Towards a Linear-Data Monotone Wrapper Algorithm for Machine Learning Algorithms

07-06-2023
Berend Kam - 4567242
EEMC- MSc Computer Science
Pattern Recognition & Bioinformatics

Preface

Machine learning algorithms can behave in surprising ways, and one of such surprises is non-monotonicity. Non-monotone machine learning algorithms may become worse given more training data, the opposite of what a scientist might expect when using an algorithm that is supposed to learn from data. This research tries to get a step closer to removing non-monotonicity from machine learning algorithms in a data efficient way, reshaping crooked learning curves like a smith hammering away at a bar of metal. The research was carried out at TU Delft under close supervision of Marco Loog and Tom Viering, who I wholeheartedly thank for guiding me during this process. Additionally, special thanks goes to the members of the Thesis Committee: Jan van Gemert and Sicco Verwer.
Berend Kam
07-06-2023

Towards a Linear-Data Monotone Wrapper Algorithm for Machine Learning Algorithms

Berend Kam, Thesis Supervisor: Jan van Gemert, Daily Supervisor: Marco Loog, Daily Co-Supervisor: Tom Viering
TU Delft

Machine learning algorithms (learners) are typically expected to produce monotone learning curves, meaning that their performance improves as the size of the training dataset increases. However, it is important to note that this behavior is not universally observed. Recently monotonicity of learning curves has gained renewed attention, as several authors have proposed 'wrapper' algorithms; algorithms that attempt at filtering the hypotheses produced by a learner to turn them into a monotone learner, even if the learner itself is not monotone. Such wrappers use part of the training data as validation data, and each newly produced hypothesis is evaluated using this validation data. However, with each new hypothesis, the validation data grows in size exponentially. As such the wrapper is data-hungry, using up to 85% of the training data as validation data in some cases. This paper investigates what happens when a linearly growing validation sample is used instead. Is it enough to retain monotonicity? We proof that selecting the best performing hypothesis from a finite set of hypotheses, based on a validation sample that grows linearly, results in a monotone learning curve. However, when introducing a new hypothesis with each increase in the validation sample size, it has been observed that this selection process does not demonstrate monotonic behavior. The authors of this paper hope that this work provides key insight into how to choose from a set of hypotheses in a monotone way, and that the work may be a stepping stone for a fully functioning linear-data monotone wrapper algorithm.

I. Introduction

Contrary to popular belief, the ability of a machine learning algorithm (learner) to generalise can become worse with larger training samples [1–6]. Such machine learners are non-monotone, meaning the expected true error of the learner may increase with each additional instance of training data[7]. Such learners thus also have a non-monotone learning curve, the curve of the generalisation error of a learner vs the amount of training data it has trained on.

Unfortunately non-monotonicity is an undesirable trait of learning curves; in general, learners are expected to become better with more training data, or at least not worse [2]. Additionally non-monotonicity makes certain learning curves analysis tasks much more difficult, such as model selection or extrapolation of learning curves to larger training sizes [2]. Ideally then, all learners are monotone learners, but is it even possible to turn a non-monotone learner into a monotone learner? And can this be done without altering the search space of the learner? Ideally, one would create a 'wrapper' algorithm that has only black-box acces to the learner. The wrapper keeps track of outputted hypotheses of the learner at increasing training sizes, and picks which of the generated hypotheses to return in order to form a monotone learning curve.

Bousquet et al. were recently able to construct the first universally consistent monotone wrapper to our knowledge, using only the aforementioned black-box access to any underlying learner[8]. Universally consistent means that the algorithm is converges to the optimal error in for a given problem, also known as the Bayes error, for any distribution of data D on the input domain $\mathbb{Z}$ in the limit [7]. They do so by splitting the training sample in two samples. One sample is used for training the learner and generating hypotheses, similar to a normal training sample. The other is used as a validation sample, and evaluates the generated hypotheses. Our work focuses mostly on the evaluation process using the validation sample. As we will show in more detail in Section III, one potential problem with Bousquet's method is that in order to guarantee monotonicity for newly generated hypotheses, the validation sample must grow in size exponentially. In more detail, the validation sample used to evaluate the hypotheses $S^{\text{validation}}$, grows according to $|S_{t+1}^{\text{validation}}| = c|S_t^{\text{validation}}|$ where c is some positive constant > 1. This severely limits the resolution of the learning curve as all training sample sizes between S_t and S_{t+1} are skipped, having only few hypotheses at large intervals. We will highlight some cases in which this may lead to unexpected results.

The main focus of this research is hence to work towards a universally consistent monotone wrapping algorithm that can generate hypotheses more frequently. What happens if the validation sample grows by a constant c each time, so linearly? A simple algorithm M is devised to test this. M chooses from a set of hypothesis H_t the best performing hypothesis on a validation sample $S_t^{\text{validation}}$. The validation sample is then increased by $|S_{t+1}^{\text{validation}}| = |S_t^{\text{validation}}| + c$, for some positive constant c. We then evaluate a set of hypotheses H_{t+1} using this new validation set, and again choose a best performing hypothesis. Is the chosen hypothesis based on $S_{t+1}^{\text{validation}}$ expected to have a lower generalisation error than the hypothesis chosen based on $S_t^{\text{validation}}$? In other words, is this algorithm monotone? We ask three subquestions:

1) Given a immutable set of hypotheses $|H_t| = 2$, is M monotone?
2) Given a immutable set of hypotheses $|H_t| = k$, with k some positive integer, is M monotone?
3) Given an expanding set of hypotheses $|H_{t+1}| = |H_t| + 1$, is M monotone?

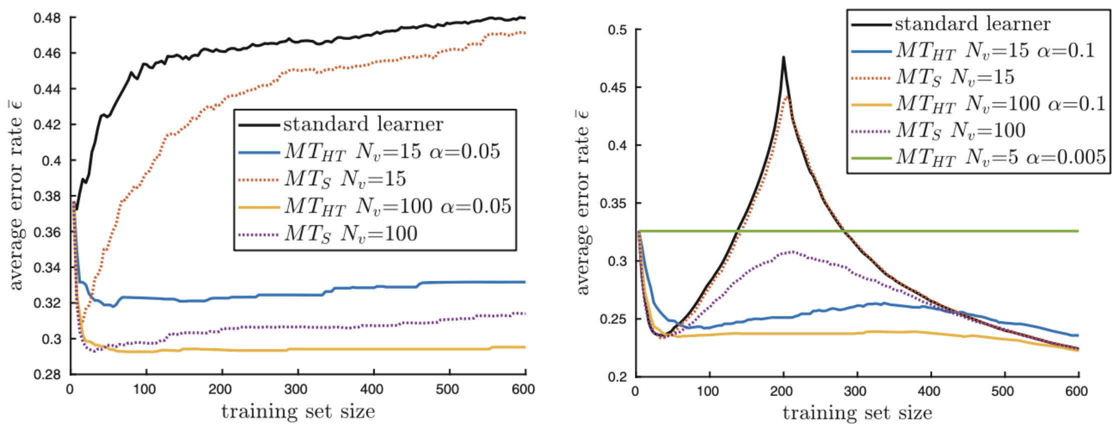
Note that the last item on an expanding set of hypotheses is our final goal, a monotone wrapper algorithm that uses linearly growing validation data.

Section II discusses related works. Section III introduces notation used, and discusses Bousquet's work in more detail, highlighting some areas of improvement. Section IV introduces our new monotone wrapper algorithm, which is proven to be monotone in Section V. All results are discussed in Section VI, and finally a conclusion is given in Section VII.

II. Related Works

In their classical work on machine learning, it was conjectured by Devroye, Györfi, and Lugosi[7] that no universally consistent monotone learner can exist, who referred to such learners as 'smart'. Viering, Mey and Loog[9] continue this line of questioning, and pose the open question to what extent you can expect learners to behave monotonically, and what causes non-monotone behaviour? Several works focus on non-monotone problems, such as the dipping phenomenon [1], or the peaking phenomenon [3]. Shortly summarised, the dipping phenomenon may occur when a learner optimizes a different loss function than it is evaluated with. The peaking phenomenon - which has become synonymous double descent - typically occurs at the point where the number of instances is equal to number of features of the data [2]. Figure 1 shows both the peaking and double descent learning curves.

To our knowledge, the first to proof wrong Devroye, Gyorfi and Lugosi's conjecture was Pestov [4], who proposes a universally consistent monotone binary classifier. Shortly summarised, Pestov splits a data sample S_t into a "labelling" (training) sample and a "testing" (validation) sample. The classifier learns to split the input domain into partitions based on training sample, with the majority of labels of a given class within a partition determining the outputted label. Each round, a new partition is generated. Then, from all previously generated partitions and this new partition the best performing partition is chosen based on the performance on the validation sample. Crudely stated, by growing the validation sample exponentially each round, Pestov guarantees enough instances are present in all partitions to make a monotone decision in expectation.



(a) Dipping Learning Curve **(b) Peaking Learning Curve**

Figure 1. Non-Monotone Learning Curves. Two graphs from "Making Learners (More) Mononotone" [6]. The black learning curves show a typical non-mononte learning curve, whereas the yellow learning curves show a (mostly) monotone learning curve

Bousquet [8] extends Pestov's method to create a universally consistent monotone wrapper algorithm for all learners, effectively answering both Devroye, Gyorfi, and Lugosi, and Viering, Mey, and Loog; all learners may be turned into universally consistent monotone learners. Their method guarantees monotonicity in a similar manner to Pestov. They too split their training data into train and validation samples, generate a new hypothesis each round, and grow the validation sample exponentially each round to guarantee monotonicity. Section III goes into more detail of the inner workings of Bousquet's algorithm.

Other works that create a nearly monotone wrapper algorithm are proposed by Viering [6] and Mhammedi [5]. These learners are non-monotonic with at most probability ϵ.[1] Viering's algorithm uses McNemar's test, which Fagerland explains well [10]. Essentially, the probability of making a non-monotone decision is bounded by the α parameter of McNemar's test. Mhammedi uses a modified version of his earlier work, the FreeGrad algorithm [11] to attain monocity.

In a different perspective, monotone learning curves are part of a broader scope of research that focuses on learning curves. Viering, Mey, and Loog [2] give a detailed survey on learning curves and what information may be extracted from them. They highlight works [12–17] that parameterize learning curves, fitting known functions to their shape at low data sizes. These functions are then extrapolated to larger training sizes to estimate performance. Additionally, they highlight cases in which learning curves are used for model selection or reducing complexity. A distinction is made between 'well-behaved' learning curves, monotone learning curves which are suited for parameterisation, and 'ill-behaved' learning curves, which are non-monotone learning curves.

III. Background

A. Notation and Setting

In order to construct a learning curve, one trains a machine learning algorithm A on a sample of data S_t where t is an index. A sample S_t consists of $|S_t|$ instances, drawn independently and identically distributed (i.i.d.) from underlying data distribution D, $S_t \sim D$. After training, the machine learning algorithm returns a hypothesis $h = A(S_t)$. To measure how well a hypothesis generated by the learner performs for its designed

[1]not to be confused with the notation we use for true error ϵ_i of a hypothesis h_i

task, we evaluate a hypothesis on either the underlying L_D or on a sample L_{S_t}. Throughout this paper, we will assume use of a classifier using the error rate to evaluate hypotheses. The error rate for a sample S_t is simply given by error rate = $\frac{\text{number of instances classified incorrectly}}{\text{number of instances in sample}}$. For a hypothesis h_i, the true error rate ϵ_i, also known as generalisation error, is the error rate of a hypothesis on the underlying D. As D is a distribution and not a sample, the true error rate ϵ_i is the probability of classifying a random instance of D incorrectly, $\epsilon_i = P(L_{S_1 \sim D}(h_i) = 0)$. The purpose of a learner is to minimize the true error rate.

We define a wrapper algorithm M as an algorithm that chooses from a set of hypotheses H a hypothesis h_{best} which it deems best to return. This is not necessarily the hypothesis with the lowest generalisation error, as other constraints may apply. Bousquet [8] and Viering [6] use a wrapper M that takes as input a learner A and a sample S_t and choose a hypothesis h_{best} from a set of generated hypotheses by A, $h_{best} = M(A, S_t)$. Alternatively, we use algorithm M to take in a predetermined set of hypotheses H instead of A to return a hypothesis $h_{best} \in H$, $h_{best} = M(H, S_t)$.

A learning curve is a plotted curve with on the horizontal axis $|S_t|$, and on the vertical axis the expected true error rate on the underlying $\mathbb{E}_{S_t \sim D}\, L_D(A(S_t))$. If no exact error rate on the underlying can be determined, an estimate on an out-of-sample test sample is used.

A monotone wrapper algorithm, is a wrapper algorithm for which the choice of h_{best} at increasing sample sizes forms a monotone learning curve. We define a linear-data monotone wrapper as follows.

Definition 1 (Linear-Data Monotone Wrapper Algorithm) *A wrapper M is monotone if for any two i.i.d. samples $S_t, S_{t+1} \sim D$ with $|S_{t+1}| = |S_t| + c$ for some positive constant integer c, and a learner A, it holds that the expected error on the underlying distribution D does not become larger from t to $t+1$ for any t, so $\mathbb{E}_{S_{t+1} \sim D}\,[L_D(M(A, S_{t+1}))] \leq \mathbb{E}_{S_t \sim D}\,[L_D(M(A, S_t))]$.*

An overview of all notation used can be found in Appendix A

B. Bousquet

Bousquet's monotone wrapper algorithm [8] is the first universally consistent monotone wrapper algorithm to our knowledge. Whilst the exact mechanisms of the wrapper are considered outside the scope of this work, we have identified some opportunities for improvement, and will explain aspects of the wrapper which are related to these improvements.

1. Training/Validation Split

Bousquet splits a training sample S_t into blocks of increasing size $B_1, B_2, ..., B_k, B_{k+1}, Bk+2$, each block 4 times larger than the previous block $|B_i| = 4|B_{i-1}|$, except for the last block $B_{k+2} = Bk+1$. See Figure 2. The best hypothesis is initialised as $h_{best} = h_\emptyset$, with $h_\emptyset$ a randomly initialised hypothesis. Each round, a new hypothesis h_j is trained on a conjunction of all blocks up to i $\cup_{i=\{1..j\}B_i}$, and then evaluated together with h_{best} on the next block B_{j+1}. The best of h_j and h_{best} is then chosen as the new h_{best}. The process of evaluating and updating h_{best} is done through means of the update rule U. After the last update, an additional block is used to regularize the outputted hypothesis, which is not relevant for our work. All in all, the wrapper is quite data-hungry, using 85.7% of S_t as validation data ($\lim_{n\to\infty} \frac{2\cdot 4^n}{(2\cdot 4^n + \sum_{i=0}^{n-1} 4^i)} \approx 0.857$).

