THE NATIONAL UNIVERSITY
of SINGAPORE

School of Computing
Lower Kent Ridge Road, Singapore 119260

TRA3/07

One-shot Learners Using Negative Counterexamples and Nearest Positive Examples□□

Sanjay JAIN and Efim KINBER

March 2007

Technical Report

Foreword

This technical report contains a research paper, development or tutorial article, which has been submitted for publication in a journal or for consideration by the commissioning organization. The report represents the ideas of its author, and should not be taken as the official views of the School or the University. Any discussion of the content of the report should be sent to the author, at the address shown on the cover.

JAFFAR, Joxan
Dean of School

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 117543. 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.

1 Introduction

There are two major formal models that have been used over the years to address various aspects of human learning: 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 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 Gold's model, the 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

[1] Supported in part by NUS grant number R252-000-212-112.

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,JLZ05]). 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**.

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 a 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 [JK07,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 [JK07,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 all positive integers, 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**.

— in the above approach, a teacher may have difficulty 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 superset 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]; 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 models can work differently 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.

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 same classes of languages more efficiently than the other type, where efficiency is measured in terms of number of queries made during the learning process. In the latter context, unlike the case when the number of queries is not bounded, if a learner can have access to (potentially) all positive data, this can significantly affect its learning power. Therefore, we also consider positive data (*texts*) as an additional factor for learners with a uniform bound on the number of queries.

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 the language does not enhance the capabilities of a learner in any of our models. Thus, in cases where we compare learning power of different models, we can assume that the learner does not receive the 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 if the teacher, in addition to

arbitrary negative counterexample, provides the nearest positive example to the 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;

(d) the nearest (and the bounded nearest) positive examples provided to learners using membership queries can help learn classes of languages that cannot be learned using membership queries alone, even if full positive data of the target concept is available (in other words, for such learners, getting the nearest positive examples at "right time" can sometimes provide more power than access to full positive data).

For all the results above, examples of exhibited classes are such that the learners on the negative sides cannot be helped even if they have access to full positive data representing the target concept.

In Section 4 we consider indexable classes of recursive languages. In Section 5, we give an example, showing how the results in Section 4 can be translated to recursively enumerable classes of recursively enumerable languages; this example (and other corresponding results) requires somewhat more complex technique that those in Section 4.

In Section 6 we primarily study the following problem: when a type of queries $Q1$ does or does not give advantage over a type of queries $Q2$ when the number of queries of the type $Q1$ is uniformly bounded? Our main results can be summarized as follows:

(a) one subset query providing the least counterexample can have more learning power 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 have more learning power 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 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* positive example can have more learning power than a prefixed bounded number of subset queries returning least negative counterexamples and the nearest positive examples — even in 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 $Q2$, can one speed up the learning process (in terms of usage of number of queries) by using query type $Q1$? We address questions such as when classes which are learnable using small number of queries of a type $Q1$, require arbitrarily large number of queries of a type $Q2$. For example, we address questions about existence of classes which can be learned using 1 query of a type $Q1$, or a finite number of queries of a type $Q2$ ($k+1$ queries of the type $Q2$), but cannot be learned using bounded number of queries of the type $Q2$ (k queries of the type $Q2$). Section 8 is dedicated entirely to the study of this problem for the cases when both types $Q1$ and $Q2$ involve returning the unbounded or bounded nearest positive examples.

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 9, Conclusion, is devoted to discussion of our results and possible directions for future re-

search.

2 Notation and Preliminaries

Any unexplained recursion theoretic notation is from [Rog67]. The symbol N denotes the set of natural numbers, $\{0, 1, 2, 3, \ldots\}$. Symbols $\emptyset$, $\subseteq$, $\subset$, $\supseteq$, and $\supset$ denote empty set, subset, proper subset, superset, and proper superset, respectively. $D_0, D_1, \ldots$, denotes a canonical recursive indexing of all the finite sets [Rog67, Page 70]. We assume that if $D_i \subseteq D_j$ then $i \leq j$ (the canonical indexing defined in [Rog67] satisfies this property). 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 $\min(\emptyset) = \infty$. χ_L denotes the characteristic function of L.

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. We define $\pi_1(\langle x, y \rangle) = x$ and $\pi_2(\langle x, y \rangle) = y$. We can extend pairing function to multiple arguments by using $\langle i_1, i_2, \ldots, i_k \rangle = \langle i_1, \langle i_2, \langle \ldots, \langle i_{k-1}, i_k \rangle \rangle \rangle \rangle$.

By φ we denote an acceptable numbering of all partial recursive functions [Rog67]. φ_i denotes the partial recursive function computed by program i in the φ system. $\mathcal{R}$ denotes the class of all recursive functions. Φ denotes an arbitrary Blum-Complexity measure [Blu67] for the φ system. W_i denotes $\text{domain}(\varphi_i)$. Symbol $\mathcal{E}$ will denote the set of all r.e. languages. Symbol L, with or without decorations, ranges over $\mathcal{E}$. By $\overline{L}$, we denote the complement of L, that is $N - L$. Symbol $\mathcal{L}$, with or without decorations, ranges over subsets of $\mathcal{E}$. $W_{i,s} = \{x \mid x < s, \Phi_i(x) < s\}$.

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 $x \in L_i$.

Let $\text{INIT} = \{L \mid (\exists i)[L = \{x \mid x \leq i\}]\}$. Let FINITE denote the class of all finite languages.

We now present some concepts from language learning theory. A *sequence* σ is a mapping from an initial segment of N into $(N \cup \{\#\})$. The empty sequence

is denoted by Λ. The *content* of a sequence σ, denoted content(σ), 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. So, $|\Lambda| = 0$.

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

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 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 and stops. If this conjecture is correct, 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 from where 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 there exists a hypothesis space $H_0, H_1, \ldots$, representing all the languages in an indexed family $\mathcal{L}$ such that

(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 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, 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 hypotheses space $H_0, H_1, \ldots$, where $H_0, H_1, \ldots$ form an indexed family and $\{H_0, H_1, \ldots\} = \mathcal{L}$.

The subset queries are now restricted to the form "$H_i \subseteq$ input language?". Correspondingly, the membership queries are also assumed to come from the hypotheses space: the learner only asks membership queries of the form "$x \in$ input 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 one can obtain all positive and negative data (so-called *informant*) for the input language, and thus the model (including the cases for the (bounded) nearest positive examples) would collapse to learning from informant, 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 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 input language), the teacher also provides a negative counterexample, which is a member of H_i, but not a member of the input 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 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 negative element, in the context of membership queries) is provided (this is denoted by using the prefix **BNP** to the query type).

In addition text may or may not be provided to the learner: this is denoted by using **Txt** in the criterion name.

The above will provide us with the the following criteria for one-shot learnability: **SubQFIN**, **LSubQFIN**, **MemQFIN NPSubQFIN**, **NPLSubQFIN**, **NPMemQFIN BNPSubQFIN**, **BNPLSubQFIN**, **BNPMemQFIN**

and **SubQTxtFIN**, **LSubQTxtFIN**, **MemQTxtFIN NPSubQTxtFIN**, **NPLSubQTxtFIN**, **NPMemQTxtFIN BNPSubQTxtFIN**, **BNPLSubQTxtFIN**, **BNPMemQTxtFIN**.

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

Definition 1 (a) **M NPLSubQFIN**-identifies $\mathcal{L}$, iff for some class preserving hypothesis space $H_0, H_1, \ldots$, for all $L \in \mathcal{L}$, **M** asks a sequence of subset queries, where the answer to each query H_i 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 $x-1, x+1, x-2, x+2, \ldots$ which belongs to L (if there is no such y, then a special answer 'none' is provided to the learner).

After asking a finite number of queries, the learner outputs an index i such that $H_i = L$.

(b) **NPLSubQFIN** $= \{\mathcal{L} \mid$ some **M NPLSubQFIN**-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, **NPMemQ**^{k}**TxtFIN** denotes the criterion **NPMemQTxtFIN** where the number of queries made by the learner is limited to k.

