# Appendix: Self-Healing Machine Learning: A framework for autonomous adaptation in real-world environments

## Table of Contents


**A Extended related work** **17**
- A.1 Comparison to other fields . . . 17
- A.2 Comparison on a component level . . . 18
- A.3 Unique properties of self-healing machine learning . . . 19

**B $\mathcal{H}$-LLM** **20**
- B.1 Algorithm and details H-LLM . . . 20
- B.2 Prompt templates used . . . 21
  - B.2.1 Prompts related to diagnosis . . . 21
  - B.2.2 Prompt templates related to adaptation . . . 23
- B.3 Example outputs of H-LLM . . . 24
- B.4 Evaluation strategies of self-healing algorithms . . . 27
- B.5 Computational notes . . . 27

**C Case study design** **28**
- C.1 Details on the experimental setup . . . 28
- C.2 Details on viability studies . . . 28
  - C.2.1 Viability Study I . . . 28
  - C.2.2 Viability Studies III - VI . . . 30
- C.3 Other experimental details . . . 32

**D Extended experiments** **33**
- D.1 Monitoring . . . 33
- D.2 Diagnosis . . . 33
- D.3 Adaptation experiment . . . 34
- D.4 Effects of Self-Healing across corruption levels . . . 34
- D.5 Extended benchmarks . . . 34
- D.6 Component-wise Ablation Analysis . . . 35
- D.7 Model agnosticism . . . 35

**E Optimal diagnosis** **36**

# A Extended related work

In this section, we describe and contrast our work with other related areas.

## A.1 Comparison to other fields

**Concept drift adaptation.** Concept drift adaptation algorithms, a key component of self- healing ML systems, primarily handle drifts by re-training models on new data [1–5] or older, pre-trained stored models [1, 6–8]. These approaches can be implicit, like continuous retraining, or explicit, based on drift detection in data or model error [5, 23, 27]. Drift detection methods compare distributions, analyze data sequentially, or use statistical process control [59]. For instance, the DDM algorithm [23] has in-control, warning, and out-of-control states.

**Specialized drift handling.** Techniques have been developed for various drift scenarios. For recurring drifts, methods store and reuse historical models [6, 7, 9]. Streaming data is handled by blind approaches, like sliding windows [10] or adaptive decision trees [11], and informed approaches with explicit drift detection [21–24]. Resampling can repair adaptation errors [27], while dynamic classifier selection finds the best model for each input [60]. Methods have been proposed for robustness to noise [12], specific drift types [61], and other issues [62, 63]. Recent work explores understanding distribution shifts through latent variable models [34] and other techniques [35]. Some adaptive methods of re-training the model also include adding more hidden layer to a learner upon detection of a drift [62, 63]. Another area of research closely linked within the field is dynamic selection which attempts to find the most suitable classifier conditional on the covariates [60].

**On "repairing concept drift"**. There have been other methods propose that implicitly try to adapt by detecting changes [27]. However, these adaptations are still based on the observed empirical distributions as opposed to observing the reason for degradation. By periodically sampling the accuracy of inactive classifers, the authors identify cases where change was missed or misclassifed. However, this falls under the broader umbrella of trying out many pre-determined actions without directly reasoning about the reason for model degradation.

**Continual learning**. One might get the impression that self-healing machine learning might bear close resemblance to continual learning. Continual learning focuses on developing models that learn continuously from a stream of data, acquiring, retaining, and transferring knowledge across tasks over time [64]. This contrasts strongly with self-healing machine learning. Below, **we outline seven criteria by which self-healing machine learning and continual learning differ**.

**Differences between continual learning and self-healing machine learning**.

1. **Objective**. The objectives of the two fields are different. Continual learning aims to learn sequentially from a stream of tasks while mitigating catastrophic forgetting. SHML focuses on autonomously diagnosing and recovering from performance degradation within a single task due to distribution shifts.
2. **Knowledge retention**. A core goal of continual learning is to preserve previously acquired knowledge while learning new tasks. SHML does not explicitly aim to retain prior knowledge or acquire new knowledge, but rather to maintain stable performance on the current task by adapting to the reason for degradation.
3. **Stability-Plasticity Dilemma**. Continual learning grapples with the trade-off between being plastic enough to learn new tasks and stable enough to remember old ones. In contrast, there is no such dilemma within SHML.
4. **Task expansion**. Continual learning seeks to expand the model's capabilities by increasing the number of tasks it can perform. In contrast, SHML operates on a single well-defined task—ensuring the optimal performance of a model, typically by minimizing empirical risk— and does not aim to increase the number of tasks. Instead, the focus is on ensuring optimal performance under a single task.
5. **Adaptation mechanism**. The underlying logic or mechanism of adaptation is different. Continual learning typically adapts by modifying model architecture, updating parameters via constrained optimization, using memory replay. In contrast, SHML explicitly adapts by diagnosing the root cause of performance drops and conditioning an adaptation action on the basis of that diagnosis. This explicit mechanism which is conditioned on is not a part of a continual learning system.

6. **Shift assumptions**. Continual learning primarily handles shifts across distinct tasks, where the input or output distribution changes between tasks. In contrast, SHML considers shifts within the same task, where the joint distribution might change.
7. **Theoretical formalism**. Continual learning is often formalized as a sequence of constrained optimization problems to mitigate interference between tasks. In contrast, SHML is formalized as finding an optimal policy that can propose actions on the basis of diagnoses.