2. Update Rule

The update rule uses a validation set to update h_{best}, $h_{best} = U(h_{bestprevious}, h_i, B_{i+1})$. One important feature we want to highlight is that the wrapper algorithm only chooses a new hypothesis as h_{best} if it outperforms a the original hypothesis with a margin of ϵ_n[2] dependent on the number of instances in the

[2]Unrelated to the true error as defined in Subsection III.A

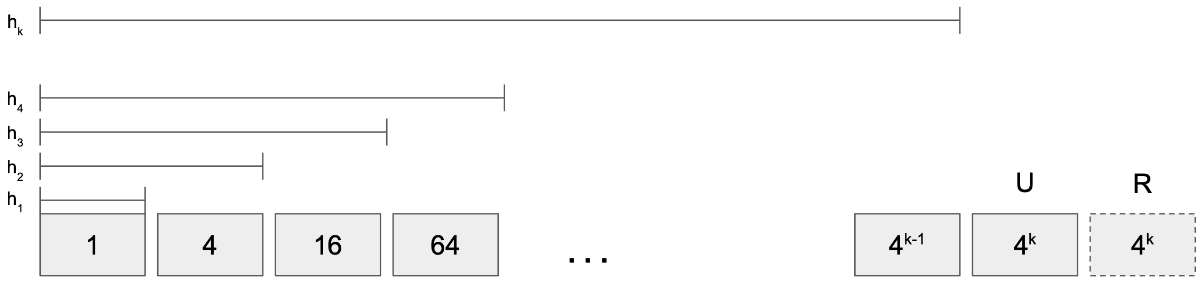
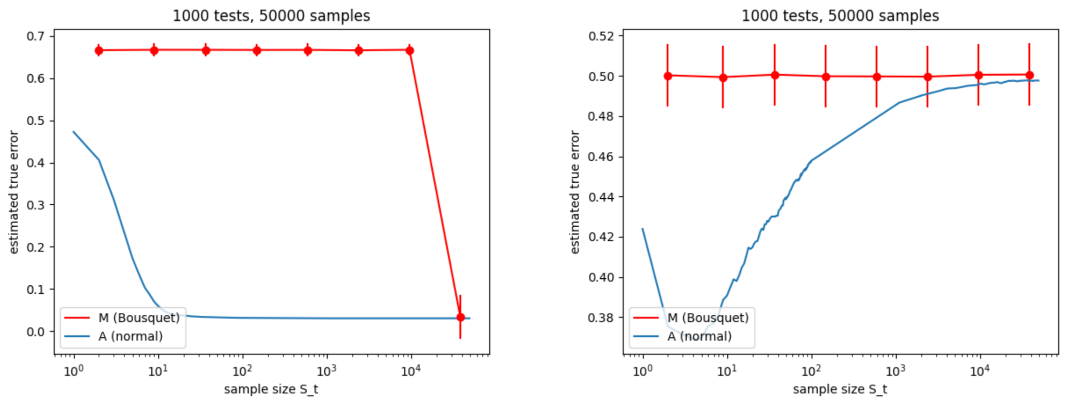


Figure 2. Data sample split. The last two data blocks are used the last update U and a regularisation step R, and contain approximately 85.7% of all data in the sample.



(a) no dipping, 3 classes **(b) dipping, 2 classes**

Figure 3. Learning curves of Bousquet's wrapper. Curves are plotted up to sample size 50k, the estimated true error is the error rate over a test sample of size 1000, the test is repeated 1000 times and averaged. The error bars are 1 standard deviation.

validation sample $|B_i| = n$. For a classification problem with more than two classes, ϵ_n is defined as $\epsilon_n = \sqrt{ln(64n)/n} + \frac{72}{\sqrt{n}}$.

As an example, imagine the first initialised hypothesis $h_\emptyset$ has an error rate on the underlying of $L_D(h_\emptyset) \approx 0.666$. This is plausible in a classification problem with 3 classes who appear equally frequently in D, and when $h_\emptyset$ is set to classify all instances as one class. In order for a new hypothesis h_i to be selected through update on a validation sample B_{i+1}, the ϵ_n term must be less than this number 0.666 when $L_{B_{i+1}}(h_i) = 0$, so $L_{B_{i+1}}(h_i) + \epsilon_n \leq L_{B_{i+1}}(h_\emptyset)$. This first happens when $n = 12891$, so when the size of the validation set B_{i+1} is 12891. Since B_i is only a split of the training sample S_t, the entire sample S_t needs to be at least 38229, $|S_t| \geq 38229$.

3. Shape of the Learning Curve

Learning curves are plotted for a toy 3-class classification problem in Figure 3a, with 1 dimensional input data. The classes appear equally frequent, and are sampled from Guassians $N(0, 0.5), N(2, 0.5), N(4, 0.5)$ respectively. Two learning curves are plotted, one of classifier $A(S_t)$ which simply matches inputs to the mean of each class seen in training data, also known as a nearest mean classifier. The other shows Bousquet's wrapper $M(A, S_t)$. Notable of these learning curves is that whilst $A(S_t)$ converges rather quickly, because of the update rule, $M(A, S_t)$ does not. This is a practical problem in low-data settings.

Figure 3b shows the learning curve of a toy 2-class classification problem, also known as the dipping problem. The classes also appear equally frequent as Gaussians, but one class is drawn from $N(0, 0.5)$ while the other is drawn from $N(-2, 0.5)$, $N(2, 0.5)$, both with equal probability. Again two learning curves are plotted, one of the nearest mean classifier $A(S_t)$, and one of Bousquet's wrapper $M(A, S_t)$. Notable of these learning curves is that $M(A, S_t)$ is unable to converge to the best performing hypothesis generated by $A(S_t)$. The reasons for these are twofold: on the one hand, the update rule prohibits choosing hypotheses generated at small validation sample sizes. On the other hand, exponentially growing validation sample sizes limit the frequency of generated hypotheses. For example, say $A(S_{70})$ is the best performing hypothesis. As $M(A, S_t)$ only generated hypotheses $A(S_{64})$ and $A(S_{256})$, it is unable to output S_{70}.

To improve the shape of the learning curve, it would be ideal to have a slower growing validation sample, such as a linear growing sample $|S_t| = |S_{t-1}| - k$. Additionally, this would cut down on the data hungriness.

IV. Algorithm

We present Algorithm 1 to investigate what happens if a linearly growing validation sample is used instead of an exponentially growing validation sample. The main idea of Algorithm 1 is to take a learner A, and randomly produce an immutable set of hypotheses H in the parameter space of this learner. Contrary to a 'normal' wrapper algorithm, which splits a sample S_t into a training sample S_t^{train} and validation sample $S_t^{\text{validation}}$, the entire training sample S_t is used as a validation sample in our algorithm. To generate a learning curve, we 'train' our learner by selecting a best performing hypothesis $h_{best} \in H$ based on the performance (lowest error rate) on S_t. In the case of a tie/draw for the lowest error rate on S_t between multiple hypotheses, any of the tying hypothesis is returned with random uniform probability. For example for hypotheses h_i and h_j this means that h_i and h_j are returned with probability 0.5 each.

By increasing the size of S_t linearly, so $|S_t| = |S_{t-1}| + c$ for some positive constant c, we show it is possible to construct a monotone learning curve. Randomly initialising H and not training A on any training data may seem a strange choice. However, the aim of this research is investigate the effects of a linearly growing validation sample. As such, we imagine that A is some learner that can only produce some fixed amount of hypotheses H^{fixed}. By setting $H = H^{fixed}$ we show that we can turn this learner into a monotone learner without altering its search space. Additionally, both our technique and a wrapper choose the best performing hypothesis from a set of previously produced hypotheses H. The only difference between the two is that a wrapper algorithm not only considers this set of previously produced hypotheses H but also a new hypothesis that the underlying learner just produced, h_{new}. Proving monotonicity for $H = H^{fixed}$, could be an integral part of proving monotonicity for $H = H + h_{new}$. More on this in Section VI. Also note that if H^{fixed} contains the hypothesis with the optimal error, the wrapper is universally consistent.

To give high level intuition of why this produces a monotone learning curve, imagine that as the size of the 'validation' sample S_t increases, so does the probability of choosing the hypothesis with the lowest error rate, as the validation sample approaches the actual underlying distribution of data D. Even a single additional instance will improve this probability, which makes that M returns, in expectation, an improving hypothesis. In the limit, the probability of choosing the hypothesis with the lowest error rate becomes 1.

V. Theoretical Results

This section contains the main technical contribution of this research. It consists of the proof of two theorems, Theorem 1 and Theorem 2. The proof of Theorem 1 serves a warmup proof for Theorem 2,.

1. Assumptions

Note that all proofs assume a classification problem, with the error rate as both the training error and the validation error function, as defined in Section III. Another, strong assumption is that any two hypotheses

Algorithm 1

```
1: procedure $M(H, S_t) \rightarrow f$
2:     if $t = 0$ then
3:         return uniform randomly picked $h_i \in H$
4:     end if
5:     $H_{best} \leftarrow \emptyset$
6:     $error_{best} \leftarrow \infty$
7:     for $h_i \in H$ do
8:         $error_{current} \leftarrow L_{S_t}(h_i)$
9:         if $error_{current} < error_{best}$ then
10:            $H_{best} \leftarrow \{h_i\}$
11:            $error_{best} \leftarrow error_{current}$
12:        else if $error_{current} = error_{best}$ then
13:            $H_{best} \leftarrow H_{best} : h_i$        ▷ ":" is the concat operation, so $h_{best}$ is a list of the best hypotheses
14:        end if
15:    end for
16:    return uniform randomly picked $h_i \in h_{best}$
17: end procedure
```

$h_i, h_j \in H$ classify independently. This means that the event that a single instance is classified correctly or incorrectly by h_i does not affect the probability that h_j classifies that instance correctly or incorrectly. For example, the given the two hypotheses h_i and h_j with respective true error rates ϵ_i and ϵ_j, the probability that both classify any given instance incorrectly is simply $\epsilon_i \epsilon_j$. Two hypotheses can actually classify independently by using a separate validation sample per hypothesis, but we choose to formulate this as an assumption for ease of notation. It is discussed in more detail in Section V.G.

Theorem 1 (Monotonicity for $|H| = 2$) *For two samples $S_t, S_{t+1} \sim D$ with t some positive integer, and $H = \{h_1, h_2\}$, learning algorithm M as defined in Algorithm 1 is monotone.*

Theorem 2 (Monotonicity for $|H| = k$) *For two samples $S_t, S_{t+1} \sim D$ with t some positive integer, and $H = \{h_1, h_2, ..., h_k\}$, learning algorithm M as defined in Algorithm 1 is monotone, under the assumption all hypotheses $h \in H$ classify independently.*

A. Preliminaries

In order to proof monotonicity of Algorithm 1, we must find that its expected true/generalisation error becomes smaller with more data. So, given M, H, and D, we must find that $E_{S_{t+1} \sim D}(L_D(M(H, S_{t+1}))) \leq E_{S_t \sim D}(L_D(M(H, S_t)))$. We first define M for $|H| = 2$, so $H = \{h_1, h_2\}$. A simple formulation of $M(H, S_t)$ can be given as Equation 1 below, with $ the uniform random selection operation.

$$M(H, S_t) = \begin{cases} h_1, & \text{if } L_{S_t}(h_1) < L_{S_t}(h_2) \\ h_2, & \text{if } L_{S_t}(h_1) > L_{S_t}(h_2) \\ h_i \overset{\$}{\leftarrow} \{h_1, h_2\}, & \text{otherwise} \end{cases} \tag{1}$$

Subsequently, $E_{S_t \sim D}(L_D(M(H, S_t)))$ is defined as below.

$$E_t = E_{S_t \sim D}(L_D(M(H, S_t))) = p^t_{[1]}\epsilon_1 + p^t_{[2]}\epsilon_2 + p^t_{[1,2]}(\frac{1}{2}\epsilon_1 + \frac{1}{2}\epsilon_2) \tag{2}$$

ϵ_i signifies the true error of hypothesis $h_i \in H$ on D. $p_{[1]}$ denotes the probability that h_1 has the lowest error rate on S_t. As a general notation p^t_B denotes the probability that all hypotheses of H with the lowest error rate on a validation sample $S_t \sim D$ are contained within B, with $B \subseteq H$. To formally define p^t_B:

$$p^t_B = P\left(\forall h_i \in B : L_{S_t \sim D}(h_i) = min_{h_j \in B} L_{S_t \sim D}(h_j) \wedge L_{S_t \sim D}(h_i) < min_{h_j \in H/B} L_{S_t \sim D}(h_j)\right) \tag{3}$$

One way to look at the definitions of the expected values E_t and E_{t+1} given by Equation 2, is that they are simply the probability of M returning a hypothesis multiplied by the true error of that hypothesis, given some i.i.d. sample $S_t \sim D$.

To proof monotonicity of M for $|H| = 2$, the expected true error of M at training size t needs to be larger or equal to the expected error of M at training size $t + 1$. In other words, the equation below needs to hold.

$$E_t - E_{t+1} = (p^t_{[1]} - p^{t+1}_{[1]})\epsilon_1 + (p^t_{[2]} - p^{t+1}_{[2]})\epsilon_2 + (p^t_{[1,2]} - p^{t+1}_{[1,2]})(\frac{1}{2}\epsilon_1 + \frac{1}{2}\epsilon_2) \geq 0 \tag{4}$$

For $|H| = k$, we can express E_t as follows:

$$E_t = \sum_{B \subseteq H} p^t_B \sum_{h_i \in B} \frac{1}{|B|}\epsilon_i \quad \text{(with } B \neq \emptyset) \tag{5}$$

In a similar fashion to the derivation of Equation 4, an expression for $E_t - E_{t+1}$ can also be derived for the case $|H| = k$. Please see the equation below.

$$E_t - E_{t+1} = \sum_{B \subseteq H} (p^t_B - p^{t+1}_B)(\frac{1}{|B|}\sum_{i \in B} \epsilon_i) \geq 0 \tag{6}$$

The proof of Equation 4 and 6 is equivalent to the proof of Theorem 1 and 2. To foreshadow the main clue of these proofs; it is useful to find an exact expression for p^{t+1}_B that contains p^t_B, so that we may rewrite $p^t_B - p^{t+1}_B$ into something more digestible.

B. Exact Expression p^t_B

In this subsection an exact expression is given for p^t_B, first for $H = \{h_1, h_2\}$ and afterwards for $H = \{h_1, h_2, ..., h_k\}$. Equation 3 shows p^t_B is the probability that all hypotheses in B are the best performing on a given sample S_t. So how does one go about finding an exact expression for p^t_B?

1. Exact Expression of p^t_B for $H = \{h_1, h_2\}$

First, the notion of a "classification outcome" is introduced. For a sample S_t and $H = \{h_1, h_2\}$, a classification outcome is a pair of nonnegative integers (n_1, n_2) denoting the amount of instances classified correctly by h_1 and h_2 respectively. It can also tell us which hypothesis algorithm M chooses to return: if $n_1 > n_2$ it returns h_1, if $n_2 > n_1$ it returns h_2, and if $n_1 = n_2$ it returns either h_1 or h_2 with equal probability. Please see Table 1.

For any classification outcome (n_1, n_2) of sample S_t, $P_t(n_1, n_2)$ denotes the probability of that classification outcome. With the help of Table 1, an exact definition of $P_t(n_1, n_2)$ is given below in Equation 7.

$$P_t(n_1, n_2) = \binom{t}{n_1}(1 - \epsilon_1)^{n_1}\epsilon_1^{t-n_1}\binom{t}{n_2}(1 - \epsilon_2)^{n_2}\epsilon_2^{t-n_2} \tag{7}$$

With the definition of a classification outcome (n_1, n_2) and its associated probability $P_t(n_1, n_2)$, p^t_B can now be defined. Note that $p^t_{[1]}$ - the probability that h_1 has the lowest error rate on a sample S_t - is simply the

Table 1. Definition ϵ and n for any $h \in H$. The first column indicates whether a hypothesis classifies the instance correct 1 or incorrect 0. The second column denotes the associated probability of that classification. For example, the probability that h_i classifies any random instance correctly is $(1-\epsilon_i)$, with ϵ_i the true error rate of h_i. The third column denotes the variable used to count how many instances of a sample are classified correctly or incorrectly. So for a sample S_{10} for which h_1 classifies the first 3 instances correctly and h_2 the first 4 instances correctly (and the other instances thus incorrectly), $n_1 = 3$ and $n_2 = 4$. The probability of this event is then $(1-\epsilon_1)^3\epsilon_1^7(1-\epsilon_2)^4\epsilon_2^6$.

h_i	probability of classification	associated instance count
0	ϵ_i	n_i
1	$1-\epsilon_i$	$t-n_i$

sum of all probabilities of classification outcomes for which $n_1 < n_2$. This allows us to write $p^t_{[1]}$ (and $p^t_{[2]}$ and $p^t_{[1,2]}$) as follows:

$$p^t_{[1]} = \sum_{n_1=1}^{t}\sum_{n_2=0}^{n_1-1} P_t(n_1,n_2) \qquad p^t_{[2]} = \sum_{n_2=1}^{t}\sum_{n_1=0}^{n_2-1} P_t(n_1,n_2)$$
$$p^t_{[1,2]} = \sum_{n_1=0}^{t} P_t(n_1,n_2) \text{ with } n_2 = n_1 \tag{8}$$

2. Exact Expression of p^t_B for $H = \{h_1, h_2, ..., h_k\}$
A similar argument is made for $H = \{h_1, h_2, ..., h_k\}$ as is made for $H = \{h_1, h_2\}$. First we redefine $P_t(n_1, n_2)$ so that it includes n_i values for all hypotheses $h_i \in H$.