4 Relationships among various query criteria

4.1 Providing Text

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

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

QFIN = **QTxtFIN**.

NPQFIN = **NPQTxtFIN**.

BNPQFIN = **BNPQTxtFIN**.

PROOF. We only show **SubQFIN** = **SubQTxtFIN**. Other parts can be proved similarly.

Suppose **M** **SubQTxtFIN**-identifies $\mathcal{L}$. Then, define $\mathbf{M}'$ not having access to the positive data as follows. It searches for a finite segment σ and a conjecture A made by **M** on σ such that content$(\sigma) \subseteq$ input language, where the answers to the queries made by **M** are answered in the same way as received by $\mathbf{M}'$ for the same queries. If and when such σ, A are found, $\mathbf{M}'$ 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. ∎

4.2 Variants of **SubQ**

In this subsection, we explore relationships between different variants of **SubQ**.

First we show that learners getting least counterexamples can sometimes learn more than any learner getting arbitrary counterexamples, as well as the nearest positive examples, bounded or not, and access to full positive data.

Theorem 3 **LSubQFIN** $-$ (**NPSubQTxtFIN** $\cup$ **BNPSubQTxtFIN**) $\neq \emptyset$.

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

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

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

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 $\mathcal{L} \in$ **LSubQFIN**, first query L. If L is a subset of the input language, then the input language must be L. Otherwise, the least counterexample is either $100i + 1$ or $100i + 3$, for some i. In the former case, the input language must be L_i. In the latter case, the input language must be X_i.

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

On input i
1. Simulate **M** with (a text for L_i) and answers to the queries being as follows.

2. For queries which contain $100i+3$, answer 'no' along with counterexample $100i+3$, and the (bounded) nearest positive example being $100i+2$.
   For queries which do not contain $100i+3$, answer 'yes'.
3. 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 NPSubQTxtFIN**-identifies (**BNPSubQTxtFIN**-identifies) $\mathcal{L}$, the above algorithm must halt (either via **M** querying a language which contains $100i+1$ but not $100i+3$, or via **M** eventually outputting a conjecture). In the step 3, when our algorithm outputs '$i \in K$', 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 $i \notin K$, then since the answers given to the queries of **M**, in the simulation above, are all consistent with the input being L_i or X_i, we have that **M** must have output a correct conjecture. This is only possible if $X_i \notin \mathcal{L}$, and thus $i \notin K$. ∎

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

Remark 4 *Suppose $\mathcal{L}$ contains 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 input language being either of L_i or X_i. Then the class is not finitely learnable (based on the corresponding query type Q).*

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 do sometimes more than the ones getting the bounded nearest positive example, even if the latter ones receive the least counterexamples and have access to full positive data.

Theorem 5 **NPSubQFIN** $-$ **BNPLSubQTxtFIN** $\neq \emptyset$.

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 $\mathcal{L} \in$ **NPSubQFIN**. First, a learner can query L. If L is contained in the input language, then the input language must be L. If there is a counterexample, it must be $100i+1$. Now the input language is X_i if the nearest positive example was $100i + 2$. Otherwise, the input language must be L_i.

We now show that $\mathcal{L} \notin$ **BNPLSubQTxtFIN**. We use Remark 4, as one can answer queries (except for X_i) of a supposed learner $\mathbf{M}$ based on input language being L_i (these answers will be consistent with input language being X_i also). Thus, by using Remark 4, we have that $\mathcal{L} \notin$ **BNPLSubQTxtFIN**. ∎

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

Corollary 6 **SubQFIN** $\subset$ **NPSubQFIN**.
LSubQFIN $\subset$ **NPLSubQFIN**.

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 **BNPSubQFIN**$-$ **NPLSubQTxtFIN** $\neq \emptyset$.

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 $\mathcal{L} \in$ **BNPSubQFIN**, a learner can query L. If L is contained in the input language, then the input language must be L. If there is a counterexample, it must be $100i + 3$, for some $i \in N$. Now the input language is

X_i, if the bounded nearest positive example is $100i + 1$. Otherwise the input language must be L_i.

We now show that $\mathcal{L} \notin$ **NPLSubQTxtFIN**. We use Remark 4, as one can answer queries (except for X_i) of a supposed learner **M** based on input language being L_i (these answers will be consistent with input language being X_i also). Thus, by using Remark 4, we have that $\mathcal{L} \notin$ **NPLSubQTxtFIN**. ∎

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

Corollary 8 **SubQFIN** $\subset$ **BNPSubQFIN**.
LSubQFIN $\subset$ **BNPLSubQFIN**.

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 **NPSubQFIN** $\cap$ **BNPSubQFIN** $-$ **LSubQTxtFIN** $\neq \emptyset$.

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 $\mathcal{L} \in$ **NPSubQFIN** (**BNPSubQFIN**), a learner can query L. If L is contained in the input language, then the input language must be L. If there is a counterexample, it must be $100i + 2$, for some $i \in N$. Now the input language is X_i, if the (bounded) nearest positive example is $100i + 1$. Otherwise the input language must be L_i.

We now show that $\mathcal{L} \notin$ **LSubQTxtFIN**. We use Remark 4, as one can answer queries (except for X_i) of a supposed learner **M** based on input language being L_i (these answers will be consistent with input language being X_i also). Thus, by using Remark 4, we have that $\mathcal{L} \notin$ **LSubQTxtFIN**. ∎

In this subsection, we compare powers of one-shot learners using membership and subset queries. We establish that, withing the framework of our models, 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 the learners using membership queries can sometimes be stronger than the ones using subset queries.

Theorem 10 **MemQFIN** $-$ (**NPLSubQTxtFIN** $\cup$ **BNPLSubQTxtFIN**) $\neq \emptyset$.

Proof. Let $L = \{0\} \cup \{100i + 2, 100i + 3, 100i + 5 \mid i \in N\}$.

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 $\mathcal{L} \in$ **MemQFIN**. Learner queries 0. If the answer is 'yes', then input language must be L; Otherwise, learner determines an i such that $100i + 1$ is in the input language. Then querying $100i + 2$ determines whether the input language is L_i or X_i.

We now show that $\mathcal{L} \notin$ **NPLSubQTxtFIN** (**BNPLSubQTxtFIN**). 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 least element of the query being the least negative counterexample. The nearest positive example will be $100i + 1$ or $100i + 5$.

Thus, by using Remark 4, we have that $\mathcal{L} \notin$ **NPLSubQTxtFIN** (**BNPLSubQTxtFIN**). ∎

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

Theorem 11 **SubQFIN**$-$(**NPMemQTxtFIN**$\cup$**BNPMemQTxtFIN**) $\neq \emptyset$.

Proof. Let $\mathcal{L} = \{L \mid \text{card}(N - L) \leq 1\}$. It is easy to verify that $\mathcal{L} \in$ **SubQFIN** (learner can query N — if the answer is 'yes', then input language

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

On the other hand, suppose by way of contradiction that some learner **NPMemQTxtFIN** (**BNPMemQTxtFIN**)-identifies $\mathcal{L}$. Let σ be an 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 that, as in the case of subset queries, learners getting the unbounded or bounded nearest positive examples (in addition to answers 'no') can learn more than learners getting the nearest positive example of the other type and access to full positive data.

Theorem 12 $\mathbf{NPMemQFIN} - \mathbf{BNPMemQTxtFIN} \neq \emptyset$.

Proof. Let $L = \{0\} \cup \{100i + 2, 100i + 3 \mid i \in N\}$.

Let $L_i = \{100j + 2, 100j + 3 \mid j \geq i\}$.

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

We now show that $\mathcal{L} \notin \mathbf{BNPMemQTxtFIN}$. We use Remark 4, as one can answer queries x (except for $100i + 1$) of a supposed learner $\mathbf{M}$ based on whether $x \in L_i$. For $x \notin L_i$, if the queried element is $100j + b$, for $b \in \{1, 2, 3\}$, for 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 $100(j - 1) + 3$.