### A.2 Comparison on a component level

Here, we focus on some related work within each sub-component. Table 6 provides key related work within each column. We do not focus separately on *adaptation* and *testing* because adaptation is covered above, whereas testing is simply a stage which helps to evaluate proposed actions.

| component | Definition | Methodological contribution | Experimental contribution | Main practical implications | Related work |
|---|---|---|---|---|---|
| **Monitoring** | Eq. 4 | n/a | Sec. 6.3 | More robust models against false positive drift detection (Sec. 6.3) | [4, 23, 24, 29, 39, 48] |
| **Diagnosis** | Eq. 5 | Def. 1, Def. 2, Prop. 2 | Sec. 6.4 | Established framework to reason about *why* models degrade (Sec. 4) | [36, 37] |
| **Adaptation** | Eq. 6 | Asmp. 1 | Sec. 6.5 | Targeted adaptation by identifying the root cause (Sec. 4) | [1, 21, 27, 30–32, 61] |
| **Testing** | Eq. 7 | Def. 3 | Sec. 6.6 | Principled framework to evaluate actions (Sec. 6.6) | [65–67] |
| **Self-healing ML** | Eq. 8, Sec. 3.3 | Fig. 1, Fig. 3, Sec. 3.3, Sec. 5.1, Sec. 5.2, Sec. 5.3 | Sec. 6.1 | New self-healing paradigm (Sec. 3.3) addressing prior limitations (Sec. 3.2); first self-healing system (Sec. 5). | [68–72] |

Table 6: Summary table of self-healing ML. Use this as a guiding source to navigate the paper. Related work defines the most similar available work within each component

**Monitoring**. Related work within monitoring largely relates to different statistical techniques for discovering the presence of shifts/drifts or model degradation. We see them as an integral part of SHML. However, they are also actively used by other adaptation methods to trigger adaptation systems.

**Diagnosis**. The diagnosis component is a core component of SHML. Two primary works are closely related. The first work, "why did the distribution change?" [36], attempts to factorize the change of the joint distribution into conditional distributions of each variable and attribute some changes to one of the marginals. This is achieved by modeling the change and relationship between variables as a causal mechanism. The second work, "why did the model fail?" [37] attributes model performance degradation via a causal mechanism. They assume that distribution shifts are induced due to an intervention in the causal mechanism which results in model performance changes, and uses Shapley values to attribute changes to specific distributions. These two methods are fundamentally different from SHML in multiple respects. First and most important, these works do not propose any actions on the basis of these failures or shifts. The primary goal of both works is to understand why a distribution has changed or a model has failed, attributing it to a causal mechanism, instead of *adapting* the model to perform optimally. Second, the theoretical formalism introduced is substantially different and comes with different properties. Both works operate within the directed acyclic graph (DAG) framework, whereas we operate under a diagnosis component which is defined as a vector over a space of possible reasons. Other key differences relate to the adaptation mechanism, shift assumptions, adaptation assumptions, level of granularity of the diagnosis, level of granularity of the adaptation, or testing.

Recent work has already started coming out on *understanding* distribution shifts [35]. It is known that understanding why a distribution shift happens is important for mitigating that shift [35]. Some other people have looked at modeling shifts via latent variable models without relying on access to labels at test time [34]. However, as before, these methods do not share the objective of finding optimal actions for adaptation.

**Self-healing systems outside ML**. Self-healing systems have been proposed outside of machine learning [68–72]. We view these as inspirations for our work but consider them disparate and separate because none of them touch upon the core problem of machine learning model degradation, and have not been applied in practice.

### A.3 Unique properties of self-healing machine learning

The core of self-healing machine learning revolves around two primary components: the deployed ML model $f$ and the healing system $\mathcal{H}$. Here, we provide additional clarity on these components and their interactions:

**Definition of the Deployed Model $f$** The model $f$ represents the deployed machine learning model that we aim to heal. It is the function that makes predictions on input data and whose performance we're trying to maintain and improve. In our viability studies, we demonstrate this framework using logistic regression models as $f$, though the approach generalizes to any predictive model.

**Relationship Between $f$ and $\pi$** While $f$ is the model making predictions, $\pi$ is the adaptation policy—a function that determines what actions to take to modify $f$ based on the diagnosed reasons for its performance degradation. The healing system $\mathcal{H}$ follows policy $\pi$ to output actions $a$ (such as $a_1$: retrain a model or $a_2$: remove corrupted features) which are then implemented onto $f$. Therefore, $\mathcal{H}$ follows policy $\pi$ which helps to determine optimal actions $a$ that change/modulate the deployed ML model $f$.

**Practical Implementation** In our viability studies with H-LLM, the policy $\pi$ is instantiated with an LLM (GPT-4) which uses the diagnosed reasons for model failures (also achieved with an LLM) to propose concrete actions. For instance, if $f$ is a diabetes prediction model and $\pi$ diagnoses that $f$'s performance has degraded due to concept drift, $\pi$ might suggest an action to retrain $f$ with more recent data or to adjust feature weights.

# B $\mathcal{H}$-LLM

This section provides more details on $\mathcal{H}$-LLM.

## B.1 Algorithm and details H-LLM

The algorithm of $\mathcal{H}$-LLM is presented in Algorithm 1.

**Extended discussion**.

**Algorithm 1:** $\mathcal{H}$-LLM

**Require :** $f, \mathcal{H}_M, \mathcal{H}_D, \mathcal{H}_A, \mathcal{H}_T, \tau, m, k$

$t \leftarrow 1$
$t^* \leftarrow$ null
**while** $t \leq T$ **do**
  $s_t \leftarrow \mathcal{H}_M(\{(\mathbf{x}_i, y_i)\}_{i=1}^t)$
  **if** $s_t > \tau$ *and* $t^* =$ *null* **then**
    $t^* \leftarrow t$
  **if** $t^* \neq$ *null and* $t - t^* >$ *Detection Window* **then**
    $t' \leftarrow t$
    $\mathbf{v} \leftarrow \mathcal{E}((\mathbf{x}, y) \sim \mathcal{P}_{t^*}, c \in \mathcal{C})$
    **for** $i = 1$ *to* $k$ **do**
      $\mathbf{z}_i \sim l(\cdot|\mathbf{v})$
    $\hat{\zeta} \leftarrow \{\mathbf{z}_i\}_{i=1}^k$
    **for** $j = 1$ *to* $m$ **do**
      $a_j \sim l(\cdot|\hat{\zeta})$
    $\hat{a}^* \leftarrow \arg\min_{j \in [m]} \mathcal{H}_T(f^{a_j}, \hat{\mathcal{D}}_{[t^*, t']})$
    $f \leftarrow f^{\hat{a}^*}$
    $t^* \leftarrow$ null
  $t \leftarrow t + 1$

**I. Monitoring**. We use statistical drift detection algorithms to monitor model degradation from $k$ previous time points [29, 39, 48]. Diagnosis is triggered if a shift is detected. For our practical implementation, we use the Drift Detection Method, a popular method for binary drift detection classification.