$$P_t(a) = \prod_{i=1}^{k}\binom{t}{i}(1-\epsilon_i)^{n_i}\epsilon_i^{t-n_i} \quad \text{with } a = \{n_1, n_2, ...n_k\} \tag{9}$$

Next, the definition for P^t_B in terms of $P_t(n_1, n_2, ...n_k)$ is given.

$$p^t_B = \sum_{a\in A} P_t(a) \qquad \text{with}$$
$$A = \{\{n_1, n_2, ..., n_k\} \in \mathbb{Z}^k_{0\le t} | \forall n_i, n_j \in \{n_1, n_2, ..., n_k\} : \begin{cases} n_i > n_j, & \text{if } h_i \in B, h_j \notin B \\ n_i = n_j, & \text{if } h_i, h_j \in B \end{cases} \tag{10}$$

C. Exact Expression p^{t+1}_B
The same derivation is made as in Section V.B to find an exact expression for p^{t+1}_B for $H = \{h_1, h_2\}$. Please see Equation 11 below.

$$p^{t+1}_{[1]} = \sum_{n_1=1}^{t+1}\sum_{n_2=0}^{n_1-1} P_{t+1}(n_1,n_2) \qquad p^{t+1}_{[2]} = \sum_{n_2=1}^{t+1}\sum_{n_1=0}^{n_2-1} P_{t+1}(n_1,n_2),$$
$$p^{t+1}_{[1,2]} = \sum_{n_1=0}^{t+1} P_{t+1}(n_1,n_2) \text{ with } n_2 = n_1 \tag{11}$$

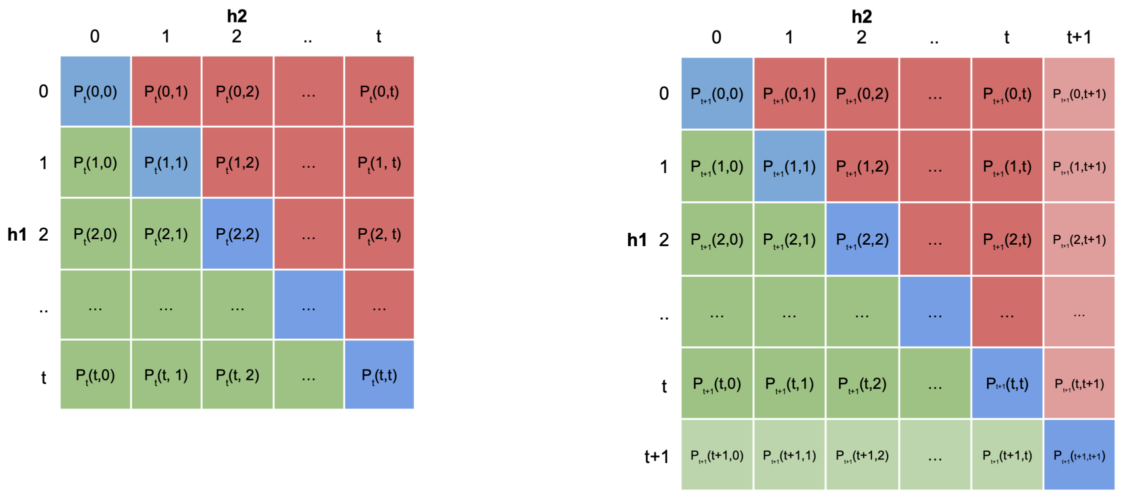


Figure 4. Visualisation of p_B^t (left) and p_B^{t+1} (right). A single box denotes the probability mass of a classification outcome $P_t(n_1, n_2)$. All green boxes together from the probability mass of $p_{[1]}^t$, all red boxes of $p_{[2]}^t$, all blue boxes $p_{[1,2]}^t$.

Whilst this does give an exact expression for p_B^{t+1}, unfortunately it is not the easiest way to express $p_B^t - p_B^{t+1}$; writing both p_B^{t+1} and p_B^t out and subtracting gets messy quickly, especially for large sizes of H. What is a better way to approach this?

1. Informal Explanation Inflow and Outflow
First, a visual representation of the probability mass of p_B^t and p_B^{t+1} is given. Please see the visualisation for $H = \{h_1, h_2\}$ in Figure 4

As all $t + 1$ instances of sample S_{t+1} are i.i.d. sampled, the probability of any classification outcome of S_{t+1} can be defined as the probability of a classification outcome of S_t with the addition of a classification outcome of a single additional instance. For example, there are four components that make up $P_{t+1}(1, 1)$ of S_{t+1}. These are: $P_1(1, 1,)P_t(0, 0)$, $P_1(1, 0)P_t(0, 1)$, $P_1(0, 1)P_t(1, 0)$, and $P_1(0, 0)P_t(1, 1)$.

Another way to look at this is that, for $H = \{h_1, h_2\}$, the probability mass of a classification outcome $P_t(n_1, n_2)$ is redistributed to 4 different classification outcomes of S_{t+1}. See Figure 5.

The redistribution of probability mass, when going from sample size t to $t + 1$, we describe here with what we refer to as flows. Consider, for instance, the probability mass indicated by A in Figure 5. The red arrow indicates that part of that mass will be redistributed from $p_{[1,2]}^t$ into $p_{[1]}^{t+1}$, which we call 'inflow'. For B, some probability mass flows out of $p_{[1]}^t$ into $p_{[1,2]}^{t+1}$, which we call 'outflow'. For C, there is no flow from $p_{[1]}^t$ into $p_{[1,2]}^{t+1}$ or vice versa: no matter the outcome of the additional instance, all probability mass of $p_{[1]}^t$ stays in $p_{[1]}^{t+1}$.

So why is this any useful? Well, as stated before, our objective is to write p_B^{t+1} in terms of p_B^t. By identifying the border region of $p_{[1]}^t$, the part of the probability mass that can "flow out" of $p_{[1]}^t$ into $p_{[1,2]}^{t+1}$, we can write $p_{[1]}^{t+1} = p_{[1]}^{t+1} - O_{[1]\to[1,2]} + I_{[1,2]\to[1]}$. Here $O_{[1]\to[1,2]}$ is the outflow of probability mass from $p_{[1]}^t$ into $p_{[1,2]}^{t+1}$, and $I_{[1,2]\to[1]}$ the inflow of $p_{[1,2]}^t$ into $p_{[1]}^{t+1}$. This is exactly what we want, an expression of $p_{[1]}^{t+1}$ in terms of $p_{[1]}^t$, which allows us to write $p_{[1]}^t - p_{[1]}^{t+1} = O_{[1]\to[1,2]} - I_{[1,2]\to[1]}$.

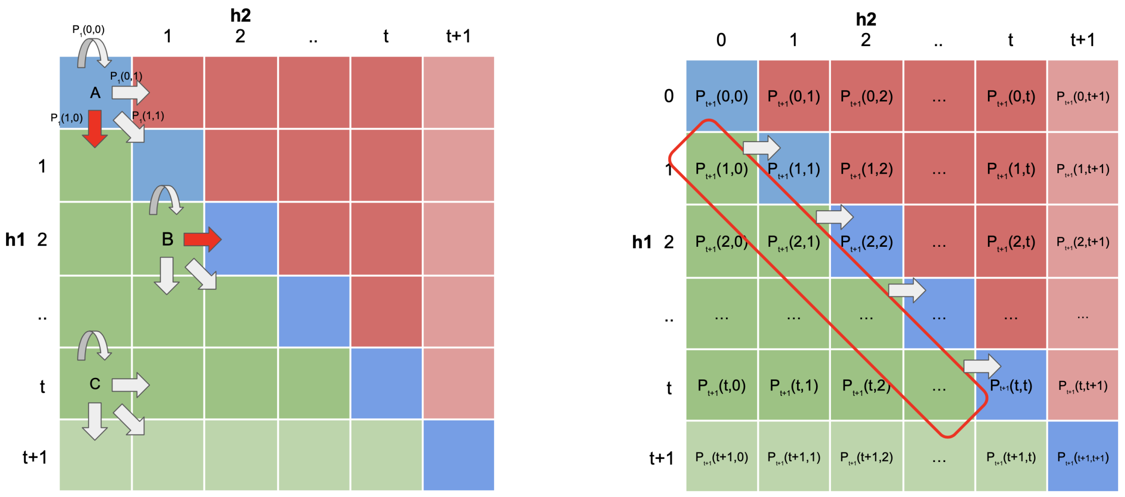

h2
h1
A
B
C
P_t(0,0)
P_t(0,1)
P_t(1,0)
P_t(1,1)
0 1 2 .. t t+1
P_{t+1}(0,0) P_{t+1}(0,1) P_{t+1}(0,2) ... P_{t+1}(0,t) P_{t+1}(0,t+1)
P_{t+1}(1,0) P_{t+1}(1,1) P_{t+1}(1,2) ... P_{t+1}(1,t) P_{t+1}(1,t+1)
P_{t+1}(2,0) P_{t+1}(2,1) P_{t+1}(2,2) ... P_{t+1}(2,t) P_{t+1}(2,t+1)
P_{t+1}(t,0) P_{t+1}(t,1) P_{t+1}(t,2) ... P_{t+1}(t,t) P_{t+1}(t,t+1)
P_{t+1}(t+1,0) P_{t+1}(t+1,1) P_{t+1}(t+1,2) ... P_{t+1}(t+1,t) P_{t+1}(t+1,t+1)

Figure 5. Flow of probability mass from $P_t(n_1, n_2)$ to $P_{t+1}(n'_1, n'_2)$. Left: Arrows indicate the direction of 'flow' of the probability mass $P_t(n_1, n_2)$ to $P_{t+1}(n'_1, n'_2)$. The probability mass $P_t(n_1, n_2)$ is completely redistributed in four directions: $P_t(n_1, n_2) = P_1(0,0)P_t(n_1, n_2) + P_1(1,0)P_t(n_1, n_2) + P_1(0,1)P_t(n_1, n_2) + P_1(1,1)P_t(n_1, n_2)$. **Right:** The border region of $p^t_{[1]}$ (red encircled), the only part of the probability mass of $p^t_{[1]}$ that can be redistributed to $p^{t+1}_{[1,2]}$.

2. Exact Definition Outflow and Inflow

We rewrite Equation 8 to Equation 12 as shown below. Notice that the border region of $p^t_{[1]}$ is highlighted in green, the border of $p^t_{[2]}$ in red, and the border of $p^t_{[1,2]}$ in blue. Black highlighted parts of Equation 12 signify parts of the probability mass that will not 'flow' from one p^t_B to another $p^{t+1}_{B'}$.

$$
\begin{aligned}
p^t_{[1]} &= \left(\sum_{n_1=1}^{t} P_t(n_1, n_2 = n_1 - 1)\right) + \left(\sum_{n_1=2}^{t}\sum_{n_2=0}^{n_1-2} P_t(n_1, n_2)\right)\\
p^t_{[2]} &= \left(\sum_{n_2}^{t} P_t(n_1 = n_2 - 1, n_2)\right) + \left(\sum_{n_2=2}^{t}\sum_{n_1=0}^{n_2-2} P_t(n_1, n_2)\right)\\
p^t_{[1,2]} &= \sum_{n_1=0}^{t} P_t(n_1, n_2 = n_1)
\end{aligned}
\tag{12}
$$

The outflow and inflow can now be easily determined: simply take the probability mass of the border region of one of the p^t_B highlighted in color, and multiply it by the probability of the classification outcome of a single instance that takes it to another $p^{t+1}_{B'}$.

With $\epsilon_{00} = \epsilon_1\epsilon_2$, $\epsilon_{10} = (1-\epsilon_1)\epsilon_2$, $\epsilon_{01} = \epsilon_1(1-\epsilon_2)$, and $\epsilon_{11} = (1-\epsilon_1)(1-\epsilon_2)$, $p^{t+1}_{B'}$ is defined for $H = \{h_1, h_2\}$.

$$
\begin{aligned}
p^{t+1}_{[1]} =& (\epsilon_{00}+\epsilon_{10}+\epsilon_{11})\left(\sum_{n_1=1}^{t} P_t(n_1, n_2=n_1-1)\right) + (\epsilon_{00}+\epsilon_{01}+\epsilon_{10}+\epsilon_{11})\left(\sum_{n_1=2}^{t}\sum_{n_2=0}^{n_1-2} P_t(n_1,n_2)\right) \\
&+\epsilon_{10}\sum_{n_1=0}^{t} P_t(n_1, n_2=n_1) \\
p^{t+1}_{[2]} =& (\epsilon_{00}+\epsilon_{01}+\epsilon_{11})\left(\sum_{n_2}^{t} P_t(n_1=n_2-1, n_2)\right) + (\epsilon_{00}+\epsilon_{01}+\epsilon_{10}+\epsilon_{11})\left(\sum_{n_2=2}^{t}\sum_{n_1=0}^{n_2-2} P_t(n_1,n_2)\right) \\
&+\epsilon_{01}\sum_{n_1=0}^{t} P_t(n_1, n_2=n_1) \\
p^{t+1}_{[1,2]} =& (\epsilon_{00}+\epsilon_{11})\sum_{n_1=0}^{t} P_t(n_1, n_2=n_1) + \epsilon_{01}\left(\sum_{n_1=1}^{t} P_t(n_1, n_2=n_1-1)\right) + \epsilon_{10}\left(\sum_{n_2}^{t} P_t(n_1=n_2-1, n_2)\right)
\end{aligned} \tag{13}
$$

Notice that red corresponds to outflow $O[2] \longrightarrow [1,2]$, green to outflow $O[1] \longrightarrow [1,2]$, and blue to inflow $I[1,2] \longrightarrow [1]$ and $I[1,2] \longrightarrow [2]$.

For the general case $H = \{h_1, h_2, ..., h_k\}$, Inflow and Outflow are formally defined as follows.