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

Theorem 13 $\mathbf{BNPMemQFIN} - \mathbf{NPMemQTxtFIN} \neq \emptyset$.

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 $\mathcal{L} \in \mathbf{BNPMemQFIN}$. A learner first finds an i such that $100i$ belongs to the input language. Then it queries $100i+6$; if $100i+6$ belongs to the input language, then the input language must be A_i. Otherwise, if the nearest bounded positive example is $100i+4$, then the input language is X_i. Otherwise the input 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 $\mathcal{L} \notin \mathbf{NPMemQTxtFIN}$. We use Remark 4, as one can answer queries (except for $100i+4$) of a supposed learner $\mathbf{M}$ based on membership relation with L_i. Thus, by using Remark 4, we have that $\mathcal{L} \notin \mathbf{BNPMemQTxtFIN}$). ∎

Now we show that the nearest positive examples of either type give sometimes more power to learners over those getting access to full positive data but not getting the nearest positive examples: the nearest positive example, coupled with the answer 'no' to a specific query, and obtained at the "right" time, can be more helpful than access to a text for the target language.

Theorem 14 $\mathbf{NPMemQFIN} \cap \mathbf{BNPMemQFIN} - \mathbf{MemQTxtFIN} \neq \emptyset$.

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

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

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

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

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

We now show that $\mathcal{L} \notin \mathbf{MemQTxtFIN}$. We use Remark 4, as one can answer queries (except for $100i+2$) of a supposed learner $\mathbf{M}$ based on membership relation with L_i. Thus, by using Remark 4, we have that $\mathcal{L} \notin$

5 RE classes of RE languages

In this section, we demonstrate that one can generalize Theorem 5 to the case when available numbering of target concepts is an r.e. numbering of r.e. languages, and, thus, queries with r.e. indices for the language queries are allowed. Similar ideas can be used for the rest of our theorems in Section 4.

Theorem 15 **NPSubQFIN – BNPLSubQFIN** $\neq \emptyset$.

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

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 the indexed family.

$\mathcal{L} \in$ **NPSubQFIN**: One can first ask the question whether $L \subseteq$ input language. If so, the input 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 subset of input language, but $L_{i,2,j+1}$ is not a subset of the input language — then the input 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 input language must be $L_{i,1,0}$.

We now show that $\mathcal{L} \notin$ **BNPLSubQFIN**. For this, we define the partial recursive function φ_e as follows.

$\varphi_e(i)\downarrow$ iff $\mathbf{M}_i$ eventually outputs a conjecture when its questions "W_m subset of input language?" are answered as follows (for the first case which applies, based on some standard 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 $\mathbf{M}_i$'s question is not answered, and $\varphi_e(i)$ will diverge.

We now consider the following cases.

Case 1: $\mathbf{M}_i$ 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 $\mathbf{M}_i$ asks a query outside the class $\mathcal{L}$, or the answers to $\mathbf{M}$ are consistent with the input language being $L_{i,1,0}$. However, $\mathbf{M}_i$ does not output any conjecture and, thus, does not **BNPLSubQFIN**-identify $L_{i,1,0}$.

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

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 often does not work when we are having only constant number of queries).

We begin with the following useful proposition.

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

PROOF. Suppose $\mathcal{L}$ consists of only $k+1$ languages. To show that $\mathcal{L} \in \mathbf{SubQ}^k\mathbf{FIN}$, 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 input language (if the answer to the query is 'no'), or determine that the input 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 input language. Thus, k queries can eliminate k of the $k+1$ languages as potential candidate for the input 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 $k \in N$.*

(a) If $\mathcal{L} \in \mathbf{MemQ}^k\mathbf{FIN}$, then $\mathrm{card}(\mathcal{L}) \leq 2^k$.

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

PROOF. (a) Immediate, as the answers to the k possible questions by $\mathbf{MemQ}^k\mathbf{FIN}$ learner determine the language.

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

Our next two results show how one-shot learners using uniformly bounded 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 input language is provided.

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

PROOF. Suppose $\mathcal{L}$ is in $\mathbf{SubQ}^1\mathbf{FIN}$. This means that for the first query A asked by a learner, either the input 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 x_L, for $L \in \mathcal{L}$. Now membership query about this x_L can determine if the input language is L or

A. ∎

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

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

PROOF. Note that any class which can be learned using k subset queries has a subset chain of size at most 2^k. Suppose $\mathbf{M}$ $\mathbf{SubQ}^k\mathbf{FIN}$-identifies $\mathcal{L}$. Since the class is finite, there exists a finite subset of N, membership of which in the input language determines the input. Thus, without loss of generality, we can assume that the rest of the inputs do not matter. Thus, we are working in finite domain, finite set of functions, and thus non-effective learning is enough to show effective learning too.

We show the theorem by induction. For base case, if $k = 1$, then, by Theorem 18, $\mathcal{L} \in \mathbf{MemQ}^1\mathbf{TxtFIN}$. For larger k, suppose A is the first question that $\mathbf{M}$ asks. We divide the class $\mathcal{L}$ into two parts: one which contain A, and one which do not. We will take at most 2^{k-1} queries to determine whether A is contained in the input language or not (along with a needed counterexample, in case of A not being a subset of the input). Note that the largest subset chain among languages which are not a subset of A is 2^{k-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 input 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 $B \in Possible$, content$(\sigma) \subseteq B$, and for all $C \in Possible$ which contain content(σ), $B \subseteq C$. (That is: there exists a unique minimal element in $Possible$ which contains σ).
2. If $A \subseteq B$, then one can proceed with the simulation of $\mathbf{M}$ with answer 'yes' to the subset query A made by $\mathbf{M}$. Note that by induction, we can simulate the rest of the queries of $\mathbf{M}$ 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 $\mathbf{M}$ with the answer 'no' to the subset query A made by $\mathbf{M}$ (along with the negative counterexample x). Note that by induction, we can simulate the rest of the queries of $\mathbf{M}$ 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 con-

tain A is reduced by at least 1 in the process (as B is contained in all the languages in the remainder of Possible).

End Loop

By the comment at the end of step 4 above, note that each round of Loop uses one query and reduces subset chain size of 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 a learner using just one subset query can sometimes do more than any learner using the strongest type of membership queries within the framework of our models — including access to a text of the input language.

Theorem 20 $\mathbf{SubQ^1FIN} - (\mathbf{NPMemQTxtFIN} \cup \mathbf{BNPMemQTxtFIN}) \neq \emptyset$.

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 that r membership queries can be simulated by $2^r - 1$ subset queries (we also show that this estimate is tight). However, if, in addition to membership queries, a learner gets either access to text of the input language, or the nearest positive examples, 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 input 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; still, such a simulation is not always possible if a learner can use two such membership queries. 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 *(a)* $\mathbf{MemQ^1TxtFIN} - (\mathbf{NPLSubQTxtFIN} \cup \mathbf{BNPLSubQTxtFIN}) \neq \emptyset$.

(b) $\mathbf{NPMemQ^1FIN} - (\mathbf{NPSubQTxtFIN} \cup \mathbf{BNPLSubQTxtFIN}) \neq \emptyset$.

(c) $\mathbf{NPMemQ^2FIN} - (\mathbf{NPLSubQTxtFIN} \cup \mathbf{BNPLSubQTxtFIN}) \neq \emptyset$.

(d) $\mathbf{NPMemQ^1FIN} \subseteq \bigcup_{r \in N} \mathbf{NPLSubQ^rFIN}$.

(e) For all k, $\mathbf{NPMemQ^1FIN} - (\mathbf{NPLSubQ^kTxtFIN} \cup \mathbf{BNPLSubQ^kTxtFIN}) \neq \emptyset$.

(f) For all k, $\mathbf{BNPMemQ^1FIN} - (\mathbf{NPLSubQ^kTxtFIN} \cup \mathbf{BNPLSubQ^kTxtFIN}) \neq \emptyset$.