**II. Diagnosis**. Upon detection, $\mathcal{H}$-LLM uses an extractor function $\mathcal{E} : \mathcal{D}^* \rightarrow \mathcal{D}_c$ to transform the dataset information into an information vector $\mathbf{v}$. This extractor function is a mapping from the dataset to information about the dataset. It takes information before the shift happened and calculates summary statistics, such as the mean, average, standard deviation, percentiles, etc., within each column, as well as the performance of a deployed model $f$ under various data slices. For instance, this would also involve looping over all variables, binning them into 10 discrete values and calculating the average model performance across each bin. This is done to ensure that the information contained within the information vector are both summary statistics, i.e. how the data has changed, as well as specific performance metrics within data slices. The information is used to generate specific diagnoses as to what has happened. We observe, for instance, that summary statistics are extremely helpful if there are any larger deviations from average, as the diagnosis module within $\mathcal{H}$-LLM picks up on these clues. This information is provided as textual information to the next step which is the diagnosis phase.

The information vector is used as a textual representation within the next LLM call to generate concrete hypotheses / diagnoses about the reason for the $f$ failure. This is where additional context $c$ could be added, if available, such as the presence of any particular exogeneous events that could have affected model performance and could guide the diagnosis search. In the future, we envision that the additional context could be acquired by the system itself. This is used in a chain-of-thought module with self-reflection, where $k$ candidates for degradation are generated along with associated scores. We employ different "diagnosis" modules within $\mathcal{H}$-LLM. For instance, there is a specific diagnosis module that only attempts to find which covariates are responsible for degradation. The system level instruction could be as follows: "Find covariates that are responsible for the model degrading". However, we also supplement this with more broader reasons for degradation, such as "Find and hypothesize reasons that could have resulted in model degradation , given the information provided". We provide three prompt templates used to hypothesize issues in Section B.2.1. We sample such prompts $m$ times using MC sampling. The chain-of-thought and self-reflection is implemented by calling $\mathcal{H}$-LLM multiple times to re-consider the evidence and hypotheses. Table 2 illustrates diagnoses generated by $\mathcal{H}$-LLM.

**III. Adaptation**. Conditioned on the empirical diagnosis distribution $\hat{\zeta}$, $\mathcal{H}$-LLM generates $m$ candidate adaptation actions $\{a_j\}_{j=1}^m \sim l(\cdot|\hat{\zeta})$ via CoT-based MC sampling. Specifically, we focus on three kinds of adaptation actions.

- Generic adaptation actions

- Adaptation actions by removing corrupted data
- Adaptation actions by training multiple models for subsets of the data

This is reflected in three different prompt templates in Appendix B.2.2.

**Generic adaptation actions**. The first attempt is to find generic adaptation actions that the diagnosis module suggests on the basis of the identified evidence. These are often quite generic, for instance, "add new covariates that could control for the seasonality". In many such cases, within the confines of our experiments, we do not have the ability to resolve the issues on the basis of the proposed solutions. Therefore, we add two more directly actionable adaptation actions that are also attempted by $\mathcal{H}$-LLM *after* the generic adaptation actions have been attempted.

**Adaptation actions by removing corrupted data**. Another concrete adaptation action is that we instruct $\mathcal{H}$-LLM to hypothesize specific data slices that might have been corrupted. This could be, for instance, biologically implausible values (negative insulin, age > 200, implausible hba1c levels), mismatches (e.g. height, weight do not match BMI), sudden shifts in the data (ages change from averages of 30 to 60), and other. The adaptation module then proposes *which data slices to remove* to achieve superior performance. These suggested data slices are then removed and re-trained in the next batch.

**Adaptation actions by training multiple models**. The final concrete adaptation action is to propose specific data slices where the model might have drifted within that slice. This is done because instead of *global drifts*, models sometimes drift locally and require complete re-training of the new dataset.

Example outputs of such strategies are presented in Appendix B.3. We note, however, that, in reality, there might be many possible adaptation actions, such as re-training the model on combinations of old and historical data, re-using old models, re-using parts of old models, creating custom ensembles, changing models altogether, changing hyperparameters or adding regularization terms, building different models for different samples based on their difficulty, switching between symbolic and predictive ML models in the face of high uncertainty, and many more. Our approach is to introduce only the primary few ways with the hope of extending this in the future.

As before, because the actions sampled from $l$ are textual representations, we use an interpreter function to execute each $a$ on $f$.

**IV. Testing**. The sampled actions are evaluated on an empirical dataset (Def. 7), and the empirically optimal action $\hat{a}^* = \arg\min_{j\in[m]} R(a)$ is implemented on $f_{t+1}$. Limited access to the shifted DGP complicates evaluating $R(a)$, but it can be approximated with empirical data $\hat{\mathcal{D}}_{\text{test}}$ by using *a backtesting window*, *continuously incoming data*, or *historical data*. In all of our experiments, we use a backtesting window. However, other strategies could be attempted. The different strategies are explained in greater detail in Appendix B.4.

**Goal**. This procedure aims to approximate the optimal action (Def. 8). We remark that there might be better adaptation policies that could be suggested on the basis of evidence. Likewise, there might be better diagnosis modules available. We see $\mathcal{H}$-LLM as a first attempt to integrate self-healing into ML.

### B.2 Prompt templates used

The following are some of the primary prompt templates used within $\mathcal{H}$-LLM.

#### B.2.1 Prompts related to diagnosis

```
"""
        Given the following information:
        - Data before the shift: {x_before.describe()}
        - Data after the shift: {x_after.describe()}
        - Context: {context}
        - Model performance across each covariate before the shift: {
    covariate_performance_before}
        - Model performance across each covariate after the shift: {
    covariate_performance_after}

```

```
        You know for a fact that the model has degraded. Analyze the covariates and think why.

        Review each existing covariate and provide a hypothesis on whether it might have changed and resulted in the model underperforming. Provide evidence for each hypothesis and the strength of belief for each covariate.

        Format your output as follows:
        Covariate: <covariate>; Hypothesis: ...; Evidence: ...; Strength of belief: ...

        After reviewing all the covariates, assign a confidence score for each covariate indicating your confidence level that the covariate has issues. Use the following confidence levels: extremely confident, confident, somewhat confident, unsure, completely unsure. Only use 'extremely confident' if you have overwhelming evidence for your decision. Prioritize making more confident beliefs. Avoid being uncertain. Use the available inputs as well as the data to make the best possible decision. Your goal is to be correct while reducing entropy of the probabilities (be confidently correct).
        """
"""
```

