learner has not queried $4j + 3$, and there exists an i such that $j = (i \bmod (k + 1))$ and the learner has neither queried $4(i + k + 1) + 3$ nor $4i + 1$, and $\text{content}(\sigma)$ does not contain $4i + 1$. But then the learner cannot distinguish between the target language being L or L_i , as the data for $4j + 3$ and $4(i + k + 1) + 3$ may be added later to the input text, and $4i + 1$ may be omitted from the input text.

(c) Let $L_i = N - \{2i + 1\}$.

Let $\mathcal{L} = \{N\} \cup \{L_i \mid i \leq k\}$. Then, $\mathcal{L} \in \mathbf{SubQ}^1 \cap \mathbf{MemQ}^{k+1}$: \mathbf{SubQ} learner can query N to determine the target language. If N is a subset of the target language, then the target language must be N . Otherwise, the negative counterexample gives away the target language. $\mathcal{L} \in \mathbf{MemQ}$ follows from Proposition 16.

However, $\mathcal{L} \notin (\mathbf{NPMemQ}^k \cup \mathbf{BNPMemQ}^k)$. To see this, every membership

question is answered ‘yes’. If the query for $2i+1$ is not made, then the answers are consistent with N as well as L_i . ■

Questions about what happens when we consider $(\mathbf{Txt})(\mathbf{NP}/\mathbf{BNP})\mathbf{SubQ}^r \cap (\mathbf{NP}/\mathbf{BNP})\mathbf{MemQ}^{k+1} - \mathbf{Txt}(\mathbf{NP}/\mathbf{BNP})\mathbf{MemQ}^k$ have not been answered optimally. The following gives some partial simulation results, which show why this is not easy.

Here note that $\mathbf{NPSubQ}^1 \cap \mathbf{NPMemQ}^1 - \mathbf{TxtBNPMemQ} \neq \emptyset$, as the class \mathcal{L} of Theorem 12 also belongs to $\mathbf{NPSubQ}^1 \cap \mathbf{NPMemQ}^1$.

A number of further results employ the following remark.

Remark 34 *Suppose $\mathcal{L} \in \mathbf{LSubQ}^k \cap \mathbf{NPMemQ}^m$. Then, \mathcal{L} is finite. This can be shown by induction. For $k = 0$, this is clearly true. Suppose it holds for $k = r$. Then, for $k = r + 1$, suppose the first query asked by the \mathbf{LSubQ}^k learner is A .*

Consider the answers to queries by the \mathbf{NPMemQ}^m learner for \mathcal{L} , when the target language contains $\min(A)$. The nearest positive example to a negatively answered membership query z is $\leq \max(\{2z - \min(A), \min(A)\})$ (as the nearest possible element to z is no further than $\min(A)$). As the \mathbf{NPMemQ}^m learner asks at most m questions before making its conjecture, we immediately have that the number of languages in \mathcal{L} which contain $\min(A)$ is finite.

The set of languages in $\mathcal{L} - \{\emptyset\}$ which do not contain $\min(A)$ is \mathbf{LSubQ}^{k-1} -learnable, and thus is finite, by induction.

Our last result shows that any k subset queries can be simulated by $2^k - 1$ simple membership queries and access to full positive data if it is known that a class is learnable via uniformly bounded number of membership queries of any type.

Corollary 35 *Suppose $k, m \in \mathbb{N}$.*

$$(a) \mathbf{SubQ}^k \cap \mathbf{MemQ}^m \subseteq \mathbf{TxtMemQ}^{2^k-1}.$$

$$(b) \mathbf{SubQ}^k \cap \mathbf{BNPMemQ}^m \subseteq \mathbf{TxtMemQ}^{2^k-1}.$$

$$(c) \mathbf{SubQ}^k \cap \mathbf{NPMemQ}^m \subseteq \mathbf{TxtMemQ}^{2^k-1}.$$

PROOF. Using Proposition 17 and Remark 34, we have that if $\mathcal{L} \in \mathbf{MemQ}^m$ or $\mathcal{L} \in \mathbf{BNPMemQ}^m$ or $\mathcal{L} \in \mathbf{SubQ}^k \cap \mathbf{NPMemQ}^m$, then \mathcal{L} is finite. Corollary now follows using Theorem 19. ■

It is open at present if the above results can be improved to give a complete picture.