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

(h) For $r \geq 1$, $\mathbf{MemQ^rFIN} - (\mathbf{BNPLSubQ^{2^r-2}TxtFIN} \cup \mathbf{NPLSubQ^{2^r-2}TxtFIN}) \neq \emptyset$.

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

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

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

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

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

For a query containing an element $< 100i+1$, one can give counterexample to be the least element $< 100i+1$ in the language (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 $100j+1$ or $100j+2$, where $j > i$, are answered 'no', with $100j+1$ or $100j+2$ being the negative counterexample, and either $100(j-1)+3$ or $100i+2$ being the bounded nearest positive example. 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 **BNPLSubQTxtFIN**-learnability of $\mathcal{L}$ by the learner).

Similarly, $\mathcal{L} \notin$ **NPSubQTxtFIN**. 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 $100j+2$ or $100j+1$ based on the least element present in the language, 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}FIN**, as one can first query 0. If 0 is in the input language, then L must be the input language. Otherwise the nearest positive example is $100i+1$ for some i. Now querying $100i+2$ determines whether the input language is L_i or X_i.

(d) Suppose the queried element by **NPMemQ^{1}FIN** learner is y. Let A be the language in $\mathcal{L}$ which contains y (note that there must be unique such A, as the learner learns the class using only one membership query). For $i \leq 2y$, let A_i be the language in $\mathcal{L}$ which does not contain y, but contains i as the element nearest to y (here, recall that the nearest elements to y have the order $y-1$, $y+1$, $y-2$, $y+2$, ...).

Now the **NPLSubQFIN** learning algorithm is:

1. Query A, A_0, A_1, ..., A_{2y}. Find which of these are subsets of the input language. If none, then proceed to the step 2. Otherwise, the subset sequence uniquely determines the input language. (If A is a subset of the input language, then the input language must be A; otherwise it is A_i, where A_i is a subset of the input language and, among all j such that $j \leq 2y$ and $A_j \subseteq$ input language, i is the nearest element to y; note that this A_i would be the conjecture returned by the **NPMemQ^{1}FIN** learner, when membership query for y is answered 'no', with the nearest positive example being i — no other nearer element to y exists in the input language, as otherwise the conjecture of **NPMemQ^{1}FIN** learner would not be a subset of the input language).

2. If the algorithm reaches this step, then A is not a subset of the input language. The least negative counterexample, to query A, has to be an element $\leq y$. Let the corresponding nearest positive example be z. Note that $z > 2y$ (otherwise step 1 would have handled the case). Give 'no' answer to the membership query for y by the **NPMemQ^{1}FIN** learner, with the nearest positive example being z (since z is also the nearest positive example in the input language for 'no' answer to the membership query for y). Output the conjecture outputted by the **NPMemQ^{1}FIN** learner.

Note that the number of queries made above by the **NPLSubQFIN** learner is bounded by $2y + 2$.

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

(g), (i) Follow from Proposition 17 and Proposition 16. ∎

Now we will compare learners using queries of the same type. Our next result shows that one subset query providing the least counterexample can do more than subset queries returning arbitrary counterexamples, even if the nearest positive examples are returned, and a text of the input language is accessible.

Theorem 22 $\mathbf{LSubQ^1FIN} - (\mathbf{NPSubQTxtFIN} \cup \mathbf{BNPSubQTxtFIN}) \neq \emptyset$.

Proof. Proof of Theorem 3 witness this too. ∎

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

Theorem 23 *For all* $k \in N$,

(a) $\mathbf{SubQ}^{k+1}\mathbf{FIN} - (\mathbf{NPLSubQ}^k\mathbf{TxtFIN} \cup \mathbf{BNPLSubQ}^k\mathbf{TxtFIN}) \neq \emptyset$.

(b) $\mathbf{MemQ}^{k+1}\mathbf{FIN} - (\mathbf{NPMemQ}^k\mathbf{TxtFIN} \cup \mathbf{BNPMemQ}^k\mathbf{TxtFIN}) \neq \emptyset$.

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

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

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

It is easy to verify that $\mathcal{L} \in \mathbf{SubQ}^{k+1}\mathbf{FIN}$ ($\mathbf{MemQ}^{k+1}\mathbf{FIN}$), as one can ask subset query for L_i (membership query for $2i + 1$), $1 \leq i \leq k + 1$. If any of these L_i is a subset of the input language (any of these $2i + 1$ is a member of the input language), then the input language must be L_i. Otherwise, the input language is L_0.

Also, $\mathcal{L} \notin (\mathbf{NPLSubQ}^k\mathbf{TxtFIN} \cup \mathbf{BNPLSubQ}^k\mathbf{TxtFIN})$, as one can give an input text for L, 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 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 input language being L_i or L, and the input read could be extended to be a text for L_i. Thus, the learner fails to identify L_i. Similarly, $\mathcal{L} \notin (\mathbf{NPMemQ}^k\mathbf{TxtFIN} \cup \mathbf{BNPMemQ}^k\mathbf{TxtFIN})$. ∎

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

Theorem 24 *(a)* $\mathbf{SubQFIN} - \bigcup_{k\in N}(\mathbf{NPLSubQ}^k\mathbf{TxtFIN} \cup \mathbf{BNPLSubQ}^k\mathbf{TxtFIN}) \neq \emptyset$.

(b) $\mathbf{MemQFIN} - \bigcup_{k\in N} \mathbf{NPMemQ}^k\mathbf{TxtFIN} \cup \mathbf{BNPMemQ}^k\mathbf{TxtFIN}) \neq \emptyset$.

Proof. INIT $\in$ **SubQFIN**, as one could find the least i such that $\{x \mid x \leq i{+}1\} \not\subseteq$ input language by using subset queries. Similarly, INIT $\in$ **MemQFIN** as one could find the least i such that $i + 1 \notin$ the input language.

However, INIT $\notin$ ($\mathbf{NPLSubQ}^k\mathbf{TxtFIN} \cup \mathbf{BNPLSubQ}^k\mathbf{TxtFIN}$) as one could always answer the query of the learner 'yes'. Then, as the number of queries is bounded, by providing a text for $\{x \mid x \leq m\}$, where m is the largest element in any of the queries asked, one gets that the learner must make a conjecture. But then, the learner identifies at most one extension of $\{x \mid x \leq m\}$.

Similarly, INIT $\notin \mathbf{NPMemQ}^k\mathbf{TxtFIN} \cup \mathbf{BNPMemQ}^k\mathbf{TxtFIN}$. ∎

Next two results compare power 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 do more than any learner using subset queries, least counterexamples, nearest positive examples of the other type, and full positive data.

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

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

Proof. Proofs of Theorems 5 and 7 witness this too. ∎

For membership queries, the picture is more complex. One *unbounded* nearest positive example can do 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)* $\mathbf{NPMemQ^1FIN} - \mathbf{BNPMemQTxtFIN} \neq \emptyset$.

(b) For $k \in N$, $\mathbf{BNPMemQ^kFIN} \subseteq \bigcup_{r \in N} \mathbf{MemQ^rFIN}$.

(c) For $k \in N$, $\mathbf{BNPMemQ^1FIN} - \mathbf{NPMemQ^kTxtFIN} \neq \emptyset$.

(d) $\mathbf{BNPMemQ^1TxtFIN} - \mathbf{NPMemQTxtFIN} \neq \emptyset$.

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

(b) Follows from Proposition 17 and Proposition 16.

(c) Let $d_0 = 1$, and $d_{i+1} = 2d_i + 1$. 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, $\mathcal{L} \in \mathbf{BNPMemQ^1FIN}$, as one could query $m - d_0$. If $m - d_0$ is in the input language, then the input language must be L_0. Otherwise, the bounded nearest positive example is $m - d_i$ for some i, and the input language must be L_i.

To see that $\mathcal{L} \notin \mathbf{NPMemQ^kTxtFIN}$, 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 $u_0 = 2^{k+1}$. If the $(j+1)$-th query 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 after k queries, the difference between l_j and u_j is non-zero, and thus there are more than one possible languages consistent with the answers given. Thus the learner cannot $\mathbf{NPMemQ^kTxtFIN}$-identify $\mathcal{L}$.