Code Listing 1: Generic diagnosis prompt

```
"""
        Given the following information:
        - Data before the shift: {x_before.describe()}
        - Data after the shift: {x_after.describe()}
        - Context: {context}
        - Model performance across each covariate before the shift: {covariate_performance_before}
        - Model performance across each covariate after the shift: {covariate_performance_after}

        You know for a fact that the model has degraded. Analyze the covariates and think why.

        Then, hypothesize {n} possible covariates or combinations of covariates that might have changed and resulted in the model underperforming. Each possibility should be mutually exclusive. For example, [X1] is one possibility, [X2] is another, and [X1, X2] is a third.
        """
```

Code Listing 2: Generic diagnosis prompt for searching combinations of covariates responsible for degradation

```
"""
        Given the following information:
        - Data before the shift: {x_before.describe()}
        - Data after the shift: {x_after.describe()}
        - Context: {context}
        - Initial hypotheses on covariates or combinations of covariates that might have changed and resulted in model underperformance: {covariate_guesses}

        Summarize the provided hypotheses and assign probabilities to each hypothesis such that the total probability sums to 100%.

        Your probabilities should be reflective of the evidence and data. Uniform probabilities (10% each) implies no knowledge. 100% probability on one covariate implies certain belief. Prioritize
```

```
    making more confident beliefs. Avoid being uncertain. Use the
    available inputs as well as the data to make the best possible
    decision. Your goal is to be correct while reducing entropy of the
     probabilities (be confidently correct).

        Format each hypothesis and its probability as follows:
        Hypothesis: [<covariate1>, <covariate2>, ...]; Probability: <
    probability>
        """
```

Code Listing 3: Diagnosis probability prompt

### B.2.2 Prompt templates related to adaptation

```
        """

       Suppose the following hypothesized issues in the dataset: {
    issues}
        Data before the shift: {x_before.describe()}
        Data after the shift: {x_after.describe()}

        Suggest {self.n} possible reasons why the model might have
    failed on the basis of the issues presented. These reasons should
    be hypotheses that might have resulted in the degradation of the
    model if such hypotheses turn out to be true. These hypotheses
    also have to be likely on the basis of the issues provided. These
    hypotheses should be specific to the data itself. The goal is to
    track down specific changes within the data that could have
    resulted in the model degradation.

        Format your output as follows:

        Hypothesis: <>; Evidence: <>

        """
```

Code Listing 4: Generic adaptation prompt

```

        f"""
        Suppose the following issues in the dataset: {issues}
        Data before the shift: {x_before.describe()}
        Data after the shift: {x_after.describe()}

        Suggest {self.n} possible subgroups that if removed could
    result in better performance for the model.
        The subgroups can be single (e.g. X > x) but could also be
    multiple combinations (e.g. X > x and Y < y)
        """
```

Code Listing 5: Subgroup adaptation prompt

```
        """Suggest solutions based on removing data"""
        task = f"""
        Suppose the following issues in the dataset: {issues}
        Data before the shift: {x_before.describe()}
        Data after the shift: {x_after.describe()}

        Suggest {self.n} possible subgroups that might need re-
    training. That is, fitting a separate model on these subgroups
    might result in superior performance.
        The subgroups can be single (e.g. X > x) but could also be
    multiple combinations (e.g. X > x and Y < y)
        """
```

Code Listing 6: Subgroup retrain prompt

### B.3 Example outputs of H-LLM

```
"""
Covariate: HbA1c; Hypothesis: The distribution of HbA1c has shifted after the shift, which might have resulted in the model underperforming; Evidence: The mean of HbA1c has slightly increased from 5.699 to 5.730, and the standard deviation has also increased from 0.492 to 0.505. The model performance across different ranges of HbA1c has significantly dropped after the shift; Strength of belief: Extremely Confident

Covariate: FastingGlucose; Hypothesis: The distribution of FastingGlucose has shifted after the shift, which might have resulted in the model underperforming; Evidence: The mean of FastingGlucose has slightly increased from 99.639 to 100.519, and the standard deviation has slightly decreased from 15.341 to 15.280. The model performance across different ranges of FastingGlucose has significantly dropped after the shift; Strength of belief: Extremely Confident

Covariate: Age; Hypothesis: The distribution of Age has shifted after the shift, which might have resulted in the model underperforming; Evidence: The mean of Age has slightly increased from 49.721 to 49.900, and the standard deviation has slightly increased from 12.066 to 12.310. The model performance across different ranges of Age has significantly dropped after the shift; Strength of belief: Extremely Confident

Covariate: BMI; Hypothesis: The distribution of BMI has shifted after the shift, which might have resulted in the model underperforming; Evidence: The mean of BMI has slightly decreased from 25.075 to 24.985, and the standard deviation has slightly decreased from 4.015 to 3.903. The model performance across different ranges of BMI has significantly dropped after the shift; Strength of belief: Extremely Confident

Covariate: BloodPressure; Hypothesis: The distribution of BloodPressure has shifted after the shift, which might have resulted in the model underperforming; Evidence: The mean of BloodPressure has slightly increased from 119.417 to 120.106, and the standard deviation has increased from 14.177 to 15.205. The model performance across different ranges of BloodPressure has significantly dropped after the shift; Strength of belief: Extremely Confident

Covariate: Cholesterol; Hypothesis: The distribution of Cholesterol has shifted after the shift, which might have resulted in the model underperforming; Evidence: The mean of Cholesterol has slightly decreased from 201.108 to 198.927, and the standard deviation has slightly increased from 39.557 to 40.269. The model performance across different ranges of Cholesterol has significantly dropped after the shift; Strength of belief: Extremely Confident

Covariate: Insulin; Hypothesis: The distribution of Insulin has shifted after the shift, which might have resulted in the model underperforming; Evidence: The mean of Insulin has slightly increased from 84.667 to 84.925, and the standard deviation has slightly decreased from 45.090 to 45.057. The model performance across different ranges of Insulin has significantly dropped after the shift; Strength of belief: Extremely Confident

Covariate: PhysicalActivity; Hypothesis: The distribution of PhysicalActivity has shifted after the shift, which might have resulted in the model underperforming; Evidence: The mean of
```

```
    PhysicalActivity has slightly decreased from 3.080 to 3.020, and
    the standard deviation has slightly increased from 0.994 to 1.014.
     The model performance across different ranges of PhysicalActivity
     has significantly dropped after the shift; Strength of belief:
    Extremely Confident.
"""

#### Generated probability outputs based on the above

# Prompt for getting these hypothesses:

"""
Your goal is to provide a summary of probabilities on likelihood of
    each of the covariates resulting in the model failing
"""

Outputs:

Hypothesis: [HbA1c]; Probability: 12.5%
Hypothesis: [FastingGlucose]; Probability: 12.5%
Hypothesis: [Age]; Probability: 12.5%
Hypothesis: [BMI]; Probability: 12.5%
Hypothesis: [BloodPressure]; Probability: 12.5%
Hypothesis: [Cholesterol]; Probability: 12.5%
Hypothesis: [Insulin]; Probability: 12.5%
Hypothesis: [PhysicalActivity]; Probability: 12.5%
```