8 Conclusion

In this paper, we extended D. Angluin’s model of learning via subset and membership queries, allowing teachers, in addition to just answers ‘no’ or arbitrary counterexamples (as suggested by D. Angluin in her original query model in [Ang88]) to return *least* counterexamples and/or the *nearest* (“correcting”) positive examples together with answer ‘no’ or a counterexample. We explored how different variants of corresponding learning models fair against each other in terms of their general learning capabilities and in terms of their complexity advantages, where the number of queries is used as the complexity measure (in the latter case, possible access to a text for the target language becomes a significant factor, contributing an interesting component to the interplay of different learning tools). Though, in most cases, just one query of one type can help more than any number of queries of another type with the strongest possible feedback, typically even coupled with access to text for the target language, the general picture is more complex — for example, sometimes one query is not enough, while two queries suffice — or one query is enough to achieve advantage (general, or complexity) if a learner has also access to full positive data.

We also studied complexity speedup advantages of using one type of query, when queries of both types are enough to learn a class of languages. Here we do not have a complete picture for the advantages of **SubQ** over **MemQ** (see Section 7.4). One may also consider similar questions regarding complexity speed up advantages of using nearest positive examples over bounded nearest positive examples and vice versa. We have some partial results on this topic, however the general picture seems likely to be complex [JK07].

Our approach to representation of *covert* feedback from a teacher in form of the *nearest positive* examples is, of course, only one of possible ways to address this problem. Still, this model gives us an opportunity to explore and compare the power of different types of data obtained from a teacher during the finite learning process. Of course, since our model uses numeric codes rather than strings, our results cannot be used for immediate practical advice. [BBBD05,BBDT06,TK07a,TK07b] suggested a somewhat different approach, using corrections as extensions of queried strings. In [BBdlHJT07], the authors use corrections at the shortest *edit distance* from the queried strings. However, as far as learning natural languages is concerned, whereas all such types of corrections are more or less natural from syntactical standpoint, they might be still semantically inadequate, as semantics of the correction would typically

heavily depend on the context (for example, the incorrect English word “milb” could be “mill” or “mild” or “mile”, depending on the context). In general, however, it would be interesting to define and explore formalizations of one-shot learnability via queries, where positive feedback would be semantically close to the negative datum, rather than being close based on coding (of course, it would not be an easy proposition to correct a wrong sentence in this case, as such a grammatically incorrect sentence could suggest multiple correct close semantics). Such models may also be interesting in the context of learning some important specific indexed classes, for example, patterns, finite automata, or regular expressions.

Acknowledgements: We thank the anonymous referees for several helpful comments which improved the presentation of this paper.

References

- [Ang80] D. Angluin. Inductive inference of formal languages from positive data. *Information and Control*, 45:117–135, 1980.
- [Ang88] D. Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988.
- [Ang01] D. Angluin. Queries revisited. In *Algorithmic Learning Theory: 12th International Conference (ALT’ 2001)*, volume 2225 of *Lecture Notes in Artificial Intelligence*, pages 12–31. Springer-Verlag, 2001.
- [BBBD05] L. Becerra-Bonache, C. Bibire, and A. H. Dediu. Learning DFA from corrections. In Henning Fernau, editor, *Theoretical Aspects of Grammar Induction (TAGI)*, pages 1–12, 2005. WSI-2005–14.
- [BBdlHJT07] L. Beccera-Bonache, C. de la Higuera, J. C. Janodet, and F. Tantini. Learning balls of strings with correction queries. In J. N. Kok, J. Koronacki, R. Lopez de Montara, S. Matwin, D. Mladenic, and A. Skowron, editors, *European Conference on Machine Learning, 2007*, volume 4701 of *Lecture Notes in Computer Science*, pages 18–29. Springer-Verlag, 2007.
- [BBDT06] L. Becerra-Bonache, A. H. Dediu, and C. Tîrnăucă. Learning DFA from correction and equivalence queries. In Y. Sakakibara, S. Kobayashi, K. Sato, T. Nishino, and E. Tomita, editors, *Grammatical Inference: Algorithms and Applications: 8th International Colloquium, ICGI 2006*, volume 4201 of *Lecture Notes in Artificial Intelligence*, pages 281–292. Springer-Verlag, September 2006.
- [FGJ⁺94] L. Fortnow, W. Gasarch, S. Jain, E. Kinber, M. Kummer, S. Kurtz, M. Pleszkoch, T. Slaman, R. Solovay, and F. Stephan. Extremes in the