(d) Class $\mathcal{L}$ used in proof of Theorem 13 also belongs to $\mathbf{BNPMemQ}^1\mathbf{TxtFIN}$ (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 input language is A_i, L_i or X_i). ∎

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

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

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

Clearly, $\mathcal{L} \in \mathbf{TxtFIN}$, and $\mathcal{L}$ can be learned using finitely many queries of **MemQ** or **SubQ** type. However, it cannot be learned using at most k-queries of any type, when text is not provided to the learner. ∎

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 $Q2$, but needs a high number of queries, whereas if we had used query type $Q1$, then a small number of queries suffice.

Ideally, for diagonalizations and complexity speedup, we would like theorems of the following types:

(a) $Q1^1\mathbf{FIN} \cap Q2\mathbf{FIN}$ diagonalizes against $\bigcup_{k\in N}(\mathbf{BNP}Q2^k\mathbf{TxtFIN} \cup \mathbf{NP}Q2^k\mathbf{TxtFIN})$.

(b) $Q1^1\mathbf{FIN} \cap Q2^{k+1}\mathbf{FIN}$ diagonalizes against $\mathbf{BNP}Q2^k\mathbf{TxtFIN} \cup \mathbf{NP}Q2^k\mathbf{TxtFIN}$.

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 Q2 being **MemQ**, and Q1 being **SubQ**).

Note also that the results of type (c): $Q1^1\mathbf{FIN}$ diagonalizes against $\mathbf{BNP}Q2\mathbf{TxtFIN} \cup \mathbf{NP}Q2\mathbf{TxtFIN}$ — have been obtained in the previous section, where possible: see Theorems 20, 21, and 22.

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 $k \in N$.*

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

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

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, $\mathcal{L} \in \mathbf{TxtFIN} \cap \mathbf{MemQ}^{k+1}\mathbf{FIN}$. Also, it is easy to verify that $\mathcal{L} \notin \mathbf{NPMemQ}^{k}\mathbf{FIN} \cup \mathbf{BNPMemQ}^{k}\mathbf{FIN})$, 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 answer.

(b) Class $\mathcal{L}$ used in (a) above works for learning in $\mathbf{SubQ}^{k+1}\mathbf{FIN}$, and diagonalization against $\mathbf{NPLSubQ}^{k}\mathbf{FIN} \cup \mathbf{BNPLSubQ}^{k}\mathbf{FIN}$ too. ∎

7.2 $Q1 = \mathbf{LSubQ}$ *and* $Q2 = \mathbf{SubQ}$

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

Theorem 29 *(a)* $\mathbf{LSubQ}^{1}\mathbf{FIN} \cap \mathbf{SubQ}^{k+1}\mathbf{FIN} - \mathbf{NPSubQ}^{k}\mathbf{TxtFIN} \cup \mathbf{BNPSubQ}^{k}\mathbf{TxtFIN}) \neq \emptyset$.

(b) $\mathbf{LSubQ}^{1}\mathbf{FIN} \cap \mathbf{SubQFIN} - \bigcup_{k \in N} \mathbf{NPSubQ}^{k}\mathbf{TxtFIN} \cup \mathbf{BNPSubQ}^{k}\mathbf{TxtFIN}) \neq \emptyset$.

Proof. (a) Let $x_i = 2i+1$. Let $A = \{x_i \mid i \in N\}$.

Let $INITA_i = \{x \mid x \notin A\} \cup \{x_j \mid j \leq i\}$.

Let $\mathcal{L} = \{N\} \cup \{INITA_i \mid i < 2^{k+1} - 1\}$.

Clearly, $\mathcal{L} \in \mathbf{LSubQ}^{1}\mathbf{FIN}$ (by querying N; if a subset, then the input language must be N; otherwise if the least counterexample is x_i, then the input language is $INITA_{i-1}$).

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

$\mathcal{L}$ cannot be learned using **NPSubQ**^{k}**TxtFIN** or **BNPSubQ**^{k}**TxtFIN**, as we could maintain that each query leaves at least half the previous possibilities of the $INITA_i$ as candidates. Nearest positive example (if needed) is from $N - A$.

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

7.3 $Q1 =$ **MemQ** *and* $Q2($**L**$)$**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). If, in addition to membership queries, a learner gets more feedback, or has access to a text of the input language, then just one such query can sometimes show speedup advantages over subset queries.

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

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

(**MemQ**^{r}**FIN** $\cap$ **SubQ**$^{k+1}$**FIN**) $-$ (**NPLSubQ**^{k}**TxtFIN** $\cup$ **BNPLSubQ**^{k}**TxtFIN**) $\neq \emptyset$.

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

(c) **NPMemQ**^{1}**FIN** $\cap$ **SubQ**$^{k+1}$**FIN**) $-$ (**NPLSubQ**^{k}**TxtFIN** $\cup$ **BNPLSubQ**^{k}**TxtFIN**) $\neq \emptyset$.

(d) **MemQ**^{1}**TxtFIN** $\cap$ **SubQ**$^{k+1}$**FIN** $-$ (**NPLSubQ**^{k}**TxtFIN** $\cup$ **BNPLSubQ**^{k}**TxtFIN**) $\neq \emptyset$.

Proof. (a) Consider $k + 2$ languages $L_0, L_1, ..., L_{k+1}$, 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 $L_0 = \{x \mid x < y_r, x \notin X \cup Y\}$.

Let $L_i = L_0 \cup \{x_i\} \cup \{y_j \mid b_j = 1$, where $b_1 \ldots b_r$ is the binary representation of $i\}$.

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

It is easy to verify that $\mathcal{L} \in \mathbf{MemQ}^r\mathbf{FIN}$ (a learner can ask questions about $y_1 \ldots y_r$ to determine the input language). $\mathcal{L} \in \mathbf{SubQ}^{k+1}\mathbf{FIN}$ 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 input 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 converge to a grammar for L_0. Let i be such that the learner does not ask query for L_i, $i > 0$. Then, the input text can be extended to a text for L_i, and the answers given are consistent with the input language being L_i. Thus, the learner fails to $\mathbf{NPLSubQ}^k\mathbf{TxtFIN}$-identify ($\mathbf{BNPLSubQ}^k\mathbf{TxtFIN}$-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 $1 \leq i \leq k+1$.

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

(c) The class considered in part(b) belongs to $\mathbf{NPMemQ}^1\mathbf{FIN}$ 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 $\mathcal{L} = \{L\} \cup \{L_i \mid i \leq k\}$.

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

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

$\mathcal{L} \notin$ **NPLSubQ^{k}TxtFIN** $\cup$ **BNPLSubQ^{k}TxtFIN**), as the learner must conjecture on an input text being for L, and answers being given based on the input language being L. If L_i is not queried before the conjecture is made (there exists such an L_i, where $0 < i \leq k+1$), then the answers are consistent with input being L_i, and the input text can be extended to a text for L_i. ∎

Note that (a) above cannot be improved due to Theorem 21 (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)* **MemQ^{1}TxtFIN** $\cap$ **SubQFIN** $-$ $\bigcup_{k\in N}$(**NPLSubQ^{k}TxtFIN** $\cup$ **BNPLSubQ^{k}TxtFIN**) $\neq \emptyset$.

(b) **NPMemQ^{1}FIN** $\cap$ **SubQFIN** $-$ $\bigcup_{k\in N}$(**NPSubQ^{k}TxtFIN** $\cup$ **BNPLSubQ^{k}TxtFIN**) $\neq \emptyset$.

(c) **NPMemQ^{2}FIN** $\cap$ **SubQFIN** $-$ $\bigcup_{k\in N}$(**NPLSubQ^{k}TxtFIN** $\cup$ **BNPLSubQ^{k}TxtFIN**) $\neq \emptyset$.

(d) **MemQFIN** $\cap$ **SubQFIN** $-$ $\bigcup_{k\in N}$(**NPLSubQ^{k}TxtFIN** $\cup$ **BNPLSubQ^{k}TxtFIN**) $\neq \emptyset$.