Code Listing 7: Output for guesses which covariates have shifted. This example showcases that when there is little evidence that any specific covariate has shifted more than the others

```
"""
1. Issue: Increase in standard deviation; Evidence: The standard
    deviation for most of the variables has increased in the new
    dataset, indicating increased variability in the data; Confidence:
     8
2. Issue: Change in mean values; Evidence: The mean values for most of
     the variables have changed, which could indicate a shift in the
    population being studied; Confidence: 7
3. Issue: Change in minimum and maximum values; Evidence: The minimum
    and maximum values for most of the variables have changed, which
    could indicate outliers or a change in the range of data;
    Confidence: 7
4. Issue: Change in quartile values; Evidence: The 25%, 50%, and 75%
    quartile values for most of the variables have changed, indicating
     a change in the distribution of the data; Confidence: 7
5. Issue: Negative values for Insulin and PhysicalActivity; Evidence:
    The minimum values for Insulin and PhysicalActivity are negative,
    which is not possible in a real-world scenario and indicates data
    errors; Confidence: 10
6. Issue: Change in distribution of data; Evidence: The changes in
    mean, standard deviation, and quartile values suggest a change in
    the distribution of the data, which could affect the model's
    performance; Confidence: 8
7. Issue: Increase in age range; Evidence: The minimum and maximum age
     has increased, indicating a broader age range in the new dataset;
     Confidence: 6
8. Issue: Decrease in BMI; Evidence: The mean BMI has decreased in the
     new dataset, which could indicate a change in the health status
    of the population being studied; Confidence: 6
9. Issue: Increase in Blood Pressure; Evidence: The mean Blood
    Pressure has increased in the new dataset, which could indicate a
    change in the health status of the population being studied;
    Confidence: 6
```

```
10. Issue: Decrease in Cholesterol; Evidence: The mean Cholesterol has decreased in the new dataset, which could indicate a change in the health status of the population being studied; Confidence: 6
"""
```

Code Listing 8: Generic issue response which identifies overall issues within the dataset.

```
"""
1. Subgroup: Individuals with age > 85; Reason: The maximum age has increased in the new dataset, which could be due to outliers or errors in the data.
2. Subgroup: Individuals with age < 12; Reason: The minimum age has decreased in the new dataset, which could be due to outliers or errors in the data.
3. Subgroup: Individuals with Insulin < 0; Reason: Negative values for Insulin are not possible in a real-world scenario and indicate data errors.
4. Subgroup: Individuals with PhysicalActivity < 0; Reason: Negative values for PhysicalActivity are not possible in a real-world scenario and indicate data errors.
5. Subgroup: Individuals with BMI < 12.8; Reason: The minimum BMI has decreased in the new dataset, which could be due to outliers or errors in the data.
6. Subgroup: Individuals with BloodPressure < 70.5; Reason: The minimum Blood Pressure has decreased in the new dataset, which could be due to outliers or errors in the data.
7. Subgroup: Individuals with Cholesterol < 66.3; Reason: The minimum Cholesterol has decreased in the new dataset, which could be due to outliers or errors in the data.
8. Subgroup: Individuals with FastingGlucose > 154; Reason: The maximum FastingGlucose has increased in the new dataset, which could be due to outliers or errors in the data.
9. Subgroup: Individuals with HbA1c < 4; Reason: The minimum HbA1c has decreased in the new dataset, which could be due to outliers or errors in the data.
10. Subgroup: Individuals with BMI > 39.6; Reason: The maximum BMI has increased in the new dataset, which could be due to outliers or errors in the data.
"""
```

Code Listing 9: Example response about which subgroups to remove

```
"""
1. Subgroup: Individuals with age > 85; Reason: The maximum age has increased in the new dataset, indicating a broader age range.
2. Subgroup: Individuals with age < 12; Reason: The minimum age has decreased in the new dataset, indicating a broader age range.
3. Subgroup: Individuals with BMI < 12.83; Reason: The minimum BMI has decreased in the new dataset, indicating a change in the health status of the population.
4. Subgroup: Individuals with BMI > 37.07; Reason: The maximum BMI has increased in the new dataset, indicating a change in the health status of the population.
5. Subgroup: Individuals with Blood Pressure > 166.85; Reason: The maximum Blood Pressure has increased in the new dataset, indicating a change in the health status of the population.
6. Subgroup: Individuals with Blood Pressure < 70.49; Reason: The minimum Blood Pressure has decreased in the new dataset, indicating a change in the health status of the population.
7. Subgroup: Individuals with Cholesterol < 44.64; Reason: The minimum Cholesterol has decreased in the new dataset, indicating a change in the health status of the population.
8. Subgroup: Individuals with Cholesterol > 347.08; Reason: The maximum Cholesterol has increased in the new dataset, indicating a change in the health status of the population.
```

```
9. Subgroup: Individuals with Insulin < -79.81; Reason: The minimum
    Insulin has decreased in the new dataset, indicating a data error.
10. Subgroup: Individuals with PhysicalActivity < -0.30; Reason: The
    minimum PhysicalActivity has decreased in the new dataset,
    indicating a data error.
"""
```

Code Listing 10: Example response about which subgroups to retrain the model on

### B.4 Evaluation strategies of self-healing algorithms

Self-healing relies on a testing phase, i.e. the ability to test whether the proposed actions perform well on a test dataset. However, given that the distribution has shifted and the historical data no longer represents the new distribution, one might ask: how can we test models on this new distribution? The primary alternative used in our experiments is a backtesting window which we define formally below.