Definition 2 (Inflow) *For any three sets of classifying hypotheses H, V, W, with $V \subseteq H$ and $W \subset V$, and any i.i.d. random samples $S_t \sim D$,* ***the inflow*** *$I_{V\to W}$ is defined as:*

$$
\begin{aligned}
&I_{V\to W} = \prod_{h_x\in H} b_x \sum_{a\in A}^{t} P_t(a) && \textit{with} \\
&b_x = \begin{cases} (1-\epsilon_j), & \textit{if } h_j \in W \\ \epsilon_j, & \textit{if } h_j \in V/W \\ (1-\epsilon_j)+\epsilon_j, & \textit{otherwise} \end{cases} && \textit{and} \\
&A = \{\{n_1, n_2, ..., n_k\} \in \mathbb{Z}^k_{0\le t} | \forall n_i, n_j \in \{n_1, n_2, ..., n_k\} : \begin{cases} n_i > n_j + 1, & \textit{if } h_i \in V, h_j \notin V \\ n_i = n_j & \textit{if } h_i, h_j \in V \end{cases}
\end{aligned} \tag{14}
$$

Definition 3 (Outflow) *For any three sets of classifying hypotheses H, V, W, with $V \subseteq H$ and $W \subset V$, and* *any i.i.d. random samples $S_t \sim D$,* ***the outflow*** *$O_{W\to V}$ is defined as:*

$$O_{W\to V} = \prod_{h_x\in H} b_x \sum_{a\in A^2}^{t} P_t(a) + \prod_{h_y\in H} b_y \sum_{a\in A^1}^{t} P_t(a) \qquad \textit{with}$$
$$b_x = \begin{cases}(1-\epsilon_x), & \textit{if } h_x \in V/W\\ \epsilon_x, & \textit{if } h_x \in W\\ (1-\epsilon_x)+\epsilon_x, & \textit{otherwise}\end{cases}, b_y = \begin{cases}(1-\epsilon_y), & \textit{if } h_y \in V/W\\ \epsilon_y & \textit{otherwise}\end{cases} \qquad \textit{and}$$
$$A^2 = \{\{n_1,n_2,...,n_k\} \in \mathbb{Z}^k_{0\le t} | \forall n_i, n_j \in \{n_1,n_2,...,n_k\} : \begin{cases} n_i = n_j + 1, & \textit{if } h_i \in W, h_j \in V/W\\ n_i > n_j + 1, & \textit{if } h_i \in W, h_j \notin W\\ n_i = n_j & \textit{if } h_i, h_j \in W\end{cases}$$
$$\textit{and}$$
$$A^1 = \{\{n_1,n_2,...,n_k\} \in \mathbb{Z}^k_{0\le t} | \forall n_i, n_j \in \{n_1,n_2,...,n_k\} : \begin{cases} n_i = n_j, & \textit{if } h_i, h_j \in W\\ n_i = n_j + 1 & \textit{otherwise}\end{cases} \tag{15}$$

Armed with the definitions of inflow and outflow, we can now finally give an expression for p_B^{t+1} in terms of p_B^{t+1} for $H = \{h_1, h_2\}$.

$$\begin{aligned} p^{t+1}_{[1]} &= p^t_{[1]} + I_{[1,2]\to[1]} - O_{[1]\to[1,2]}\\ p^{t+1}_{[2]} &= p^t_{[2]} + I_{[1,2]\to[2]} - O_{[2]\to[1,2]}\\ p^{t+1}_{[1,2]} &= p^t_{[1,2]} - I_{[1,2]\to[1]} - I_{[1,2]\to[2]} + O_{[1]\to[1,2]} + O_{[2]\to[1,2]} \end{aligned} \tag{16}$$

With this, Equation 4 is rewritten to Equation 17 below. Proving Equation 17 is equivalent to proving Theorem 1.

$$\begin{aligned} E_t - E_{t+1} =&(O_{[1]\to[1,2]} - I_{[1,2]\to[1]})\epsilon_{[1]}\\ &+(O_{[2]\to[1,2]} - I_{[1,2]\to[2]})\epsilon_{[2]}\\ &+(I_{[1,2]\to[1]} - O_{[1]\to[1,2]} + I_{[1,2]\to[2]} - O_{[2]\to[1,2]})(\frac{1}{2}\epsilon_{[1]} + \frac{1}{2}\epsilon_{[2]})\\ \ge&0 \end{aligned} \tag{17}$$

It can further be simplify to:

$$\begin{aligned} E_t - E_{t+1} =&I_{[1,2]\to[1]}(\frac{-1}{2}\epsilon_1 + \frac{1}{2}\epsilon_2)\\ &+I_{[1,2]\to[2]}(\frac{1}{2}\epsilon_1 + \frac{-1}{2}\epsilon_2)\\ \ge&0 \end{aligned} \tag{18}$$

Equation 18 might seem to come a bit unexpectedly; there are no outflow terms in Equation 18. This is no accident, as all outflow terms together equate to zero. Section V.D proves this in the form of Lemma 1 and Lemma 2.

For $H = \{h_1, h_2, ...h_k\}$, things are a bit more complicated; there is not only inflow and outflow between $p_{[1]}$, $p_{[2]}$, and $p_{[1,2]}$, but any two sets $V, W \subseteq H$ with $W \subset V$. Please see the closed form below, where $[A]^i$ is the notation for all unique subsets of A of size i.

$$E_t - E_{t+1} = \sum_{i=2}^{k} \sum_{V=[H]^i} \sum_{j=1}^{i-1} \sum_{W=[V]^j} I_{V\to W} \left((\sum_{h_x \in V/W} \frac{1}{i}\epsilon_x) + (\sum_{h_y \in W} -\frac{i-j}{ij}\epsilon_y) \right) \\ \geq 0 \tag{19}$$

For a full derivation of Equation 19, see Appendix F.

D. Inflow Inequality and Outflow Equality

Lemma 1 (Outflow Equality) *For two outflows $O_{B\to A}$ and $O_{C\to A}$, if both hypothesis sets B and C contain an equal amount of hypotheses, so $|B| = |C|$, then they are equal: $O_{B\to A} = O_{C\to A}$.*

Lemma 2 (Inflow Inequality) *For two outflows $I_{A\to B}$ and $I_{A\to C}$, with both hypothesis sets B and C containing an equal amount of hypotheses, so $|B| = |C|$, and B and C differing by one hypothesis $h_b = B/C, h_c = C/B$, then if $L_{S_{t+1}}(h_b) \leq L_{S_{t+1}}(h_c)$, it must hold that $I_{A\to B} \geq I_{A\to C}$.*

As a warmup, Appendix B and C proof that for $H = \{h_1, h_2\}$, Lemma 1 and Lemma 2 hold. For $H = \{h_1, h_2, ..., h_k\}$, the general case, Appendix D and E are provided.

E. Theorem 1

1. To Prove

$$E_t - E_{t+1} \geq 0 \tag{20}$$

2. Proof

$$\begin{aligned}
E_t - E_{t+1} =& (p^t_{[1]} - p^{t+1}_{[1]})\epsilon_1 + (p^t_{[2]} - p^{t+1}_{[2]})\epsilon_2 + (p^t_{[1,2]} - p^{t+1}_{[1,2]})(\frac{1}{2}\epsilon_1 + \frac{1}{2}\epsilon_2) && \text{(by Equation 4)} \\
=& (O_{[1]\to[1,2]} - I_{[1,2]\to[1]})\epsilon_{[1]} + (O_{[2]\to[1,2]} - I_{[1,2]\to[2]})\epsilon_{[2]} && \text{(by Definition 2, 3)} \\
& + (I_{[1,2]\to[1]} - O_{[1]\to[1,2]} + I_{[1,2]\to[2]} - O_{[2]\to[1,2]})(\frac{1}{2}\epsilon_{[1]} + \frac{1}{2}\epsilon_{[2]}) \\
=& I_{[1,2]\to[1]}(\frac{-1}{2}\epsilon_1 + \frac{1}{2}\epsilon_2) + I_{[1,2]\to[2]}(\frac{1}{2}\epsilon_1 + \frac{-1}{2}\epsilon_2) \\
& + (\frac{1}{2}O_{[1]\to[1,2]} - \frac{1}{2}O_{[2]\to[1,2]})\epsilon_1 + (\frac{1}{2}O_{[2]\to[1,2]} - \frac{1}{2}O_{[1]\to[1,2]})\epsilon_2 \\
=& I_{[1,2]\to[1]}(\frac{-1}{2}\epsilon_1 + \frac{1}{2}\epsilon_2) + I_{[1,2]\to[2]}(\frac{1}{2}\epsilon_1 + \frac{-1}{2}\epsilon_2) && \text{(by Lemma 1)} \\
=& I_{[1,2]\to[1]}(A) + I_{[1,2]\to[2]}(-A) && \text{(with } A = \frac{-1}{2}\epsilon_1 + \frac{1}{2}\epsilon_2) \\
& \geq 0 && \text{(by Lemma 2)}
\end{aligned} \tag{21}$$

$\therefore$ As $E_t - E_{t+1} \geq 0$, the expected true error of Algorithm 1 does not increase with more training data. QED Theorem 1 holds.

F. Theorem 2

1. To Prove

$$E_t - E_{t+1} \geq 0 \tag{22}$$

2. Proof

$$
\begin{aligned}
E_t - E_{t+1} &= \sum_{B \subseteq H} (p^t_B - p^{t+1}_B)(\frac{1}{|B|}\sum_{i\in B}\epsilon_i) && \text{(by Equation 6)}\\
&= \sum_{i=2}^{k}\sum_{V=[H]^i}\sum_{j=1}^{i-1}\sum_{W=[V]^j} I_{V\to W}\left((\sum_{h_x\in V/W}\frac{1}{i}\epsilon_x) + (\sum_{h_y\in W} -\frac{i-j}{ij}\epsilon_y)\right) && \text{(by Equation 19)}\\
&= \sum_{i=2}^{k}\sum_{V=[H]^i}\sum_{j=1}^{i-1}\sum_{\{W,W'\}\in\{[V]^j\}^2_*}\Bigg(I_{V\to W}(\frac{1}{ij}h_{W'/W} - \frac{1}{ij}h_{W/W'}) && \text{(see note below)}\\
&\quad + I_{V\to W'}(\frac{1}{ij}h_{W/W'} - \frac{1}{ij}h_{W'/W})\Bigg) && \text{(see Appendix G)}\\
&\geq 0 && \text{(by Lemma 2)} \qquad (23)
\end{aligned}
$$

$\therefore$ As $E_t - E_{t+1} \geq 0$, the expected true error of Algorithm 1 does not increase with more training data. QED Theorem 2 holds

Note: $\{[V]^j\}^2_*$ is the group of all unique pairs of subsets of V of size j that have all overlapping elements except one. So for $W \subset V$, $W' \subset V$, $|W| = |W'| = j$ and $|W/W'| = |W'/W| = 1$

G. Dependent Hypotheses and Growing Hypothesis Set

In its current form Algorithm 1 grows linearly $|S_t| = |S_{t-1}| + 1$. However, for both proofs of Theorem 1 and 2 it was assumed that for any two hypotheses h_i and h_j with respective true error rates ϵ_i and ϵ_j, the probability that both classify a random instance incorrectly is simply $\epsilon_i\epsilon_j$. This assumption of independent classification holds when each hypothesis h_i is given it's own i.i.d. drawn sample S^i_t to evaluate and compare error rates. The growth remains linear, going from $|S_t| = |S_{t-1}| + 1$ to $\sum_{h_i\in H}|S^i_t| = \sum_{h_i\in H}|S^i_{t-1}| + |H|$.

In practice however, hypotheses generally do not classify independently. Joint classification probabilities must be defined, and so monotonicity does not necessarily hold for a single validation set. It should be noted that for $H = \{h_1, h_2\}$, Algorithm 1 is monotone, using only a single validation set S_t. See Appendix H.A for a proof of this using joint probabilities.

For a growing hypothesis set $|H_{t+1} = |H| + 1$, it is shown Algorithm 1 is not monotone, even if hypotheses classify independently. Imagine two hypotheses h_1 and h_2 that classify almost identically, so $\epsilon_1 \approx \epsilon_2$. The expected value at sample size t is then $E_t \approx p^t_{[1,2]}\epsilon_1$, with $p^t_{[1,2]} \approx 1$. Now imagine that at sample size $t+1$ a new hypothesis h_3 is introduced with $\epsilon_3 > \epsilon_1$. We can now show $E_{t+1} \approx p^{t+1}_{[1,2]}\epsilon_1 + p^{t+1}_{[3]}\epsilon_3$, which is larger than E_t when $p^{t+1}_{[3]} > 0$.

VI. Discussion

It is proven that Algorithm 1 is monotone for an immutable set of hypotheses H. As such, a new monotone learning algorithm is introduced that randomly produces k hypotheses at the start, and becomes (monotonically) better at returning the best hypothesis from this set with more data. However, this new monotone machine learning algorithm's function is very limited. Firstly, randomly generating a finite amount of hypotheses and selecting the best performing hypothesis in a near infinite hypothesis space is not an effective strategy for learning a good solution. Instead the hypothesis set should grow at each step $|H_{t+1}| = |H_t| + 1$, with each additional hypothesis being learned from data by a machine learning algorithm as normally done. It is shown in Section V.G that Algorithm 1 does not produce a monotone learning curve in this case. Furthermore, the linear growth of the validation set S_t with $|S_{t+1}| = |S_t| + k$ is directly tied to the amount of hypotheses k, which causes data inefficiency; the more hypotheses you initialise, the more validation data you need at each step. Ideally, the size of the validation set is not dependent on the amount of hypotheses generated, and can grow linearly as $|S_t| = |S_{t-1}| + 1$, as is the case for $H = \{h_1, h_2\}$. Even though this work is not an effective monotone machine learning algorithm or monotone wrapper, this work may be a step in the right direction. By growing the validation sample linearly, one can monotonically improve at choosing from an existing set of hypotheses H. In order to turn Algorithm 1 into a monotone wrapper algorithm using linearly growing validation data, one only needs to find a solution for the non-monotonicity of introducing a new hypothesis to the hypothesis set.

For future research, it would be interesting to find this solution for the expanding hypothesis set $|H_{t+1}| = |H_t| + 1$. Finding such a wrapper would mean one can use a machine learning algorithm to learn from training data and then evaluate the produced hypotheses with said wrapper. This is much more efficient in finding a good solution than randomly generating hypotheses at the start, and then evaluating them. One possible solution is to use an upper bound of α for the probability of choosing a new, worse performing hypothesis, which might possibly be done with the McNemar test as in Viering's work[6, 10]. Using an upper bound in conjunction with an expanding validation data set allows the upper bound α to become smaller over time, decreasing the probability of making a non-monotone decision. This decrease then amounts to a monotone learning curve. Another option that could produce a linear-data monotone wrapper would be to find the lower bound of $E_t - E_{t+1}$. Once this is known, a regularisation term can be used to increase the expected error E_t so that $E_t \geq E_{t+1}$ in the worst case, forcing the wrapper to be monotone. This method is similar to Bousquet[8]. Both of these avenues of exploration could successfully yield a linear-data monotone wrapper algorithm, and were considered but not explored due to time constraints.

Additionally, finding a wrapper algorithm that bounds the learning curve to be not only monotone, but also fit a parametric model could be very interesting for extrapolation methods, allowing much of the parametric model selection to be skipped. As shown in Appendix I, at a quick glance it seems Algorithm 1 already yields an easily parameterisable learning curve. Future work could look at what functions fit the learning curves of Algorithm 1 best.

In a broader perspective, we may ask whether monotone wrapper algorithms offer substantial improvement over normal machine learning algorithms. When it comes to learning curve analysis, monotonicity of learning curves is listed as a desirable trait in order to extrapolate learning curves or to perform model selection [2]. However, as Figure 3a shows, a monotone learning curve may not necessarily be more suited to be parametrically fitted, as the wrapper warps the shape of the learning curve severely. Is monotonicity alone truly the solution for making learning curves more interpretable? Or should there be additional requirements for 'well-behaved' learning curves? Furthermore, if we are interested in selecting the best possible hypothesis, instead of forcing the learning curve to be monotone, it may be more efficient to simply analyse all hypotheses on the learning curve on a single validation set at the end of training. We make the case that in general, one should not train on a single training size, but on multiple training sizes and construct a learning curve.

VII. Conclusion

In conclusion, we have presented a simple algorithm that constructs a monotone learning curve by choosing the best performing hypothesis out of set of hypotheses H based on a validation sample, and doing this for multiple validation sample sizes. We show that simple choosing process is monotone in Theorem 1 and Theorem 2 when H is immutable. It is also shown that the algorithm is not monotone for an expanding set of hypotheses. Even though our algorithm alone is not enough to create a fully functioning linear-data monotone wrapper algorithm, we hope our work provides insight for future works to address this issue.

References

[1] Loog, M., and Duin, R. P. W., "The Dipping Phenomenon," *Structural, Syntactic, and Statistical Pattern Recognition*, Vol. 7626, edited by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum,

G. Gimel'farb, E. Hancock, A. Imiya, A. Kuijper, M. Kudo, S. Omachi, T. Windeatt, and K. Yamada, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 310–317. URL `http://link.springer.com/10.1007/978-3-642-34166-3_34`, series Title: Lecture Notes in Computer Science.

[2] Viering, T., and Loog, M., "The Shape of Learning Curves: a Review," , Mar. 2021. URL `http://arxiv.org/abs/2103.10948`, arXiv:2103.10948 [cs].

[3] Krijthe, J. H., and Loog, M., "The Peaking Phenomenon in Semi-supervised Learning," , 2016.

[4] Pestov, V., "A universally consistent learning rule with a universally monotone error," *CoRR*, 2021, p. 27.

[5] Mhammedi, Z., "Risk-Monotonicity in Statistical Learning," , Jan. 2022. URL `http://arxiv.org/abs/2011.14126`, arXiv:2011.14126 [cs, math, stat].

[6] Viering, T. J., Mey, A., and Loog, M., "Making Learners (More) Monotone," *Advances in Intelligent Data Analysis XVIII*, Vol. 12080, edited by M. R. Berthold, A. Feelders, and G. Krempl, Springer International Publishing, Cham, 2020, pp. 535–547. URL `http://link.springer.com/10.1007/978-3-030-44584-3_42`, series Title: Lecture Notes in Computer Science.

[7] Devroye, L., Györfi, L., and Lugosi, G., *A Probablistic Theory of Pattern Recognition*, Vol. 31, Springer, 1996. Journal Abbreviation: NewYork: Springer Verlag Publication Title: NewYork: Springer Verlag.