- degrees of inferability. *Annals of Pure and Applied Logic*, 66:231–276, 1994.
- [GL08] W. Gasarch and A. Lee. Inferring answers to queries. *Journal of Computer and System Sciences*, 74:490–512, 2008.
- [Gol67] E. M. Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.
- [GS91] W. Gasarch and C. Smith. Learning via queries. *Journal of the ACM*, pages 649–674, 1991.
- [JK06] S. Jain and E. Kinber. Iterative learning from positive data and negative counterexamples. In J. L. Balcazar, P. Long, and F. Stephan, editors, *Algorithmic Learning Theory: 17th International Conference (ALT' 2006)*, volume 4264 of *Lecture Notes in Artificial Intelligence*, pages 154–168. Springer-Verlag, 2006.
- [JK07] S. Jain and E. Kinber. One-shot learners using negative counterexamples and nearest positive examples. Technical Report TRA3/07, School of Computing, National University of Singapore, 2007.
- [JK08] S. Jain and E. Kinber. Learning languages from positive data and negative counterexamples. *Journal of Computer and System Sciences*, 74(4):431–456, 2008. Speical Issue: Carl Smith memorial issue.
- [JLZ07] S. Jain, S. Lange, and S. Zilles. A general comparision of language learning from examples and from queries. *Theoretical Computer Science A*, 387(1):51–66, 2007. Special Issue on Algorithmic Learning Theory, 2005.
- [KS96] M. Kummer and F. Stephan. On the structure of degrees of inferability. *Journal of Computer and System Sciences*, 52(2):214–238, 1996.
- [LZ04] S. Lange and S. Zilles. Formal language identification: Query learning vs Gold-style learning. *Information Processing Letters*, 91:285–292, 2004.
- [LZ05] S. Lange and S. Zilles. Relations between Gold-style learning and query learning. *Information and Computation*, 203:211–237, 2005.
- [LZZ08] S. Lange, T. Zeugmann, and S. Zilles. Learning indexed families of recursive languages from positive data: A survey. *Theoretical Computer Science A*, 397(1–3):194–232, 2008. Special Issue on Forty Years of Inductive Inference. Dedicated to the 60th Birthday of Rolf Wiehagen.
- [MPS92] W. Gasarch M. Pleszkoch and R. Solovay. Learning via queries to $[+,.]$. *Journal of Symbolic Logic*, 57:53–81, 1992.

- [Rog67] H. Rogers. *Theory of Recursive Functions and Effective Computability*. McGraw-Hill, 1967. Reprinted by MIT Press in 1987.
- [RP99] D. L. T. Rohde and D. C. Plaut. Language acquisition in the absence of explicit negative evidence: how important is starting small? *Cognition*, 72:67–109, 1999.
- [TK07a] C. Tîrnăucă and T. Knuutila. Polynomial time algorithms for learning k-reversible languages and pattern languages with correction queries. In M. Hutter, R. Servedio, and E. Takimoto, editors, *Algorithmic Learning Theory: 18th International Conference (ALT' 2007)*, volume 4754 of *Lecture Notes in Artificial Intelligence*, pages 272–284. Springer-Verlag, 2007.
- [TK07b] C. Tîrnăucă and S. Kobayashi. A characterization of the language classes learnable with correction queries. In J. Y. Cai, S. B. Cooper, and H. Zhu, editors, *Theory and Applications of Models of Computation, Fourth International Conference, TAMC 2007, Proceedings*, volume 4484 of *Lecture Notes in Computer Science*, pages 398–407. Springer-Verlag, 2007.
- [ZL95] T. Zeugmann and S. Lange. A guided tour across the boundaries of learning recursive languages. In K. Jantke and S. Lange, editors, *Algorithmic Learning for Knowledge-Based Systems*, volume 961 of *Lecture Notes in Artificial Intelligence*, pages 190–258. Springer-Verlag, 1995.