**Definition 3** (Backtesting Window)**.** *Let $\{\mathcal{P}_t\}_{t\in\mathbb{T}}$ be a sequence of probability measures on $\mathcal{X}\times\mathcal{Y}$, and suppose a distributional shift occurs at time $t^*\in\mathbb{T}$, i.e., $\mathcal{P}_{t^*}\neq\mathcal{P}_{t^*-1}$. Let $t'>t^*$ be the time at which the self-healing system detects the shift. The* ***backtesting window*** *is the time interval $[t^*,t']$ satisfying the following properties:*

$$\forall t\in[t^*,t']:(\mathbf{x}_t,y_t)\sim\mathcal{P}_{t^*},$$
$$\forall t\in[t^*,t']:(\mathbf{x}_t,y_t)\not\sim\mathcal{P}_{t^*-1}.$$

We notice that the backtesting window is a unique property that arises uppon sudden shifts in the data generating process. Specifically, because we assume only two data generating processes and a transition between them at time point $t$, then all points $k$ where $k>t$ will be from the new DGP and all points $k<t$ will be from the old DGP. Since a drift detection algorithm requries some time to detect the drift, by the time a drift has been detected, we have some collected data from the new distribution which we call the *backtesting window*. We can therefore optimize our actions on this specific window of the dataset.

Clearly, this does not hold when the assumptions about the nature of the shift change. In such a case, we could always use continuously incoming streaming data. Upon the arrival of each new batch, we can test each proposed action and validate it, consistently upgrading and using the actions that perform well on the most recent batch of data. This strategy assumes that the labels are almost immediately available at prediction time. If not, another strategy employed could be to test such actions on the mot recent available data with labels.

Other approaches could include generating synthetic data to imitate the new shift with labels or using historical data by de-biasing it. However, these are experimental approaches which need further validation.

### B.5 Computational notes

**Computational overhead.** SHML methods have larger overhead than reason-agnostic approaches due to the self-healing system (LLM pipeline) identifying model failure reasons. Practically, it takes 20-40 seconds to implement a full pipeline and correct a model upon drift detection. This overhead is negligible for real-world systems given the benefits. Overhead may vary across systems.

**Sample efficiency.** No differences exist as failure detection doesn't depend on sample size, but on self-healing pipeline complexity.

# C Case study design

Code can be found at: https://github.com/pauliusrauba/Self_Healing_ML or https://github.com/vanderschaarlab/Self_Healing_ML

## C.1 Details on the experimental setup

**Experimental setup**. To evaluate the performance of self-healing systems, we require to manipulate the data generating process (DGP) and ask "what-if" questions. Real-world datasets, while valuable, do not offer control over the DGP and come with pre-embedded biases that can implicitly affect detection systems [73]. In contrast, by using synthetic data to control the DGP, we can run controlled *in silico* experiments and perform viability studies [74]. Furthermore, the overwhelming majority of model adaptation methods are designed for tabular data (refer to Sec. 2 and Sec. A) which includes our benchmarks (see Sec. 6.1). Therefore, we simulate a diabetes prediction task [49–51]. We perfectly mimic the introduced setup in Sec. 3.1. Our goal is to predict the presence of diabetes $Y_t \in \{0, 1\}$ at each time point $t$ for a set of $n$ observations, generated according to a (changing) pre-specified DGP $\log\left(\frac{P(Y_t=1|X_t)}{P(Y_t=0|X_t)}\right) = \alpha_t + \sum_{k \in K} \beta_{t,k} X_{t,k} + \epsilon_t$, where $K$ includes relevant covariates such as Age or BMI, $\beta_{t,k}$ are time-varying covariates and $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ is a noise component.

We generated synthetic data for the diabetes prediction task. Each feature is sampled from a normal distribution with specified parameters:

- Hemoglobin A1c (HbA1c) levels are sampled from a normal distribution: HbA1c $\sim \mathcal{N}(5.7, 0.5^2)$.
- Fasting Glucose levels are sampled from a normal distribution: Fasting Glucose $\sim \mathcal{N}(100, 15^2)$.
- Age is sampled from a normal distribution: Age $\sim \mathcal{N}(50, 12^2)$.
- Body Mass Index (BMI) is sampled from a normal distribution: BMI $\sim \mathcal{N}(25, 4^2)$.
- Blood Pressure is sampled from a normal distribution: Blood Pressure $\sim \mathcal{N}(120, 15^2)$.
- Cholesterol levels are sampled from a normal distribution: Cholesterol $\sim \mathcal{N}(200, 40^2)$.
- Insulin levels are sampled from a normal distribution: Insulin $\sim \mathcal{N}(85, 45^2)$.
- Physical Activity is sampled from a normal distribution: Physical Activity $\sim \mathcal{N}(3, 1^2)$.

The observations $X$ are constructed as a matrix where each row is an instance of the generated features. The outcomes are then determined by running the model through a logistic regression and obtaining a binary outcome value.

## C.2 Details on viability studies

### C.2.1 Viability Study I

**Viability Study I**. To simulate covariate shift and introduce data corruption, we follow these steps:

1. Generate two datasets with different coefficients and noise parameters:
   - The first dataset with $n_1 = 100,000$ samples, coefficients $\beta_1 = [0.3, 0.0075, -0.01, 0.05, 0.04, -0.03, -0.02, -0.1]$, and noise $\epsilon_1 \sim \mathcal{N}(0, 0.2^2)$.
   - The second dataset with $n_2 = 100,000$ samples, coefficients $\beta_2 = [-0.3, -0.0075, 0.2, -0.05, -0.015, -0.001, 0.02, -2]$, and noise $\epsilon_2 \sim \mathcal{N}(0, 0.2^2)$.
2. Split the first dataset into training and testing sets, using a 70/30 split.
3. Combine the testing set of the first dataset with the entire second dataset to form the complete testing set. The second testing set therefore contains a shift where the transitions between the DGPs happen.
4. In addition to the shift in the DGP, we introduce outliers in the second dataset by multiplying selected features by an outlier factor that control. By default, it is set to

5. This outlier factor corrupts the number of columns corrupted by $k$, and corrupts a percentage of values within the column, denoted as $\tau$.
6. The shift index is determined as the starting point of the second dataset in the combined testing set.
7. We measure and report the performance of the model during the second data generating process.