[8] Bousquet, O., Daniely, A., Kaplan, H., Mansour, Y., Moran, S., and Stemmer, U., "Monotone Learning," , Aug. 2022. URL `http://arxiv.org/abs/2202.05246`, arXiv:2202.05246 [cs, math, stat].

[9] Viering, T., Mey, A., and Loog, M., "Open Problem: Monotonicity of Learning," *Proceedings of the Thirty-Second Conference on Learning Theory*, PMLR, 2019, pp. 3198–3201. URL `https://proceedings.mlr.press/v99/viering19a.html`, iSSN: 2640-3498.

[10] Fagerland, M. W., Lydersen, S., and Laake, P., "The McNemar test for binary matched-pairs data: mid-p and asymptotic are better than exact conditional," *BMC Medical Research Methodology*, Vol. 13, No. 1, 2013, p. 91. doi: 10.1186/1471-2288-13-91, URL `https://doi.org/10.1186/1471-2288-13-91`.

[11] Mhammedi, Z., and Koolen, W. M., "Lipschitz and Comparator-Norm Adaptivity in Online Learning," , Aug. 2020. doi: 10.48550/arXiv.2002.12242, URL `http://arxiv.org/abs/2002.12242`, arXiv:2002.12242 [cs, stat].

[12] Kolachina, P., Cancedda, N., Dymetman, M., and Venkatapathy, S., "Prediction of Learning Curves in Machine Translation," , Jul. 2012. URL `https://aclanthology.org/P12-1003`.

[13] Gu, B., Hu, F., and Liu, H., "Modelling Classification Performance for Large Data Sets," , Jul. 2001.

[14] Last, M., "Predicting and Optimizing Classifier Utility with the Power Law," , Oct. 2007. ISSN: 2375-9259.

[15] Singh, S., "Modeling Performance of Different Classification Methods: Deviation from the Power Law," , 2005.

[16] Frey, L. J., and Fisher, D. H., "Modeling decision tree performance with the power law," , Jan. 1999. URL `https://proceedings.mlr.press/r2/frey99a.html`, iSSN: 2640-3498.

[17] Cortes, C., Jackel, L. D., Solla, S., Vapnik, V., and Denker, J., "Learning Curves: Asymptotic Values and Rate of Convergence," , 1993. URL `https://papers.nips.cc/paper/1993/hash/1aa48fc4880bb0c9b8a3bf979d3b917e-Abstract.html`.

A. Notation

A. Refresher Notation

- $[]$, subset notation, $B = [H]^j$ is all unique subsets B of H of size j
- $/$, excluding notation, B/C denotes all elements in B not in C
- $()$, binomial coefficient, $\binom{n}{k} = \frac{n!}{k!(n-k)!}$
- $\overset{\$}{\leftarrow}$ is the uniform random selection operator

B. General Notation

- A is a machine learning algorithm, produces hypotheses h using a sample of data $h = A(S)$
- $\mathbb{E}_t = \mathbb{E}[L_D(A(S_t))]$ is the expected error on the underlying D of the hypothesis returned by $A(S_t)$
- h is a hypothesis, is be produced by a learner A
- H is a set of hypotheses
- L is an error function, which evaluates an estimate of Y by a hypothesis $L(h(X), Y)$
- $L_D(h)$ is the generalisation error of a hypothesis h on the underlying distribution D
- $L_{S_t}(h)$ is the population error of a hypothesis h on a sample with sample size t
- M is a monotone wrapper algorithm such that for any learner A, $M(A, S)$ is monotone
- S_t is a sample of size t

C. Paper specific notation

- ϵ_i is the generalisation error rate $L_D(h_i)$ of hypothesis $h_i \in H$
- p^t_B is the probability that on a sample S_t, all the hypotheses $h \in B$ have the same classification score on that sample, and better than all other hypotheses
- $P_t(n_{10}, n_{01})$ is the probability that for a sample S_t and a hypothesis class $H = \{h_1, h_2\}$, there are n_{10} instances which h_1 classifies correctly and h_2 incorrectly, and n_{01} instances which h_1 classifies incorrectly and h_2 correctly.
- $\rho_{b_1,b_2,\ldots,b_k}$ is the probability that a single instance is classified correctly by a hypothesis $b_i = 1$ and incorrectly by another $b_j = 0$ or either $b_s = *$ with $h_i, h_j, h_s \in H$ and $|H| = k$.
- $\{[V]^j\}^2_*$ is the group of all unique pairs of subsets of V of size j that have all overlapping elements except one. So for $W \subset V$, $W' \subset V$, $|W| = |W'| = j$ and $|W/W'| = |W'/W| = 1$.

B. Inflow Proof 2 Hypotheses

To prove

$$\epsilon_1 \leq \epsilon_2 \implies I_{[1,2]\rightarrow[1]} - I_{[1,2]\rightarrow[2]} \geq 0 \tag{24}$$

$$\epsilon_1 \geq \epsilon_2 \implies I_{[1,2]\rightarrow[1]} - I_{[1,2]\rightarrow[2]} \leq 0 \tag{25}$$

Proof Both $I_{[1,2]\rightarrow[1]}$ and $I_{[1,2]\rightarrow[2]}$ are defined, with all variables as shown in Table 1:

$$\begin{aligned} I_{[1,2]\rightarrow[1]} &= (1-\epsilon_1)\epsilon_2 p^t_{[1,2]} \\ &= (1-\epsilon_1)\epsilon_2 \sum_{i=0}^{t} P_t(i,i) \\ &= (1-\epsilon_1)\epsilon_2 \sum_{i=0}^{t} \left[\binom{t}{i}(1-\epsilon_1)^i \epsilon_1^{t-i}\binom{t}{i}(1-\epsilon_2)^i\epsilon_2^{t-i}\right] \end{aligned} \tag{26}$$

$$\begin{aligned} I_{[1,2]\rightarrow[2]} &= \epsilon_1(1-\epsilon_2) p^t_{[1,2]} \\ &= \epsilon_1(1-\epsilon_2) \sum_{i=0}^{t} P_t(i,i) \\ &= \epsilon_1(1-\epsilon_2) \sum_{i=0}^{t} \left[\binom{t}{i}(1-\epsilon_1)^i \epsilon_1^{t-i}\binom{t}{i}(1-\epsilon_2)^i\epsilon_2^{t-i}\right] \end{aligned} \tag{27}$$

Note that both $I_{[1,2]\rightarrow[1]}$ and $I_{[1,2]\rightarrow[1]}$ can be divided by $p^t_{[1,2]}$. All that remains is to proof:

$$\epsilon_1 \leq \epsilon_2 \rightarrow (1-\epsilon_1)(\epsilon_2) \geq (\epsilon_1)(1-\epsilon_2) \tag{28}$$

$$\epsilon_1 \geq \epsilon_2 \rightarrow (1-\epsilon_1)(\epsilon_2) \leq (\epsilon_1)(1-\epsilon_2) \tag{29}$$

Note that ϵ_1 and ϵ_2 are both ≤ 1 as they error rates, and can hence be written as $\frac{1}{a}$ and $\frac{1}{b}$ respectively, with $a, b \geq 1$. Thus $\epsilon_1 \leq \epsilon_2 \leftrightarrow a \geq b \rightarrow (1-\epsilon_1)(\epsilon_2) \geq (\epsilon_1)(1-\epsilon_2)$ and similarly $\epsilon_1 \geq \epsilon_2 \leftrightarrow a \leq b \rightarrow (1-\epsilon_1)(\epsilon_2) \leq (\epsilon_1)(1-\epsilon_2)$. We can then derive:

$$(1-\epsilon_1)(\epsilon_2) = (\frac{a-1}{a})(\frac{1}{b}) = (\frac{ab-b}{ab}) \tag{30}$$

$$(\epsilon_1)(1-\epsilon_2) = (\frac{1}{a})(\frac{b-1}{b}) = (\frac{ab-a}{ab}) \tag{31}$$

And thus equivalently:

$$a \geq b \implies \frac{ab-b}{ab} \geq (\frac{ab-a}{ab}) \tag{32}$$

$$a \leq b \implies \frac{ab-b}{ab} \leq (\frac{ab-a}{ab}) \tag{33}$$

With these definitions it is immediately apparent that both Equations 28 and 29 hold.

C. Outflow Proof 2 Hypotheses

To prove

$$O_{[1]\to[1,2]} = O_{[2]\to[1,2]} \tag{34}$$

Proof We define both $O_{[1]\to[1,2]}$ and $O_{[2]\to[1,2]}$ using a binomial coefficients:

$$\begin{aligned} O_{[1]\to[1,2]} &= \epsilon_1(1-\epsilon_2)\sum_{n_1=1}^{t} P_t(n_1, n_2 = n_1 - 1) \\ &= \epsilon_1(1-\epsilon_2)\sum_{n_1=1}^{t}\left[\binom{t}{n_1}(1-\epsilon_1)_1^n\epsilon_1^{t-n_1}\binom{t}{n_1-1}(1-\epsilon_2)^{n_1-1}\epsilon_2^{t+1-n_1}\right] \end{aligned} \tag{35}$$

$$\begin{aligned} O_{[2]\to[1,2]} &= (1-\epsilon_1)\epsilon_2\sum_{n_2=1}^{t} P_t(n_1 = n_2 - 1, n_2) \\ &= (1-\epsilon_1)\epsilon_2\sum_{n_2=1}^{t}\left[\binom{t}{n_2-1}(1-\epsilon_1)^{n_2-1}\epsilon_1^{t+1-n_2}\binom{t}{n_2}(1-\epsilon_2)^{n_2}\epsilon_2^{t-n_2}\right] \end{aligned} \tag{36}$$

With these definitions, a direct derivation is possible:

$$\begin{aligned} O_{[1]\to[1,2]} =&(\epsilon_1)(1-\epsilon_2)\sum_{n_1=1}^{t}\binom{t}{n_1}(1-\epsilon_1)_1^n\epsilon_1^{t-n_1}\binom{t}{n_1-1}(1-\epsilon_2)^{n_1-1}\epsilon_2^{t+1-n_1} \\ =&\sum_{n_1=1}^{t}\binom{t}{n_1}(1-\epsilon_1)_1^n\epsilon_1^{t+1-n_1}\binom{t}{n_1-1}(1-\epsilon_2)^{n_1}\epsilon_2^{t+1-n_1} \\ =&(1-\epsilon_1)(\epsilon_2)\sum_{n_2=1}^{t}\binom{t}{n_2-1}(1-\epsilon_1)^{n_2-1}\epsilon_1^{t+1-n_2}\binom{t}{n_2}(1-\epsilon_2)^{n_2-1}\epsilon_2^{t-n_2} \\ =&O_{[2]\to[1,2]} \end{aligned} \tag{37}$$

D. Inflow Proof k Hypotheses

To prove

For hypothesis set H, $B \subseteq H$, $W \subset B$, $W' \subset B$ with $|W| = |W'|$ and $|W \cap W'| = |W| - 1$.

$$\epsilon_{W/W'} \leq \epsilon_{W'/W} \implies I_{B \to W} - I_{B \to W'} \geq 0 \tag{38}$$

$$\epsilon_{W/W'} \geq \epsilon_{W'/W} \implies I_{B \to W} - I_{B \to W'} \leq 0 \tag{39}$$

Proof

We define both $I_{B \to W}$ and $I_{B \to W'}$.

$$I_{B \to W} = \prod_{i \in W} (1 - \epsilon_i) \prod_{j \in B/W} \epsilon_j \prod_{l \in H/B} ((1 - \epsilon_l) + \epsilon_l) p_B^t \tag{40}$$

$$I_{B \to W'} = \prod_{i \in W'} (1 - \epsilon_i) \prod_{j \in B/W'} \epsilon_j \prod_{l \in H/B} ((1 - \epsilon_l) + \epsilon_l) p_B^t \tag{41}$$

We divide both $I_{B \to W}$ and $I_{B \to W'}$ by $\prod_{l \in H/B} ((1 - \epsilon_l) + \epsilon_l) p_B^t$

$$I_{B \to W} = \prod_{i \in W} (1 - \epsilon_i) \prod_{j \in B/W} \epsilon_j \tag{42}$$

$$I_{B \to W'} = \prod_{i \in W'} (1 - \epsilon_i) \prod_{j \in B/W'} \epsilon_j \tag{43}$$

Since there is only one element in W/W' and only one element in W'/W, and all elements of W and W' are contained within B, the expression can be further simplified. Both $I_{B \to W}$ and $I_{B \to W'}$ are divided by their common factors.

$$I_{B \to W} = (1 - \epsilon_{W/W'}) \epsilon_{W'/W} \tag{44}$$

$$I_{B \to W'} = (1 - \epsilon_{W'/W}) \epsilon_{W/W'} \tag{45}$$

Wll that remains is to proof:

$$\epsilon_{W/W'} \leq \epsilon_{W'/W} \implies (1 - \epsilon_{W/W'}) \epsilon_{W'/W} \geq (1 - \epsilon_{W'/W}) \epsilon_{W/W'} \tag{46}$$

$$\epsilon_{W/W'} \geq \epsilon_{W'/W} \implies (1 - \epsilon_{W/W'}) \epsilon_{W'/W} \leq (1 - \epsilon_{W'/W}) \epsilon_{W/W'} \tag{47}$$

Note that $\epsilon_{W/W'}$ and $\epsilon_{W'/W}$ are both fractions, and can hence be written as $\frac{1}{a}$ and $\frac{1}{b}$ respectively, with $a, b \geq 1$. $\epsilon_{W/W'} \leq \epsilon_{W'/W}$ then implies $a \geq b$ and similarly $\epsilon_{W/W'} \geq \epsilon_{W'/W}$ then implies $a \leq b$. We can then derive:

$$(1 - \epsilon_{W/W'})(\epsilon_{W'/W}) = (\frac{a-1}{a})(\frac{1}{b}) = (\frac{ab-b}{ab}) \tag{48}$$

$$(\epsilon_{W/W'})(1 - \epsilon_{W'/W}) = (\frac{1}{a})(\frac{b-1}{b}) = (\frac{ab-a}{ab}) \tag{49}$$

Conclusion

Since $\epsilon_{W/W'} \leq \epsilon_{W'/W}$ implies $a \geq b$, it also implies $(\frac{ab-b}{ab}) > (\frac{ab-a}{ab})$, and similarly $\epsilon_{W/W'} \geq \epsilon_{W'/W}$ implies $(\frac{ab-b}{ab}) \leq (\frac{ab-a}{ab})$, proving both cases.