Proof. (a): P is a partial recursive function defined later.

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 $\mathcal{L} \in$ **MemQ^{1}TxtFIN**. First, wait for either 0 or $F(i)+1$ or $F(i)+100j+2$ appear in the text, for some $1 \leq j \leq i$. If 0 appears

in the input text, then L is the input language; if $F(i)+1$ appears in the input text, then query $F(i)+2$: if the answer is 'no', then the input language is L_i, otherwise the input 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 input language is X_i, otherwise the input language is $A_{i,j}$.

For seeing that $\mathcal{L}$ is in **SubQFIN**, note that one can first query L: if it is a subset then input 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 input language, then the input 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 input language, then if some $A_{i,j}$ is also a subset, $1 \leq j \leq i$, then input language is X_i; otherwise the input language is L_i.

To see that k queries of **LSubQTxtFIN** learner (with the (bounded) nearest positive examples) are not enough, consider a learner $\mathbf{M}_i$. Assume, without loss of generality, that $i > k$. The text provided to $\mathbf{M}_i$ is 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 query L or query L_j/X_j, where $j \neq i$, or query $A_{j,r}$'s, with the nearest positive example given based on the input language being L_i. Answer 'yes' to subset query for L_i. If X_i is queried, then $P(i)\uparrow$. Now if $k+1$ queries are made, then we are done. Otherwise, if M_i outputs a conjecture, then there exists an 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 $\mathbf{M}_i$ fails to identify at least one of L_i or X_i.

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

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

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

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

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

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

(d) Note that INIT $\in$ **MemQFIN**. Thus, part (d) follows from proof of Theorem 24 (a). ∎

Note that parts (b) and (c) above cannot be improved as Theorem 21 (d) showed $\mathbf{NPMemQ^1FIN} \subseteq \bigcup_{r\in N} \mathbf{NPLSubQ^rFIN}$.

7.4 $Q1 = (\mathbf{L})\mathbf{SubQ}$ *and* $Q2 = \mathbf{MemQ}$

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 $\mathbf{SubQ^1FIN} \cap \mathbf{MemQFIN} - \bigcup_{k\in N}(\mathbf{NPMemQ^kTxtFIN} \cup \mathbf{BNPMemQ^kTxtFIN}) \neq \emptyset$.

Proof. Let $L = \{2x \mid x \in N\}$. Let $L_i = L \cup \{1\} - \{2i\}$. Then, $\mathcal{L} \in \mathbf{SubQ^1FIN} \cap \mathbf{MemQFIN}$ (for the **SubQ** learner, answer to the query for L gives away the language; For **MemQ** learner, query of 1 determines whether the language is L or one of L_i; in the latter case, one can sequentially query the languages L_i to determine the unique i such that $L_i \subseteq$ input language — which must be the input language itself).

$\mathcal{L} \notin \mathbf{NPMemQ^kTxtFIN} \cup \mathbf{BNPMemQ^kTxtFIN}$. To see this, suppose answers to queries are 'yes', for all questions in $L \cup \{1\}$. Then, if there exists an initial segment σ (of a text for $L_i \cup \{1\}$) after reading which the learner makes a conjecture, then the input and the answers are consistent with any L_i, where $2i >$ max(content(σ)) and $2i >$ queries made. Thus, the learner cannot identify all members of $\mathcal{L}$. ∎

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

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

does not hold. 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 $k \in N$,*

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

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

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

Proof. (a) $\mathcal{L}$ given in Theorem 29(a) is in $\mathbf{MemQ}^{k+1}\mathbf{FIN}$ (by finding the largest i such that x_i belongs to the input language, in a binary search manner). $\mathcal{L}$ can be shown not to be in $(\mathbf{NPMemQ}^k\mathbf{TxtFIN} \cup \mathbf{BNPMemQ}^k\mathbf{TxtFIN})$, using essentially 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 $j = (i \mod (k+1))$.

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

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

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

On the other hand, $\mathcal{L}$ is not in $\mathbf{NPMemQ}^k\mathbf{TxtFIN}$ or $\mathbf{BNPMemQ}^k\mathbf{TxtFIN}$. To see this, note that one can give the input text for L to a learner, and answer all queries based on the input language being L. If the learner asks only k queries, then there is a $j < k+1$, such that the learner has not queried $4j+3$, and there exists an i such that $j = (i \mod (k+1))$ and the learner has neither queried $4(i+k+1)+3$ or $4i+1$, nor has received data about $4(i+k+1)+3, 4i+1$ in the input when it makes its conjecture for L. But

then the learner cannot distinguish between input 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.

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

Let $\mathcal{L} = \{N\} \cup \{L_i \mid i \leq k\}$. Then, $\mathcal{L} \in \mathbf{SubQ}^1\mathbf{FIN} \cap \mathbf{MemQ}^{k+1}\mathbf{FIN}$ (**SubQ** learner can query N to determine the language. **MemQ** learner can ask queries about $2i+1$, for $i \leq k$, to determine the language).

However, $\mathcal{L} \notin (\mathbf{NPMemQ}^k\mathbf{FIN} \cup \mathbf{BNPMemQ}^k\mathbf{FIN})$. 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 $(\mathbf{NP}/\mathbf{BNP})\mathbf{SubQ}^r(\mathbf{Txt})\mathbf{FIN} \cap (\mathbf{NP}/\mathbf{BNP})\mathbf{MemQ}^{k+1}\mathbf{FIN} - (\mathbf{NP}/\mathbf{BNP})\mathbf{MemQ}^k(\mathbf{Txt})\mathbf{FIN}$ have not been answered optimally. The following gives some partial simulation results, which show why this is not easy.

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

A number of further results employs the following remark.

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

Consider $L \in \mathcal{L}$ *which contains* $\min(A)$*: the answers to* **NPMemQ** *query for such languages have bounded possible nearest positive examples (as any query for* z *has the nearest possible no further than* $\min(A)$*); thus the number of such languages is finite as the class* $\mathcal{L}$ *is in* $\mathbf{NPMemQ}^m\mathbf{FIN}$.

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

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* $k, m \in N$.

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

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

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

PROOF. Using Proposition 17 and Remark 34, we have that if $\mathcal{L} \in \mathbf{MemQ}^m\mathbf{FIN}$ or $\mathcal{L} \in \mathbf{BNPMemQ}^m\mathbf{FIN}$ or $\mathcal{L} \in \mathbf{SubQ}^k\mathbf{FIN} \cap \mathbf{NPMemQ}^m\mathbf{FIN}$, 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 Complexity Speedup Results for Nearest Positive Examples

This section addresses problems similar to those discussed in the previous section for the cases when one considers separation of nearest positive examples versus bounded nearest positive examples, and vice versa, rather than based on number of queries.

8.1 $Q1 = \mathbf{LSubQ}$ *and* $Q2 = \mathbf{SubQ}$

Theorem 36 $\mathbf{LSubQ}^1\mathbf{FIN} \cap \mathbf{NPSubQ}^1\mathbf{FIN} - \mathbf{BNPSubQTxtFIN} \neq \emptyset$.

PROOF.

Let $L = \{100i + 2, 100i + 5 \mid i \in N\}$.

$L_i = \{100i + 3\} \cup \{100j + 2, 100j + 5 \mid j \neq i\}$.

$X_i = L_i \cup \{100i + 2, 100i + 6\}$.

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

$\mathcal{L}$ is in $\mathbf{LSubQ}^1\mathbf{FIN}$, as the least negative counterexample to L being $100i+2$ or $100i+5$, gives away the language (if L is subset of the input language, then the input language must be L).

$\mathcal{L}$ is also in $\mathbf{NPSubQ}^1\mathbf{FIN}$ as either L is a subset of the input language (in which case the input language must be L), or counterexample is $100i + 2$ (in which case the input language is L_i), or counterexample is $100i + 5$, in which case the nearest positive example being $100i + 6$ or $100i + 3$ gives away the language.