**Summary of parameters**:

- $n_1 = 100,000$: Number of samples in the first dataset.
- $n_2 = 100,000$: Number of samples in the second dataset.
- $\beta_1 = [0.3, 0.0075, -0.01, 0.05, 0.04, -0.03, -0.02, -0.1]$: Coefficients for the first dataset.
- $\beta_2 = [-0.3, -0.0075, 0.2, -0.05, -0.015, -0.001, 0.02, -2]$: Coefficients for the second dataset.
- $\epsilon_1 \sim \mathcal{N}(0, 0.2^2)$: Noise for the first dataset.
- $\epsilon_2 \sim \mathcal{N}(0, 0.2^2)$: Noise for the second dataset.
- Seed for reproducibility: 42.
- Proportion of outliers introduced: 20%.
- Features corrupted varies.

**Viability Study I. Summary of the benchmarks**. Below we describe the key benchmarks.

**Benchmark 1. New model retraining**. We use the Drift Detection Method (DDM) to monitor changes in the data distribution and retrain the model when a drift is detected. The procedure includes:

1. Split the test data into multiple batches.
2. Train the model on the initial training dataset.
3. For each batch in the test set:
   - Predict the outcomes and calculate the accuracy.
   - Update the drift detector with the prediction error (1 - accuracy).
   - If drift is detected:
     - Retrain the model on the most recent batch.

**Benchmark 2. Ensemble method**. This algorithm uses an ensemble of models to improve robustness against data shifts. It combines the predictions of multiple models, each trained on different segments of the data. The procedure involves:

1. Initialize an ensemble with a single model trained on the initial training dataset.
2. Split the test data into multiple batches.
3. For each batch in the test set:
   - Aggregate predictions from all models in the ensemble, weighted by their current accuracies.
   - Make final predictions based on the weighted aggregation.
   - Calculate the accuracy and update the drift detector with the prediction error.
   - If drift is detected:
     - Train a new model on the current batch and add it to the ensemble.
     - Update the weights of all models based on their accuracies.

This method maintains a diverse set of models that can adapt to different aspects of the data distribution, enhancing overall performance and stability.

**Benchmark 3. Partial updating**. The model is retrained using a sliding window of the most recent data batches. This allows continuous adaptation to recent changes in the data distribution. The steps are:

1. Split the test data into multiple batches.
2. Train the model on the initial training dataset.
3. Maintain a buffer to store the most recent batches.
4. For each batch in the test set:
    - Predict the outcomes and calculate the accuracy.
    - Update the buffer with the current batch.
    - If the buffer exceeds a predefined size (window size), remove the oldest batch.
    - Retrain the model using the data in the buffer.

**Our method. $\mathcal{H}$-LLM**. In this example, we use $\mathcal{H}$-LLM to identify corrupted columns and values and identify whether they need removal. The overall setup is as follows:

1. Split the test data into multiple batches.
2. Train the model on the initial training dataset.
3. Maintain buffers to store the most recent batches and a backtesting window.
4. For each batch in the test set:
    - Predict the outcomes and calculate the accuracy.
    - Update the buffers with the current batch.
    - Update the drift detector with the prediction error.
    - If drift is detected:
        - Use the self-healing mechanism to inspect the most recent and previous batches.
        - Propose multiple adaptation strategies
        - Select the best adaptation strategy on a backtesting window.
        - Retrain the model on the inspected and backtesting data to recover from the detected drift.

In all cases, the optimal strategy was removing a corrupted batch of data, where the amount of corrupted values or their extent varied.

**Comments on the experimental setup of viability study I**. The goal of this setup is to showcase that blindly retraining the model or using pre-determined actions is not necessarily optimal. In this case, the strategy required is to understand that the model requires full re-training *and* some values have been corrupted which require careful dealing, such as adjustments or removal.

### C.2.2 Viability Studies III - VI

.

**Viability Study III**. We employ the Drift Detection Method (DDM) and vary the sensitivity parameter indicated on the x-axis. We then calculate the recovery time — how much time it takes to detect the shift—, as well as the post-intervention accuracy. As discussed in the main paper, this is purely determined by the DDM. For each detected drift, we fully run $\mathcal{H}$-LLM to detect issues and propose adaptation strategies that are tested on a backtesting window. If none of them beat the performance of the current model, the existing model $f$ is deployed.

**Viability Study IV**. We evaluate how well self-healing systems identify the root causes of problems. We corrupt a proportion of observations (*corruption coefficient*) by multiplying their values by a factor (*outlier factor*) and see if the $\mathcal{H}$-LLM detects issues related to these factors. We output a probability distribution over diagnoses of which variable is corrupted. Knowing the true corrupted variable, we measure the difference between the distributions using KL-Divergence, with lower values indicating better matches between true and estimated corruption. A uniform diagnosis baseline represents random guessing. Here is an example of what it means for the "true probabilities" to be corrupted when the corrupted column is "Age".

```
true_probabilities = {'Age': 1,
'HbA1c': 0,
'FastingGlucose': 0,
'BMI': 0,
```

```
 'BloodPressure': 0,
 'Cholesterol': 0,
 'Insulin': 0,
 'PhysicalActivity': 0}
```

Code Listing 11: An example of true corrupted probabilities

Recall that $\mathcal{H}$-LLM produces normalized probability guesses, as shown in Sec. B.3. Therefore, the obtained predicted guesses of which variable is corrupted in this setup looks as follows:

```
predicted_probabilities = {'Age': 0.125,
'HbA1c': 0.125,
'FastingGlucose': 0.125,
'BMI': 0.125,
'BloodPressure': 0.125,
'Cholesterol': 0.125,
'Insulin': 0.125,
'PhysicalActivity': 0.125}
```

Code Listing 12: An example of predicted corrupted probabilities

When the corruption coefficient is higher, the output looks as follows:

```
predicted_probabilities = {'Age': 0.4,
'HbA1c': 0.2,
'FastingGlucose': 0.15,
'BMI': 0.05,
'BloodPressure': 0.05,
'Cholesterol': 0.05,
'Insulin': 0.05,
'PhysicalActivity': 0.05}
```

Code Listing 13: An example of true corrupted probabilities

Therefore, the KL divergence is computed between these two probability distributions. The KL is the highest when the outputted probability distribution is uniform (first example) and the lowest when it perfectly matches the reference/true probability distribution. It has been shown that with certain techniques, LLMs can generally output calibrated confidence scores or probabilities [75].