E. Outflow Proof k Hypotheses

To prove

For hypothesis set H, with $B \subseteq H$, $W \subset B$, $W' \subset B$ with $|W| = |W'|$

$$|W| = |W'| \implies O_{W\to B} = O_{W'\to B} \tag{50}$$

Proof

We define both $O_{W\to B} = O_{W'\to B}$ using a binomial coefficients.

$$\begin{aligned} O_{W\to B} =& \prod_{i\in B/W}(1-\epsilon_i)\prod_{j\in W}\epsilon_j\prod_{l\in H/B}((1-\epsilon_l)+\epsilon_l) \\ &\left[\sum_{a=2}^{t}\sum_{b=0}^{a-2}\left(\prod_{i\in B}(1-\epsilon_i)^a\epsilon_i^{t-a}\right)\left(\prod_{j\in W/B}(1-\epsilon_j)^{a-1}\epsilon_j^{t+1-a}\right)\left(\prod_{l\in H/B}(1-\epsilon_l)^b\epsilon_l^t\right)\right] \\ &+\prod_{i\in B/W}(1-\epsilon_i)\prod_{j\in W}\epsilon_j\prod_{l\in H/B}\epsilon_l \\ &\left[\sum_{a=1}^{t}\sum_{b=0}^{a-1}\left(\prod_{i\in B}(1-\epsilon_i)^a\epsilon_i^{t-a}\right)\left(\prod_{j\in W/B}(1-\epsilon_j)^{a-1}\epsilon_j^{t+1-a}\right)\left(\prod_{l\in H/B}(1-\epsilon_l)^b\epsilon_l^t\right)\right] \\ =& \prod_{l\in H/B}((1-\epsilon_l)+\epsilon_l)\left[\sum_{a=2}^{t}\sum_{b=0}^{a-2}\left(\prod_{i\in B}(1-\epsilon_i)^a\epsilon_i^{t+1-a}\right)\left(\prod_{j\in W/B}(1-\epsilon_j)^{a}\epsilon_j^{t+1-a}\right)\left(\prod_{l\in H/B}(1-\epsilon_l)^b\epsilon_l^t\right)\right] \\ &+\prod_{l\in H/B}\epsilon_l\left[\sum_{a=1}^{t}\sum_{b=0}^{a-1}\left(\prod_{i\in B}(1-\epsilon_i)^a\epsilon_i^{t+1-a}\right)\left(\prod_{j\in W/B}(1-\epsilon_j)^{a}\epsilon_j^{t+1-a}\right)\left(\prod_{l\in H/B}(1-\epsilon_l)^b\epsilon_l^t\right)\right] \end{aligned} \tag{51}$$

$$\begin{aligned} O_{W'\to B} =& \prod_{i\in B/W'}(1-\epsilon_i)\prod_{j\in W'}\epsilon_j\prod_{l\in H/B}((1-\epsilon_l)+\epsilon_l) \\ &\left[\sum_{a=2}^{t}\sum_{b=0}^{a-2}\left(\prod_{i\in B}(1-\epsilon_i)^a\epsilon_i^{t-a}\right)\left(\prod_{j\in W'/B}(1-\epsilon_j)^{a-1}\epsilon_j^{t+1-a}\right)\left(\prod_{l\in H/B}(1-\epsilon_l)^b\epsilon_l^t\right)\right] \\ &+\prod_{i\in B/W'}(1-\epsilon_i)\prod_{j\in W'}\epsilon_j\prod_{l\in H/B}\epsilon_l \\ &\left[\sum_{a=1}^{t}\sum_{b=0}^{a-1}\left(\prod_{i\in B}(1-\epsilon_i)^a\epsilon_i^{t-a}\right)\left(\prod_{j\in W'/B}(1-\epsilon_j)^{a-1}\epsilon_j^{t+1-a}\right)\left(\prod_{l\in H/W}(1-\epsilon_l)^b\epsilon_l^t\right)\right] \\ =& \prod_{l\in H/B}((1-\epsilon_l)+\epsilon_l)\left[\sum_{a=2}^{t}\sum_{b=0}^{a-2}\left(\prod_{i\in B}(1-\epsilon_i)^a\epsilon_i^{t+1-a}\right)\left(\prod_{j\in W'/B}(1-\epsilon_j)^{a}\epsilon_j^{t+1-a}\right)\left(\prod_{l\in H/W}(1-\epsilon_l)^(b)\epsilon_l^t\right)\right] \\ &+\prod_{l\in H/B}\epsilon_l\left[\sum_{a=1}^{t}\sum_{b=0}^{a-1}\left(\prod_{i\in W}(1-\epsilon_i)^a\epsilon_i^{t+1-a}\right)\left(\prod_{j\in W'/B}(1-\epsilon_j)^{a}\epsilon_j^{t+1-a}\right)\left(\prod_{l\in H/B}(1-\epsilon_l)^(b)\epsilon_l^t\right)\right] \end{aligned} \tag{52}$$

From these definitions, it can be directly seen that the two terms are equal.

F. Derivation of Equation 19

A. Derivation for $|H| = 3$

First the derivation is shown for $|H| = 3$. This is a warm up for the general case, $|H| = k$ for any positive integer k.

$$\begin{aligned}E_t =& p^t_{[1]}\epsilon_{[1]}\\ &+p^t_{[2]}\epsilon_{[2]}\\ &+p^t_{[3]}\epsilon_{[3]}\\ &+p^t_{[1,2]}(\frac{1}{2}\epsilon_{[1]} + \frac{1}{2}\epsilon_{[2]})\\ &+p^t_{[1,3]}(\frac{1}{2}\epsilon_{[1]} + \frac{1}{2}\epsilon_{[3]})\\ &+p^t_{[2,3]}(\frac{1}{2}\epsilon_{[2]} + \frac{1}{2}\epsilon_{[3]})\\ &+p^t_{[1,2,3]}(\frac{1}{3}\epsilon_{[1]} + \frac{1}{3}\epsilon_{[2]} + \frac{1}{3}\epsilon_{[3]})\end{aligned} \tag{53}$$

We can derive a similar expression for E_{t+1}. From previous proofs we know that p^{t+1}_B can be expressed as a function of p^t_B with the help of inflow and outflow. This leads to the next expression for E_{t+1}.

$$\begin{aligned}E_{t+1} =& (p^t_{[1]} + I_{[1,2]\to[1]} - O_{[1]\to[1,2]} + I_{[1,3]\to[1]} - O_{[1]\to[1,3]} + I_{[1,2,3]\to[1]} - O_{[1]\to[1,2,3]})\epsilon_{[1]}\\ &+(p^t_{[2]} + I_{[1,2]\to[2]} - O_{[2]\to[1,2]} + I_{[2,3]\to[2]} - O_{[2]\to[2,3]} + I_{[1,2,3]\to[2]} - O_{[2]\to[1,2,3]})\epsilon_{[2]}\\ &+(p^t_{[3]} + I_{[1,3]\to[3]} - O_{[3]\to[1,3]} + I_{[2,3]\to[3]} - O_{[3]\to[2,3]} + I_{[1,2,3]\to[3]} - O_{[3]\to[1,2,3]})\epsilon_{[3]}\\ &+(p^t_{[1,2]} + O_{[1]\to[1,2]} - I_{[1,2]\to[1]} + O_{[2]\to[1,2]} - I_{[1,2]\to[2]} + I_{[1,2,3]\to[1,2]} - O_{[1,2]\to[1,2,3]})(\frac{1}{2}\epsilon_{[1]} + \frac{1}{2}\epsilon_{[2]})\\ &+(p^t_{[1,3]} + O_{[1]\to[1,3]} - I_{[1,3]\to[1]} + O_{[3]\to[1,3]} - I_{[1,3]\to[3]} + I_{[1,2,3]\to[1,3]} - O_{[1,3]\to[1,2,3]})(\frac{1}{2}\epsilon_{[1]} + \frac{1}{2}\epsilon_{[3]})\\ &+(p^t_{[2,3]} + O_{[2]\to[2,3]} - I_{[2,3]\to[2]} + O_{[3]\to[2,3]} - I_{[2,3]\to[3]} + I_{[1,2,3]\to[2,3]} - O_{[2,3]\to[1,2,3]})(\frac{1}{2}\epsilon_{[2]} + \frac{1}{2}\epsilon_{[3]})\\ &+(p^t_{[1,2,3]} + O_{[1]\to[1,2,3]} - I_{[1,2,3]\to[1]} + O_{[2]\to[1,2,3]} - I_{[1,2,3]\to[2]} + O_{[3]\to[1,2,3]} - I_{[1,2,3]\to[3]}+\\ &O_{[1,2]\to[1,2,3]} - I_{[1,2,3]\to[1,2]} + O_{[1,3]\to[1,2,3]} - I_{[1,2,3]\to[1,3]} + O_{[2,3]\to[1,2,3]} - I_{[1,2,3]\to[2,3]})\\ &(\frac{1}{3}\epsilon_{[1]} + \frac{1}{3}\epsilon_{[2]} + \frac{1}{3}\epsilon_{[3]})\end{aligned} \tag{54}$$

We can then write $E_t - E_{t+1}$ as follows.

$$
\begin{aligned}
E_t - E_{t+1} =& (O_{[1]\to[1,2]} - I_{[1,2]\to[1]} + O_{[1]\to[1,3]} - I_{[1,3]\to[1]} + O_{[1]\to[1,2,3]} - I_{[1,2,3]\to[1]})\epsilon_{[1]} \\
&+(O_{[2]\to[1,2]} - I_{[1,2]\to[2]} + O_{[2]\to[2,3]} - I_{[2,3]\to[2]} + O_{[2]\to[1,2,3]} - I_{[1,2,3]\to[2]})\epsilon_{[2]} \\
&+(O_{[3]\to[1,3]} - I_{[1,3]\to[3]} + O_{[3]\to[2,3]} - I_{[2,3]\to[3]} + O_{[3]\to[1,2,3]} - I_{[1,2,3]\to[3]})\epsilon_{[3]} \\
&+(I_{[1,2]\to[1]} - O_{[1]\to[1,2]} + I_{[1,2]\to[2]} - O_{[2]\to[1,2]} + O_{[1,2]\to[1,2,3]} - I_{[1,2,3]\to[1,2]})(\frac{1}{2}\epsilon_{[1]} + \frac{1}{2}\epsilon_{[2]}) \\
&+(I_{[1,3]\to[1]} - O_{[1]\to[1,3]} + I_{[1,3]\to[3]} - O_{[3]\to[1,3]} + O_{[1,3]\to[1,2,3]} - I_{[1,2,3]\to[1,3]})(\frac{1}{2}\epsilon_{[1]} + \frac{1}{2}\epsilon_{[3]}) \\
&+(I_{[2,3]\to[2]} - O_{[2]\to[2,3]} + I_{[2,3]\to[3]} - O_{[3]\to[2,3]} + O_{[2,3]\to[1,2,3]} - I_{[1,2,3]\to[2,3]})(\frac{1}{2}\epsilon_{[2]} + \frac{1}{2}\epsilon_{[3]}) \\
&+(I_{[1,2,3]\to[1]} - O_{[1]\to[1,2,3]} + I_{[1,2,3]\to[2]} - O_{[2]\to[1,2,3]} + I_{[1,2,3]\to[3]} - O_{[3]\to[1,2,3]} + \\
&\quad I_{[1,2,3]\to[1,2]} - O_{[1,2]\to[1,2,3]} + I_{[1,2,3]\to[1,3]} - O_{[1,3]\to[1,2,3]} + I_{[1,2,3]\to[2,3]} - O_{[2,3]\to[1,2,3]}) \\
&\quad (\frac{1}{3}\epsilon_{[1]} + \frac{1}{3}\epsilon_{[2]} + \frac{1}{3}\epsilon_{[3]})
\end{aligned}
\tag{55}
$$

We group all terms by ϵ.

$$\begin{aligned}E_t - E_{t+1} =&(\frac{-1}{2}I_{[1,2,3]\rightarrow[1,2]} + \frac{1}{3}I_{[1,2,3]\rightarrow[1,2]} + \frac{-1}{2}I_{[1,2,3]\rightarrow[1,3]} + \frac{1}{3}I_{[1,2,3]\rightarrow[1,3]} + \frac{-1}{1}I_{[1,2,3]\rightarrow[1]} + \frac{1}{3}I_{[1,2,3]\rightarrow[1]}+\\
&\frac{1}{3}I_{[1,2,3]\rightarrow[2,3]} + \frac{1}{3}I_{[1,2,3]\rightarrow[2]} + \frac{1}{3}I_{[1,2,3]\rightarrow[3]} + \frac{-1}{1}I_{[1,2]\rightarrow[1]} + \frac{1}{2}I_{[1,2]\rightarrow[1]} + \frac{1}{2}I_{[1,2]\rightarrow[2]}+\\
&\frac{-1}{1}I_{[1,3]\rightarrow[1]} + \frac{1}{2}I_{[1,3]\rightarrow[1]} + \frac{1}{2}I_{[1,3]\rightarrow[3]} + \frac{1}{2}O_{[1,2]\rightarrow[1,2,3]} + \frac{-1}{3}O_{[1,2]\rightarrow[1,2,3]} + \frac{1}{2}O_{[1,3]\rightarrow[1,2,3]}+\\
&\frac{-1}{3}O_{[1,3]\rightarrow[1,2,3]} + \frac{1}{1}O_{[1]\rightarrow[1,2,3]} + \frac{-1}{3}O_{[1]\rightarrow[1,2,3]} + \frac{1}{1}O_{[1]\rightarrow[1,2]} + \frac{-1}{2}O_{[1]\rightarrow[1,2]} + \frac{1}{1}O_{[1]\rightarrow[1,3]}+\\
&\frac{-1}{2}O_{[1]\rightarrow[1,3]} + \frac{-1}{3}O_{[2,3]\rightarrow[1,2,3]} + \frac{-1}{3}O_{[2]\rightarrow[1,2,3]} + \frac{-1}{2}O_{[2]\rightarrow[1,2]} + \frac{-1}{3}O_{[3]\rightarrow[1,2,3]} + \frac{-1}{2}O_{[3]\rightarrow[1,3]})\epsilon_1\\
&+(\frac{-1}{2}I_{[1,2,3]\rightarrow[1,2]} + \frac{1}{3}I_{[1,2,3]\rightarrow[1,2]} + \frac{1}{3}I_{[1,2,3]\rightarrow[1,3]} + \frac{1}{3}I_{[1,2,3]\rightarrow[1]} + \frac{-1}{2}I_{[1,2,3]\rightarrow[2,3]} + \frac{1}{3}I_{[1,2,3]\rightarrow[2,3]}+\\
&\frac{-1}{1}I_{[1,2,3]\rightarrow[2]} + \frac{1}{3}I_{[1,2,3]\rightarrow[2]} + \frac{1}{3}I_{[1,2,3]\rightarrow[3]} + \frac{1}{2}I_{[1,2]\rightarrow[1]} + \frac{-1}{1}I_{[1,2]\rightarrow[2]} + \frac{1}{2}I_{[1,2]\rightarrow[2]}+\\
&\frac{-1}{1}I_{[2,3]\rightarrow[2]} + \frac{1}{2}I_{[2,3]\rightarrow[2]} + \frac{1}{2}I_{[2,3]\rightarrow[3]} + \frac{1}{2}O_{[1,2]\rightarrow[1,2,3]} + \frac{-1}{3}O_{[1,2]\rightarrow[1,2,3]} + \frac{-1}{3}O_{[1,3]\rightarrow[1,2,3]}+\\
&\frac{-1}{3}O_{[1]\rightarrow[1,2,3]} + \frac{-1}{2}O_{[1]\rightarrow[1,2]} + \frac{1}{2}O_{[2,3]\rightarrow[1,2,3]} + \frac{-1}{3}O_{[2,3]\rightarrow[1,2,3]} + \frac{1}{1}O_{[2]\rightarrow[1,2,3]} + \frac{-1}{3}O_{[2]\rightarrow[1,2,3]}+\\
&\frac{1}{1}O_{[2]\rightarrow[1,2]} + \frac{-1}{2}O_{[2]\rightarrow[1,2]} + \frac{1}{1}O_{[2]\rightarrow[2,3]} + \frac{-1}{2}O_{[2]\rightarrow[2,3]} + \frac{-1}{3}O_{[3]\rightarrow[1,2,3]} + \frac{-1}{2}O_{[3]\rightarrow[2,3]})\epsilon_2\\
&+(\frac{1}{3}I_{[1,2,3]\rightarrow[1,2]} + \frac{-1}{2}I_{[1,2,3]\rightarrow[1,3]} + \frac{1}{3}I_{[1,2,3]\rightarrow[1,3]} + \frac{1}{3}I_{[1,2,3]\rightarrow[1]} + \frac{-1}{2}I_{[1,2,3]\rightarrow[2,3]} + \frac{1}{3}I_{[1,2,3]\rightarrow[2,3]}+\\
&\frac{1}{3}I_{[1,2,3]\rightarrow[2]} + \frac{-1}{1}I_{[1,2,3]\rightarrow[3]} + \frac{1}{3}I_{[1,2,3]\rightarrow[3]} + \frac{1}{2}I_{[1,3]\rightarrow[1]} + \frac{-1}{1}I_{[1,3]\rightarrow[3]} + \frac{1}{2}I_{[1,3]\rightarrow[3]}+\\
&\frac{1}{2}I_{[2,3]\rightarrow[2]} + \frac{-1}{1}I_{[2,3]\rightarrow[3]} + \frac{1}{2}I_{[2,3]\rightarrow[3]} + \frac{-1}{3}O_{[1,2]\rightarrow[1,2,3]} + \frac{1}{2}O_{[1,3]\rightarrow[1,2,3]} + \frac{-1}{3}O_{[1,3]\rightarrow[1,2,3]}+\\
&\frac{-1}{3}O_{[1]\rightarrow[1,2,3]} + \frac{-1}{2}O_{[1]\rightarrow[1,3]} + \frac{1}{2}O_{[2,3]\rightarrow[1,2,3]} + \frac{-1}{3}O_{[2,3]\rightarrow[1,2,3]} + \frac{-1}{3}O_{[2]\rightarrow[1,2,3]} + \frac{-1}{2}O_{[2]\rightarrow[2,3]}+\\
&\frac{1}{1}O_{[3]\rightarrow[1,2,3]} + \frac{-1}{3}O_{[3]\rightarrow[1,2,3]} + \frac{1}{1}O_{[3]\rightarrow[1,3]} + \frac{-1}{2}O_{[3]\rightarrow[1,3]} + \frac{1}{1}O_{[3]\rightarrow[2,3]} + \frac{-1}{2}O_{[3]\rightarrow[2,3]})\epsilon_3\end{aligned}\tag{56}$$