On the other hand, for **BNPSubQTxtFIN** learner, counterexample to any language not being L_i/X_i could be $100i+5$, and the nearest bounded positive example would be $100i+3$. Thus, no information about L_i/X_i differentiation is revealed on text for L_i, unless a query about X_i is made. Thus, using Remark 4, we have that $\mathcal{L} \notin$ **BNPSubQTxtFIN**. ∎

Theorem 37 $\mathbf{LSubQ^1FIN} \cap \mathbf{BNPSubQ^1FIN} - \mathbf{NPSubQTxtFIN} \neq \emptyset$.

Proof. Let $L = \{100i+2, 100i+4, 100i+5 \mid i \in N\}$.

$L_i = \{100i+5\} \cup \{100j+2, 100j+4, 100j+5 \mid j \neq i\}$.

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

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

$\mathcal{L}$ is in $\mathbf{LSubQ^1FIN}$, as the least negative counterexample to L being $100i+2$ or $100i+4$, gives away the language (if L is subset of input language, then the input language must be L).

$\mathcal{L}$ is also in $\mathbf{BNPSubQ^1FIN}$ as either L is a subset of the input language (in which case the input language must be L), or counterexample is $100i+2$ (in which case the input language is L_i), or counterexample is $100i+4$, in which case the nearest bounded positive example being $100i+2$ or not gives away the language.

On the other hand, for **NPSubQFIN** learner, counterexample to any language not being L_i/X_i could be $100i+4$, and the nearest positive example would be $100i+5$. Thus, no information about L_i/X_i differentiation is revealed on text for L_i, unless a query about X_i is made. Thus, using Remark 4, we have that $\mathcal{L} \notin$ **BNPSubQTxtFIN**. ∎

8.2 $Q1 =$ **MemQ** *and* $Q2 =$ **SubQ**

Theorem 38 $\mathbf{MemQ^1TxtFIN} \cap \mathbf{NPSubQ^1FIN} - \mathbf{BNPLSubQTxtFIN} \neq \emptyset$.

Proof. Let $L = \{0\} \cup \{100i+1 \mid i \in N\}$.

Let $L_i = \{100i+2\} \cup \{100j+1 \mid j > i\}$.

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

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

$\mathcal{L} \in \mathbf{MemQ^1TxtFIN}$ as one can search for 0 or $100i+2$ in the input. If 0 appears in the input, then input language must be L. If $100i+2$ appears in the input, then one can query $100i+1$ to determine whether the input language is L_i or X_i.

$\mathcal{L} \in \mathbf{NPSubQ^1FIN}$ as on the query for L, either the answer is 'yes' (in which case the input is L), or there is a negative counterexample 0 or of the form $100j+1$. In the latter case, the nearest positive example being $100i+1$ or $100i+2$, gives away the input language.

For $\mathcal{L} \notin \mathbf{BNPLSubQTxtFIN}$, note that if the input language is L_i or X_i, then on input text for L_i, all queries, except for X_i, have the same answer. Thus, we can use Remark 4 to show that $\mathcal{L} \notin \mathbf{BNPLSubQTxtFIN}$. ∎

Theorem 39 $\mathbf{MemQ^2TxtFIN} \cap \mathbf{BNPSubQ^1FIN} - \mathbf{NPLSubQTxtFIN} \neq \emptyset$.

Proof. Let $L = \{100i+2, 100i+3, 100i+5, 100i+6 \mid i \in N\}$.

Let $L_i = \{8, 100i+7\} \cup L - \{100i+2, 100i+6\}$.

Let $X_i = L_i \cup \{100i, 100i+6\}$.

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

$\mathcal{L} \in \mathbf{MemQ^2TxtFIN}$ as one can query 8 to check if the input language is L or one of the L_i/X_i, for some i. If 8 is in the input language, then one can search for $100i+7$, for some i being in the input, and then query $100i+6$.

$\mathcal{L} \in \mathbf{BNPSubQ^1FIN}$, as one can query L. If the answer is 'yes', then the input language must be L. Otherwise, counterexample is either $100i+2$ or $100i+6$. In the latter case, the input language is L_i. In the first case, the bounded nearest positive example gives away the language.

For $\mathcal{L} \notin \mathbf{NPLSubQTxtFIN}$, note that if the input language is L_i or X_i, then on input text for L_i, all queries, except for X_i, have the same answer. Thus, we can use Remark 4 to show that $\mathcal{L} \notin \mathbf{NPLSubQTxtFIN}$. ∎

Open 40 $\mathbf{MemQ^1TxtFIN} \cap \mathbf{BNPSubQ^1FIN} - \mathbf{NPLSubQTxtFIN} \neq \emptyset$*?*

Theorem 41 $\mathbf{MemQFIN} \cap \mathbf{NPSubQ^1FIN} - \mathbf{BNPLSubQTxtFIN} \neq \emptyset$.

Proof. The following modification of Theorem 5 proof will show the theorem.

Let $A = \{4\} \cup \{100i+2, 100i+3 \mid i \in N\}$.

Let $L = \{100i + x \mid i \in N, x \in \{0, 6\}\}$

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

Let $X_i = L_i \cup \{100i + 3\}$.

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

$\mathcal{L} \in$ **MemQFIN**, as one can query $3, 4$, and based on their membership decide whether the input language is A or L or one of L_i/X_i, $i \in N$. In the latter case, one can query $100i$, $100i + 3$ for various i to eventually determine the input language.

$\mathcal{L} \in$ **NPSubQ^{1}FIN**, as one can query L. If counterexample is not there, then the input language must be L. If counterexample is of form $100j + 6$, then the input language must be A. Otherwise, the counterexample would be $100i$ for some i, and the nearest positive example will let us know whether the input language is A or L_i or X_i.

However, $\mathcal{L} \notin$ **BNPLSubQTxtFIN**, as one can give input text for L_i, and all queries except X_i have same answers for the input language being L_i or X_i. Thus, we can use Remark 4 to show that $\mathcal{L} \notin$ **BNPLSubQTxtFIN**. ∎

Theorem 42 **MemQFIN** $\cap$ **BNPSubQ^{1}FIN** $-$ **NPLSubQTxtFIN** $\neq \emptyset$.

Proof. Let $A = \{4\} \cup \{100i, 100i + 1 \mid i \in N\}$.

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

Let $L_i = \{3\} \cup L - \{100i + 2\}$.

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

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

$\mathcal{L} \in$ **MemQFIN** as one can query $3, 4$, and based on their membership decide whether the input language is A or L or one of L_i/X_i, for $i \in N$. In the latter case, one can query $100i$, $100i + 2$ for various i to eventually determine the input language.

$\mathcal{L} \in$ **BNPSubQ^{1}FIN**, as one can query L. If counterexample is not there, then the input language must be L. If counterexample is of form $100i + 3$, then the input language must be A. Otherwise, the counterexample would be $100i+2$, for some i, and the bounded nearest positive example will let us know whether the input language is A or L_i or X_i.

However, $\mathcal{L} \notin$ **NPLSubQTxtFIN**, as one can give input text for L_i, and all queries except X_i have same answers for the input language being L_i or X_i. Thus, we can use Remark 4 to show that $\mathcal{L} \notin$ **NPLSubQTxtFIN**. ∎

Theorem 43 $\mathbf{NPMemQ^1FIN} \cap \mathbf{NPSubQ^1FIN} - \mathbf{BNPLSubQTxtFIN} \neq \emptyset$.

Proof. Let $L = \{0\} \cup \{100i + 3 \mid i \in N\}$

Let $L_i = \{100j + 3 \mid j \geq i\}$.

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

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

Then $\mathcal{L} \in$ **NPMemQ^{1}FIN** ($\mathcal{L} \in$ **NPSubQ^{1}FIN**), as if 0 is in the input language, then the input language must be L; otherwise, the nearest positive example to 0 will be $100i + 2$ or $100i + 3$, for some i, which would give away the input language as X_i or L_i.