We assume Lemma 1 holds. The expression becomes the following after eliminating all outflow terms.

$$\begin{aligned}E_t - E_{t+1} =&(\frac{-1}{6}I_{[1,2,3]\rightarrow[1,2]} + \frac{-1}{6}I_{[1,2,3]\rightarrow[1,3]} + \frac{-2}{3}I_{[1,2,3]\rightarrow[1]} + \frac{1}{3}I_{[1,2,3]\rightarrow[2,3]} + \frac{1}{3}I_{[1,2,3]\rightarrow[2]} + \frac{1}{3}I_{[1,2,3]\rightarrow[3]}+\\
&\frac{-1}{2}I_{[1,2]\rightarrow[1]} + \frac{1}{2}I_{[1,2]\rightarrow[2]} + \frac{-1}{2}I_{[1,3]\rightarrow[1]} + \frac{1}{2}I_{[1,3]\rightarrow[3]})\epsilon_1\\
&+(\frac{-1}{6}I_{[1,2,3]\rightarrow[1,2]} + \frac{1}{3}I_{[1,2,3]\rightarrow[1,3]} + \frac{1}{3}I_{[1,2,3]\rightarrow[1]} + \frac{-1}{6}I_{[1,2,3]\rightarrow[2,3]} + \frac{-2}{3}I_{[1,2,3]\rightarrow[2]} + \frac{1}{3}I_{[1,2,3]\rightarrow[3]}+\\
&\frac{1}{2}I_{[1,2]\rightarrow[1]} + \frac{-1}{2}I_{[1,2]\rightarrow[2]} + \frac{-1}{2}I_{[2,3]\rightarrow[2]} + \frac{1}{2}I_{[2,3]\rightarrow[3]})\epsilon_2\\
&+(\frac{1}{3}I_{[1,2,3]\rightarrow[1,2]} + \frac{-1}{6}I_{[1,2,3]\rightarrow[1,3]} + \frac{1}{3}I_{[1,2,3]\rightarrow[1]} + \frac{-1}{6}I_{[1,2,3]\rightarrow[2,3]} + \frac{1}{3}I_{[1,2,3]\rightarrow[2]} + \frac{-2}{3}I_{[1,2,3]\rightarrow[3]}+\\
&\frac{1}{2}I_{[1,3]\rightarrow[1]} + \frac{-1}{2}I_{[1,3]\rightarrow[3]} + \frac{1}{2}I_{[2,3]\rightarrow[2]} + \frac{-1}{2}I_{[2,3]\rightarrow[3]})\epsilon_3\end{aligned}\tag{57}$$

We can group like terms together to obtain.

$$\begin{aligned} E_t - E_{t+1} =& I_{[1,2]\to[1]}(\frac{-1}{2}\epsilon_1 + \frac{1}{2}\epsilon_2) \\ &+ I_{[1,2]\to[2]}(\frac{1}{2}\epsilon_1 + \frac{-1}{2}\epsilon_2) \\ &+ I_{[1,3]\to[1]}(\frac{-1}{2}\epsilon_1 + \frac{1}{2}\epsilon_3) \\ &+ I_{[1,3]\to[3]}(\frac{1}{2}\epsilon_1 + \frac{-1}{2}\epsilon_3) \\ &+ I_{[2,3]\to[2]}(\frac{-1}{2}\epsilon_2 + \frac{1}{2}\epsilon_3) \\ &+ I_{[2,3]\to[3]}(\frac{1}{2}\epsilon_2 + \frac{-1}{2}\epsilon_3) \\ &+ I_{[1,2,3]\to[1]}(\frac{-2}{3}\epsilon_1 + \frac{1}{3}\epsilon_2 + \frac{1}{3}\epsilon_3) \\ &+ I_{[1,2,3]\to[2]}(\frac{1}{3}\epsilon_1 + \frac{-2}{3}\epsilon_2 + \frac{1}{3}\epsilon_3) \\ &+ I_{[1,2,3]\to[3]}(\frac{1}{3}\epsilon_1 + \frac{1}{3}\epsilon_2 + \frac{-2}{3}\epsilon_3) \\ &+ I_{[1,2,3]\to[1,2]}(\frac{-1}{6}\epsilon_1 + \frac{-1}{6}\epsilon_2 + \frac{1}{3}\epsilon_3) \\ &+ I_{[1,2,3]\to[1,3]}(\frac{-1}{6}\epsilon_1 + \frac{1}{3}\epsilon_2 + \frac{-1}{6}\epsilon_3) \\ &+ I_{[1,2,3]\to[2,3]}(\frac{1}{3}\epsilon_1 + \frac{-1}{6}\epsilon_2 + \frac{-1}{6}\epsilon_3) \end{aligned} \tag{58}$$

This is an expression of the form of Equation 19.

B. Derivation for $|H| = k$

Note that this sections follows the same steps as for the derivation given for $|H| = 3$. V is a subset of hypotheses of H, so $V \subseteq W$. For ease of notation, we do not consider the emptyset $\emptyset$ to be part of any subset $V \subseteq W$

$$E_t - E_{t+1} = \sum_{V \subseteq H} (p_V^t - p_V^{t+1})(\frac{1}{|V|} \sum_{h_x \in V} \epsilon_x) \geq 0 \tag{59}$$

Substitute p_V^{t+1} with inflow and outflow. Remember that another subset $W \subseteq H$ with $W \subset V$, W receives inflow from V, and loses outflow to V. Similarly if $V \subset W$, W loses inflow to V, and gains outflow from V.

$$
\begin{aligned}
E_t - E_{t+1} &= \sum_{V\subseteq H} (p_V^t - \left(p_V^t + \sum_{W\subset V}(O_{W\to V} - I_{V\to W}) + \sum_{W\subseteq H:V\subset W}(I_{W\to V} - O_{V\to W})\right)(\frac{1}{|V|}\sum_{h_x\in V}\epsilon_x)\\
&= \sum_{V\subseteq H}\left(\sum_{W\subset V} I_{V\to W} - \sum_{W\subset V} O_{W\to V} - \sum_{W\subseteq H:V\subset W} I_{W\to V} + \sum_{W\subseteq H:V\subset W} O_{V\to W}\right)(\frac{1}{|V|}\sum_{h_x\in V}\epsilon_x)\\
&= \sum_{V\subseteq H}\sum_{h_x\in V}\epsilon_x\left(\sum_{W\subset V}\frac{1}{|V|}I_{V\to W} - \sum_{W\subset V}\frac{1}{|V|}O_{W\to V} - \sum_{W\subseteq H:V\subset W}\frac{1}{|V|}I_{W\to V} + \sum_{W\subseteq H:V\subset W}\frac{1}{|V|}O_{V\to W})\right)\\
&\geq 0
\end{aligned}
\tag{60}
$$

We group all variables by value of ϵ_i. Refer to Appendix F.A if this step is unclear. With $|H| = k$:

$$
\begin{aligned}
E_t - E_{t+1} = \sum_{h_x\in H}\epsilon_x\sum_{i=2}^{k}\sum_{V=[H]^i:h_i\in V}&\left(\sum_{W\subset V:h_i\in W}(\frac{1}{|W|}O_{W\to V}) - \sum_{W'\subset V}(\frac{1}{|V|}O_{W'\to V})\right.\\
&\left.- \sum_{W\subset V:h_i\in W}(\frac{1}{|W|}I_{V\to W}) + \sum_{W'\subset V}(\frac{1}{|V|}I_{V\to W'})\right)\\
= \sum_{h_x\in H}\epsilon_x\sum_{i=2}^{k}\sum_{V=[H]^i:h_i\in V}\sum_{j=1}^{i-1}&\left(\sum_{W=[V]^j:h_i\in W}(\frac{1}{j}O_{W\to V}) - \sum_{W'=[V]^j}(\frac{1}{i}O_{W'\to V})\right.\\
&\left.- \sum_{W=[V]^j:h_i\in W}(\frac{1}{j}I_{V\to W}) + \sum_{W'=[V]^j}(\frac{1}{i}I_{V\to W'})\right)
\end{aligned}
\tag{61}
$$

For any given set $V \subseteq H$ of size i and positive integer $1 < j < i$, there are $\binom{i}{j}$ subsets $W \subset V$ of size j. For any given hypothesis $h_x \in V$, there are $\binom{i-1}{j-1}$ subsets $W \subset V$ of size j containing h_x, so $h_x \in W$. Remember that any two outflows $O_{W\to V}$ and $O_{W'\to V}$ are equal by Lemma 1 if $|W| = |W'|$. Using $O_{W^j\to V}$ as any outflow term where $|W^j| = j$, we can now write:

$$
\begin{aligned}
E_t - E_{t+1} = \sum_{h_x\in H}\epsilon_x\sum_{i=2}^{k}\sum_{V=[H]^i:h_x\in V}\sum_{j=1}^{i-1}&\left(\binom{i-1}{j-1}\frac{1}{j}O_{W^j\to V} - \binom{i}{j}\frac{1}{i}O_{W^j\to V}\right.\\
&\left.- \sum_{W=[V]^j:h_x\in W}(\frac{1}{j}I_{V\to W}) + \sum_{W'=[V]^j}(\frac{1}{i}I_{V\to W'})\right)
\end{aligned}
\tag{62}
$$

Using the fact that $\binom{i}{j}\frac{1}{i} = \binom{i-1}{j-1}\frac{1}{j}$ all outflow terms can now be eliminated. We write:

$$
E_t - E_{t+1} = \sum_{h_x\in H}\epsilon_x\sum_{i=2}^{k}\sum_{V=[H]^i:h_x\in V}\sum_{j=1}^{i-1}\left(-\sum_{W=[V]^j:h_x\in W}(\frac{1}{j}I_{V\to W}) + \sum_{W'=[V]^j}(\frac{1}{i}I_{V\to W'})\right)
\tag{63}
$$

The term $\sum_{W'=[V]^j}(\frac{1}{i}I_{V\to W'})$ is rearranged, yielding:

$$
E_t - E_{t+1} = \sum_{h_x\in H}\epsilon_x\sum_{i=2}^{k}\sum_{V=[H]^i}\sum_{j=1}^{i-1}\left(\sum_{W=[V]^j:h_x\in W} -\frac{i-j}{ij}I_{V\to W} + \sum_{W'=[V]^j:h_x\notin W'}\frac{1}{i}I_{V\to W'}\right)
\tag{64}
$$

Grouping by inflow terms instead of ϵ finally yields:

$$\begin{aligned}E_t - E_{t+1} =& \sum_{i=2}^{k} \sum_{V=[H]^i} \sum_{j=1}^{i-1} \sum_{W=[V]^j} I_{V\to W} \left((\sum_{h_y \in W} -\frac{i-j}{ij}\epsilon_y) + (\sum_{h_x \in V/W} \frac{1}{i}\epsilon_x) \right) \\ \geq& 0 \end{aligned} \tag{65}$$

G. Derivation of Equation 23

First, the following equation is given.

$$\begin{aligned}E_t - E_{t+1} =& \sum_{i=2}^{k} \sum_{V=[H]^i} \sum_{j=1}^{i-1} \sum_{W=[V]^j} I_{V\to W} \left((\sum_{h_x \in V/W} \frac{1}{i}\epsilon_x) + (\sum_{h_y \in W} -\frac{i-j}{ij}\epsilon_y) \right) \\ =& \sum_{i=2}^{k} \sum_{V=[H]^i} \sum_{j=1}^{i-1} \sum_{W=[V]^j} I_{V\to W} \left((\sum_{h_x \in V/W} \frac{j}{ij}\epsilon_x) + (\sum_{h_y \in W} -\frac{i-j}{ij}\epsilon_y) \right) \end{aligned} \tag{66}$$

We can make pairs of any two hypotheses $h_x \in V/W$ and $h_y \in W$ and their respective ϵ values as follows:

$$E_t - E_{t+1} = \sum_{i=2}^{k} \sum_{V=[H]^i} \sum_{j=1}^{i-1} \sum_{W=[V]^j} \sum_{h_x \in V/W} \sum_{h_y \in W} I_{V\to W} \left(\frac{1}{ij} h_x - \frac{1}{ij} h_y \right) \tag{67}$$

Note that it is possible to make pairs with factor $\frac{1}{ij}$ because there are j elements in the set W, so $\frac{1}{j}\frac{j}{ij} = \frac{1}{ij}$. There are also $i - j$ elements in the set V/W, so $\frac{1}{i-j}\frac{i-j}{ij} = \frac{1}{ij}$.

We use the information that we may express $E_t - E_{t+1}$ for any V and W and any two pairs of $h_x \in V/W$ and $h_y \in W$ as a sum of all $I_{V\to W}(\frac{1}{ij}h_x - \frac{1}{ij}h_y)$. We mirror W with another subset of V, $W' \subset V$, given that $|W \cap W'| = |W| - 1 = |W'| - 1$. If we match unique pairs of W and W' such that $h_x = W'/W$ and $h_y = W/W'$, we may rewrite our expression a final time.

$$E_t - E_{t+1} = \sum_{i=2}^{k} \sum_{V=[H]^i} \sum_{j=1}^{i-1} \sum_{\{W,W'\} \in \{[V]^j\}_*^2} I_{V\to W}(\frac{1}{ij}h_x - \frac{1}{ij}h_y) + I_{V\to W'}(\frac{1}{ij}h_y - \frac{1}{ij}h_x) \tag{68}$$

Here $\{[V]^j\}_*^2$ denotes the set of all pairs of unique subsets of V size j, so $|W| = |W'| = j$ that overlap on all elements except one in each subset so $|W/W'| = |W'/W| = 1$

H. Proof of monotonicity for $|H| = 2$, dependent

We first need to redefine Table 1. As we no longer assume hypotheses classify independently, so we need to define the joint probabilities of classification outcomes of h_1 and h_2.

Table 2. Definition ρ and n for $|H| = 2$ There are four possible classifications of a single instance $(x, y) \in S_t$. The first two columns indicate whether a hypothesis classifies the instance correct 1 or incorrect 0. The third column denotes the associated probability of that classification. For example, the probability that both h_1 and h_2 classify an instance correctly is equal to ρ_{11}, row 4. The 4th column denotes variable associated with the amount of instances of a sample S_t that are classified according to column one and two, with $t = n_{00} + n_{10} + n_{01} + n_{11}$. So for a sample S_t for which both hypotheses classify all instances correctly, $d = t$. The probability of this outcome is $\rho_{11}^{n_{11}}$

h_1	h_2	probability of classification	associated instance count
0	0	ρ_{00}	n_{00}
1	0	ρ_{10}	n_{10}
0	1	ρ_{01}	n_{01}
1	1	ρ_{11}	n_{11}

We need to redefine p_B^t for the case hypotheses do not necessarily classify independently.

$$P_t(n_{10}, n_{01}) = \sum_{n_{00}=0}^{t-n_{10}-n_{01}} \binom{t}{n_{00}} \rho_{00}^{n_{00}} \binom{t-n_{00}}{n_{10}} \rho_{10}^{n_{10}} \binom{t-n_{00}-n_{10}}{n_{01}} \rho_{01}^{n_{01}} \rho_{11}^{t-n_{00}-n_{10}-n_{01}} \tag{69}$$

With the definition of a classification outcome (n_1, n_2) and its associated probability $P_t(n_1, n_2)$, p_B^t can now be defined. Note that $p_{[1]}^t$ - the probability that h_1 has the lowest error rate on a sample S_t - is simply the sum of all probabilities of classification outcomes for which $n_1 < n_2$. This allows us to write $p_{[1]}^t$ (and $p_{[2]}^t$ and $p_{[1,2]}^t$) as follows:

$$\begin{aligned} p_{[1]}^t &= \sum_{n_{10}=1}^{t} \sum_{n_{01}=0}^{min(n_{10}-1, t-n_{10})} P_t(n_{10}, n_{01}) \\ p_{[2]}^t &= \sum_{n_{01}=1}^{t} \sum_{n_{10}=0}^{min(n_{01}-1, t-n_{01})} P_t(n_{10}, n_{01}) \\ p_{[1,2]}^t &= \sum_{n_{10}=0}^{\lfloor t/2 \rfloor} P_t(n_{10}, n_{01}) \qquad \text{(with } n_{10} = n_{01}) \end{aligned} \tag{70}$$

Next we define $p_{[1]}^{t+1}$ as

$$\begin{aligned} p_{[1]}^{t+1} &= p_{[1]}^t + I_{[1,2]\to[1]} - O_{[1]\to[1,2]} \\ p_{[2]}^{t+1} &= p_{[2]}^t + I_{[1,2]\to[2]} - O_{[2]\to[1,2]} \\ p_{[1,2]}^{t+1} &= p_{[1,2]}^t - I_{[1,2]\to[1]} + O_{[1]\to[1,2]} - I_{[1,2]\to[2]} + O_{[2]\to[1,2]} \end{aligned} \tag{71}$$

Note that this is the exact same as presented in Subsection 11. Hence, if we proof Lemma 1 and 2, we proof monotonicity for the dependent case.

A. Inflow Proof 2 Hypotheses
To prove

$$\rho_{10} > \rho_{01} \implies I_{[1,2]\to[1]} - I_{[1,2]\to[2]} > 0 \tag{72}$$

$$\rho_{10} < \rho_{01} \implies I_{[1,2]\to[1]} - I_{[1,2]\to[2]} < 0 \tag{73}$$

Proof We define both I_{p_1} and I_{p_2} :

$$I_{[1,2]\to[1]} = \rho_{10} p^t_{[1,2]} \tag{74}$$

$$I_{[1,2]\to[2]} = \rho_{01} p^t_{[1,2]} \tag{75}$$

With these definitions it is immediately apparent that $\rho_{10} > \rho_{01}$ implies $I_{p_1} > I_{p_2}$, and similarly $\rho_{10} < \rho_{01}$ implies $I_{p_1} < I_{p_2}$.

B. Outflow Proof 2 Hypotheses
Using Table 2, we can write out $O_{[1]\to[1,2]}$ and $O_{[2]\to[1,2]}$ for sample S_t.

$$\begin{aligned}
O_{[1]\to[1,2]} &= \rho_{01} \sum_{n_{10}=1}^{\lceil t/2 \rceil} P(n_{10}, n_{01}) && \text{(with } n_{10} = n_{01} + 1\text{)} \\
&= \rho_{01} \sum_{n_{10}=1}^{\lceil t/2 \rceil} \sum_{n_{00}=0}^{t-n_{10}-n_{01}} \binom{t}{n_{10}} \binom{t-n_{10}}{n_{01}} \binom{t-n_{10}-n_{01}}{n_{00}} \rho_{00}^{n_{00}} \rho_{10}^{n_{10}} \rho_{01}^{n_{01}} \rho_{11}^{t-n_{00}-n_{01}-n_{10}}
\end{aligned} \tag{76}$$

$$\begin{aligned}
O_{[1]\to[1,2]} &= \rho_{10} \sum_{n_{01}=1}^{\lceil t/2 \rceil} P(n_{10}, n_{01}) && \text{(with } n_{01} = n_{10} + 1\text{)} \\
&= \rho_{01} \sum_{n_{01}=1}^{\lceil t/2 \rceil} \sum_{n_{00}=0}^{t-n_{10}-n_{01}} \binom{t}{n_{10}} \binom{t-n_{10}}{n_{01}} \binom{t-n_{10}-n_{01}}{n_{00}} \rho_{00}^{n_{00}} \rho_{10}^{n_{10}} \rho_{01}^{n_{01}} \rho_{11}^{t-n_{00}-n_{01}-n_{10}}
\end{aligned} \tag{77}$$

With these definitions, a direct derivation is possible:

$$\begin{aligned}
O_{[1]\to[1,2]} &= \rho_{01} \sum_{n_{10}=1}^{\lceil t/2 \rceil} \sum_{n_{00}=0}^{t-n_{10}-n_{01}} \binom{t}{n_{10}} \binom{t-n_{10}}{n_{01}} \binom{t-n_{10}-n_{01}}{n_{00}} \rho_{00}^{n_{00}} \rho_{10}^{n_{10}} \rho_{01}^{n_{01}} \rho_{11}^{t-n_{00}-n_{01}-n_{10}} && \text{(with } n_{10} = n_{01} + 1\text{)} \\
&= \sum_{n_{10}=1}^{\lceil t/2 \rceil} \sum_{n_{00}=0}^{t-n_{10}-n_{01}} \binom{t}{n_{10}} \binom{t-n_{10}}{n_{01}} \binom{t-n_{10}-n_{01}}{n_{00}} \rho_{00}^{n_{00}} \rho_{10}^{n_{10}} \rho_{01}^{n_{01}} \rho_{11}^{t-n_{00}-n_{01}-n_{10}} && \text{(with } n_{10} = n_{01}\text{)} \\
&= \rho_{01} \sum_{n_{10}=1}^{\lceil t/2 \rceil} \sum_{n_{00}=0}^{t-n_{10}-n_{01}} \binom{t}{n_{10}} \binom{t-n_{10}}{n_{01}} \binom{t-n_{10}-n_{01}}{n_{00}} \rho_{00}^{n_{00}} \rho_{10}^{n_{10}} \rho_{01}^{n_{01}} \rho_{11}^{t-n_{00}-n_{01}-n_{10}} && \text{(with } n_{01} = n_{10} + 1\text{)} \\
&= O_{[1]\to[1,2]}
\end{aligned} \tag{78}$$

With these proofs of Lemma 1 and 2, we proof monotonicity by Equation 23.

I. Exponential Fitting

For $|H| = k$, we plot E_t for different t, k and ϵ values. Simply choose and values and fill in in Equation 5, repeated below.

$$E_t = \sum_{B \subseteq H} p_B^t \sum_{h_i \in B} \frac{1}{|B|} \epsilon_i \quad \text{(with } B \neq \emptyset) \tag{79}$$

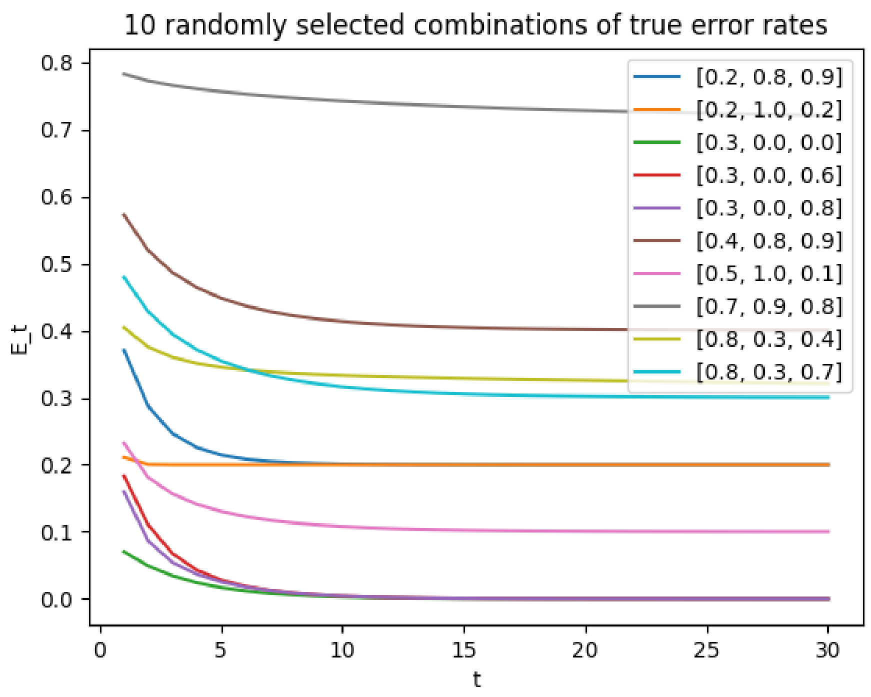
with

$$p_B^t = \sum_{a \in A} P_t(a) \qquad \text{with}$$
$$A = \{\{n_1, n_2, ..., n_k\} \in \mathbb{Z}^k_{0 \leq t} | \forall n_i, n_j \in \{n_1, n_2, ..., n_k\} : \begin{cases} n_i > n_j, & \text{if } h_i \in B, h_j \notin B \\ n_i = n_j, & \text{if } h_i, h_j \in B \end{cases} \tag{80}$$

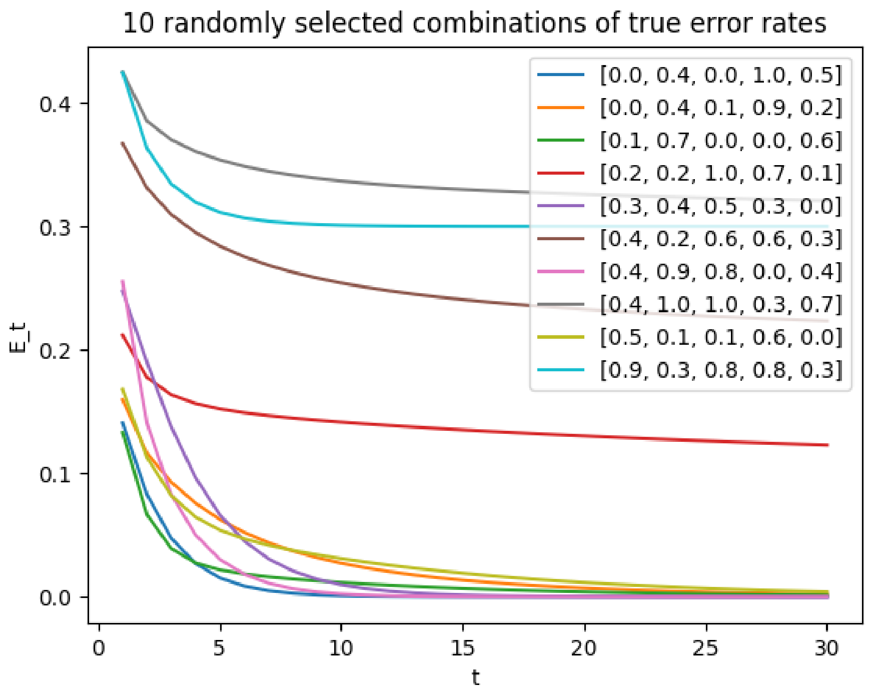
and

$$P_t(a) = \prod_{i=1}^{k} \binom{t}{i} (1 - \epsilon_i)^{n_i} \epsilon_i^{t - n_i} \quad \text{with } a = \{n_1, n_2, ...n_k\} \tag{81}$$

For $|H| = 3$ is plotted for uniform random combinations of values of true error rates ϵ_1, ϵ_2 and ϵ_3 in the range $\epsilon = (0, 1)$, for the range $t = [1, 30]$. The legend contains the randomly selected epsilon values in order, so $\epsilon_1 = 0.2$, $\epsilon_2 = 0.3$ and $\epsilon_3 = 0.8$ is listed as $[0.2, 0.3, 0.8]$



The same is shown for and $|H| = 5$.



At a glance, for $|H| = 3$ and $|H| = 5$ Algorithm 1 seems to produce well-behaved learning curves. Not only are they monotone, but the curve of E_t seems to be smooth.

Lastly, a plot is shown where an expanding hypothesis set is used $|H| = t$. The additional random hypothesis each turn has great impact on the shape of the learning curve.

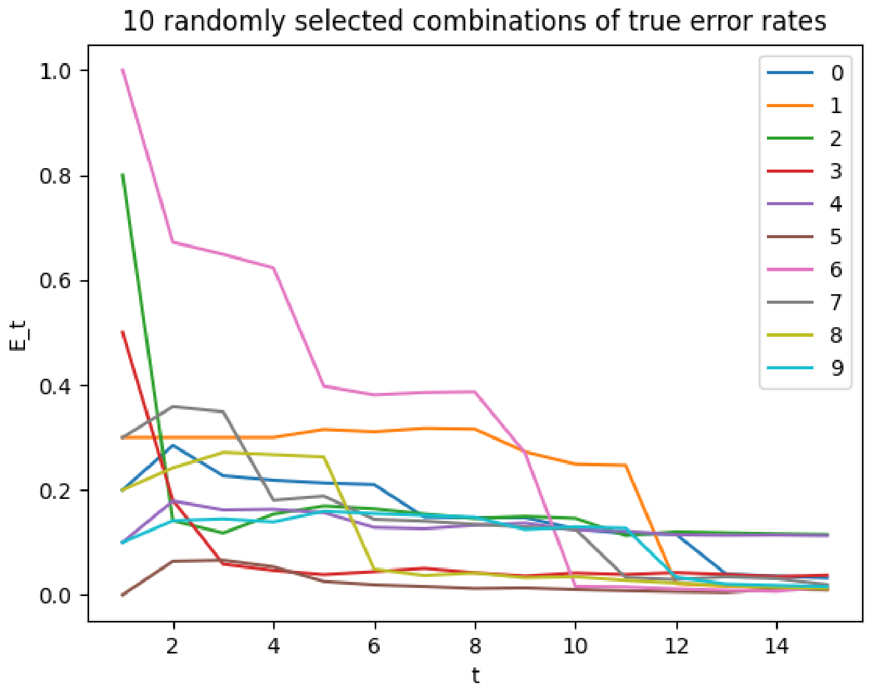


Figure 6. The indexes in the legend correspond to the ϵ values below

The following are the randomly chosen ϵ values. They are ordered in order of inclusion in H.

$$\begin{aligned}
0 =&[0.2, 0.6, 0.2, 1.0, 0.9, 0.8, 0.1, 0.6, 0.5, 0.1, 0.1, 1.0, 0.0, 0.5, 0.4]\\
1 =&[0.3, 0.3, 0.3, 1.0, 0.4, 0.3, 0.5, 0.8, 0.2, 0.2, 0.9, 0.0, 0.8, 0.4, 0.7]\\
2 =&[0.8, 0.1, 1.0, 0.4, 0.4, 0.6, 0.7, 0.7, 0.4, 0.5, 0.1, 0.2, 0.6, 0.9, 0.5]\\
3 =&[0.5, 0.1, 0.0, 0.7, 0.6, 0.3, 0.2, 1.0, 0.5, 0.1, 0.3, 0.1, 0.3, 0.6, 0.11\\
4 =&[0.1, 0.4, 1.0, 0.2, 0.8, 0.1, 0.6, 0.2, 0.3, 0.1, 0.7, 0.1, 0.5, 0.3, 0.9]\\
5 =&[0.0, 0.2, 0.6, 0.6, 0.0, 1.0, 0.5, 0.7, 0.3, 0.9, 0.9, 0.7, 0.5, 0.1, 0.7]\\
6 =&[1.0, 0.6, 0.7, 0.6, 0.3, 0.4.0.5, 0.6, 0.2, 0.0, 0.3, 0.8, 1.0, 0.7, 0.1]\\
7 =&[0.3, 0.9, 0.4, 0.1, 0.3, 0.1, 0.5, 0.7, 0.8, 0.9, 0.0, 0.6, 0.1, 1.0, 0.0]\\
8 =&[0.2, 0.3, 0.4, 0.9, 0.9, 0.0, 0.7, 0.2, 0.8, 0.2, 1.0, 0.4, 0.8, 0.7, 0.3]\\
9 =&[0.1, 0.8, 0.2, 0.9, 0.3, 0.8, 0.7, 0.9, 0.1, 0.3, 0.6, 0.0, 0.0, 1.0, 0.8]
\end{aligned} \tag{82}$$