$\mathcal{L} \notin$ **BNPLSubQTxtFIN** as all queries except for X_i would give the same answer for input language being L_i or X_i. Thus, we can use Remark 4 to show that $\mathcal{L} \notin$ **BNPLSubQTxtFIN**. ∎

The above results leave many questions open, such as:

Open 44 $\mathbf{NPMemQ^1FIN} \cap \mathbf{BNPSubQ^1FIN} - \mathbf{NPLSubQTxtFIN} \neq \emptyset$*?*

Theorem 45 *Suppose* $k < 2^r - 1$.

(a) $\mathbf{MemQ^rFIN} \cap \mathbf{BNPSubQ^1FIN} - \mathbf{NPLSubQ^kTxtFIN} \neq \emptyset$.

(b) $\mathbf{MemQ^rFIN} \cap \mathbf{NPSubQ^1FIN} - \mathbf{BNPLSubQ^kTxtFIN} \neq \emptyset$.

Proof. (a) Consider $L_0 = \{2j \mid j \leq k\} \cup \{7k + 7k * w \mid 0 \leq w \leq r\}$.

For $0 \leq i < k$, $L_{i+1} = L_0 \cup \{2i + 1\} \cup \{7k + 7k * w - i - 3 \mid 0 \leq w \leq r\} \cup \{7k + 7k * w - 1 \mid 1 \leq w \leq r$ and $b_w = 1$, for $i + 1 = b_1 b_2 \ldots b_r$ in binary$\}$.

$L_{k+1} = L_0 \cup \{7k - 1\} \cup \{7k + 7k * w - 1 \mid 1 \leq w \leq r$ and $b_w = 1$, for $k + 1 = b_1 b_2 \ldots b_r$ in binary$\}$.

(Note that $2^r > k + 1$, so above coding in binary is possible).

$\mathcal{L} = \{L_i \mid 0 \leq i \leq k + 1\}$.

$\mathcal{L} \in \mathbf{MemQ}^r\mathbf{FIN}$, since we can ask queries for $7k + 7k * w - 1$, for $1 \leq w \leq r$ to get $b_1 b_2 \ldots b_r = i$, where L_i is the input language.

$\mathcal{L} \in \mathbf{BNPSubQ}^1\mathbf{FIN}$, as one can query L_{k+1}, and either it is a subset of the input language (in which case the input language is L_{k+1}) or any counterexample of form $7k + 7k * w - 1$, where $0 \leq w \leq r$, will give the bounded nearest positive example as some number of form $7k + 7k * w - i - 3$, for some $0 \leq i < k$ (in which case input language is L_{i+1}) or $2k$ or $7k(w-1)$ for some w, (in which case input language is L_0).

$\mathcal{L} \notin \mathbf{NPLSubQTxtFIN}$, as input text for L_0 along with answering negatively all queries of form L_{i+1}, for $0 \leq i < k$ (with the counterexample being $2i + 1$, and the nearest positive example being $2i$); for query L_{k+1}, counterexample would be $7k - 1$, with the nearest positive example being $7k$. Then each query spoils at most one of L_{i+1}, $0 \leq i \leq k$, and thus, one needs at least $k + 1$ queries to determine that the input language is L_0 (otherwise there is at least one more possible language consistent with the input).

(b) Similar to part (a), but this time we use slightly different languages to make the class in $\mathbf{NPSubQ}^1\mathbf{FIN}$ but not in $\mathbf{BNPLSubQFIN}$.

L_0 as in part (a).

For $0 \leq i < k$, $L_{i+1} = L_0 \cup \{2i + 1\} \cup \{7k + 7k * w + 3k + i + 3 \mid 0 \leq w \leq r\} \cup \{7k + 7k * w + 3k + 1 \mid 1 \leq w \leq r \text{ and } b_w = 1, \text{ for } i + 1 = b_1 b_2 \ldots b_r \text{ in binary}\}$.

$L_{k+1} = L_0 \cup \{7k + 3k + 1\} \cup \{7k + 7k * w + 3k + 1 \mid 1 \leq w \leq r \text{ and } b_w = 1, \text{ for } k + 1 = b_1 b_2 \ldots b_r \text{ in binary}\}$.

$\mathcal{L} = \{L_i \mid 0 \leq i \leq k + 1\}$.

Remainder of the proof is similar to part (a). ∎

Theorem 46 *For all $k \in N$,*

(a) $\mathbf{BNPMemQ}^1\mathbf{FIN} \cap \mathbf{BNPSubQ}^1\mathbf{FIN} - \mathbf{NPLSubQ}^k\mathbf{TxtFIN} \neq \emptyset$.

(b) $\mathbf{BNPMemQ}^1\mathbf{FIN} \cap \mathbf{NPSubQ}^1\mathbf{FIN} - \mathbf{BNPLSubQ}^k\mathbf{TxtFIN} \neq \emptyset$.

Proof. We can use the classes in Theorem 45 for the respective parts. The **MemQ** query would be for $7k - 1$ (for part (a)). For part (b), we add an additional element $12k$ for L_{k+1}, and membership query will be for $12k$. This would give the class $\mathcal{L}$ to be in $\mathbf{BNPMemQ}^1\mathbf{FIN}$. ∎

Theorem 47 $\mathbf{NPSubQ^1FIN} \cap \mathbf{NPMemQ^1FIN} - \mathbf{BNPMemQTxtFIN} \neq \emptyset$.

PROOF. The class $\mathcal{L}$ of Theorem 12 also belongs to $\mathbf{NPSubQ^1FIN} \cap \mathbf{NPMemQ^1FIN}$. ∎

Open 48 *(a)* $\mathbf{SubQFIN} \cap \mathbf{NPMemQ^1FIN} - \mathbf{BNPMemQTxtFIN} \neq \emptyset$.

(b) $\mathbf{SubQFIN} \cap \mathbf{BNPMemQ^1FIN} - \mathbf{NPMemQTxtFIN} \neq \emptyset$.

Open 49 *(a)* $\mathbf{SubQ^1FIN} \cap \mathbf{NPMemQFIN} - \mathbf{BNPMemQTxtFIN} \neq \emptyset$.

(b) $\mathbf{SubQ^1FIN} \cap \mathbf{BNPMemQFIN} - \mathbf{NPMemQTxtFIN} \neq \emptyset$.

Open 50 *(a)* $\mathbf{NPSubQ^1FIN} \cap \mathbf{NPMemQ^1FIN} - \mathbf{BNPMemQTxtFIN} \neq \emptyset$.

(b) $\mathbf{BNPSubQ^1FIN} \cap \mathbf{NPMemQ^1FIN} - \mathbf{BNPMemQTxtFIN} \neq \emptyset$.

9 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* 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 number of queries is used as 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 do 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. Some questions — mostly related to complexity advantages of models using the nearest or bounded nearest positive examples — have been left open.

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. It would be interesting also to define and explore formalizations of one-shot learnability via queries, where positive feedback were semantically close to the negative datum, rather than being close based on coding. Such models may also be interesting in the context of learning some important specific indexed classes, for example, patterns, finite automata, or regular expressions.

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.

[Blu67] M. Blum. A machine-independent theory of the complexity of recursive functions. *Journal of the ACM*, 14:322–336, 1967.

[Gol67] E. M. Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.

[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: Seventeenth International Conference (ALT' 2006)*, volume 4264 of *Lecture Notes in Artificial Intelligence*, pages 154–168. Springer-Verlag, 2006.

[JK07] S. Jain and E. Kinber. Learning languages from positive data and negative counterexamples. *Journal of Computer and System Sciences*, 2007. To appear.

[JLZ05] S. Jain, S. Lange, and S. Zilles. Gold-style and query learning under various constraints on the target class. In Sanjay Jain, Hans Ulrich Simon, and Etsuji Tomita, editors, *Algorithmic Learning Theory: Sixteenth International Conference (ALT' 2005)*, volume 3734 of *Lecture Notes in Artificial Intelligence*, pages 226–240. Springer-Verlag, 2005.

[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.

[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.

[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.