The reason why the KL-divergence decreases is because the predicted probabilties put greater relative value on the true corrupted value (i.e. the "Age" column in this example) as (i) the outlier factor increases and as (ii) the percent of values corrupted increase.

**Viability Study V**. We study the sensitivity of SHML adaptation policies by examining how well actions perform based on (i) the number of corrupted values and (ii) the size of the backtesting dataset. Fig. 6 shows this relationship. The corruption coefficient is described in the overall experimental setup. The size of the backtesting window is the size of the dataset used to evaluat the proposed actions. Recall that $\mathcal{H}$-LLM has three adaptation actions in place: (i) generic; (ii) filtering corrupted data slices; and (iii) training slice-specific models (Appendix B). For this experiment, we focus on actions proposed by the second adaptation strategy: filtering corrupted data slices. Each adaptation action is an identified data slice by $\mathcal{H}$-LLM that might be corrupted, the removal of which might improve performance. The following is an example of proposed adaptation actions by the removal of the following queries (each query is a separate candidate adaptation action):

```
['FastingGlucose > 376.145108',
 'Insulin > 320.642677',
 'HbA1c > 21.553946',
 'Age > 187.805319',
 'BMI > 93.998780',
 'BloodPressure > 452.899287',
 'Cholesterol > 757.675355',
 'PhysicalActivity > 11.314583',
 '(HbA1c > 21.553946) & (FastingGlucose > 376.145108)',
 '(Age > 187.805319) & (BMI > 93.998780)',
 '(BloodPressure > 452.899287) & (Cholesterol > 757.675355)',
 '(Insulin > 320.642677) & (PhysicalActivity > 11.314583)',
```

```
'(HbA1c > 21.553946) & (FastingGlucose > 376.145108) & (Age > 187.805319)',
'(BMI > 93.998780) & (BloodPressure > 452.899287) & (Cholesterol > 757.675355)',
'(Insulin > 320.642677) & (PhysicalActivity > 11.314583) & (HbA1c > 21.553946)',
'(FastingGlucose > 376.145108) & (Age > 187.805319) & (BMI > 93.998780)',
'(BloodPressure > 452.899287) & (Cholesterol > 757.675355) & (Insulin > 320.642677)',
'(PhysicalActivity > 11.314583) & (HbA1c > 21.553946) & (FastingGlucose > 376.145108)',
'(Age > 187.805319) & (BMI > 93.998780) & (BloodPressure > 452.899287)']
```

Code Listing 14: Proposed adaptation actions by removing candidate corrupted slices

Such actions are proposed for each range of values corrupted and evaluated accordingly.

**Viability study VI**. We study the importance of the testing component (Eq. 7) by evaluating $\mathcal{H}$-LLM suggested actions with and without the testing phase (backtesting window) and comparing their accuracies. Fig. 7 shows this relationship. The action with the backtesting window is the action which has received the highest empirical performance on the backtesting window. In contrast, the action proposed by "no backtesting window" is the action that is selected as the most likely one by $\mathcal{H}$-LLM without any empirical validation. "Most likely" implies that after a few iteration loops, this was the action that was listed as the first action to perform. This showcases the usefulness of having a way to filter out actions with some specific actions. We mimic the setup from study IV where each action is a specific subgroup to filter out to achieve better performance due to the corrupted nature of the data.

### C.3 Other experimental details

We note that all experiments were performed using two compute resources: a server with NVIDIA RTX A4000 GPU and 18-Core Intel Core i9-10980XE, as well as an Apple M1 Pro 32GB RAM. We exemplify $\mathcal{H}$-LLM with GPT-4 via an API.

# D Extended experiments

This section provides a few additional experiments or more detail regarding the experiments presented in the main paper.

## D.1 Monitoring

**Setup**. We vary the warm-star criterion within drift detection methods to evaluate the recovery time and post-intervention accuracy of $\mathcal{H}$-LLM. The warm start parameter is the minimum number of samples required to conclude that a drift has been detected and trigger re-training or self-healing.

**Discussion**. Fig. 8 showcases the relationship between the warm-start parameter and the average recovery tiem and post-intervention accuracy. You see the massive increase in average covery time that jumps when the warm-start is set at a relatively high threshold. This results from a drift detection algoritihm detecting a false positive drift just before the actual drift. However, given the wwarm-star parameter, there was a significant delay in re-triggering the self-healing system. This suggests self-healing systems benefit from lower warm-start parameters in case the drift detection algorithms are sensitive to false positives. This corresponds with a relative drop in the post-intervention accuracy because of the longer time it took to trigger self-healing.

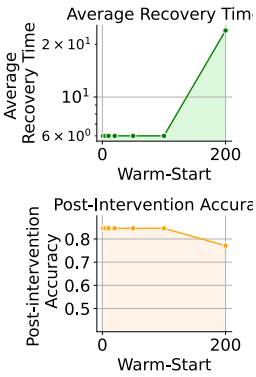


Figure 8: Adaptation strategies of different methods in response to three shifts.

**Takeaway**. Self-healing systems benefit from lower warm-start parameters in case drift detection systems are sensitive to false positive drifts.-intervention accuracy with smaller thresholds.

## D.2 Diagnosis

**Setup**. In this experiment, instead of corrupting a single variable which is responsible for model degradation, we corrupt $n$ variables to evaluate how well $\mathcal{H}$-LLM can diagnose multiple corrupted values at once. With each corrupted columns, the true corrupted probability changes. For instance, if there are four columns and there is a single corrupted column, the true corruption vector is [1, 0, 0, 0]. If there are four corrupted columns, then it is [0.25, 0.25, 0.25, 0.25]. We use these probabilities and compare them to the corruption probabilities outputted by $\mathcal{H}$-LLM. This is shown in Fig. 9.

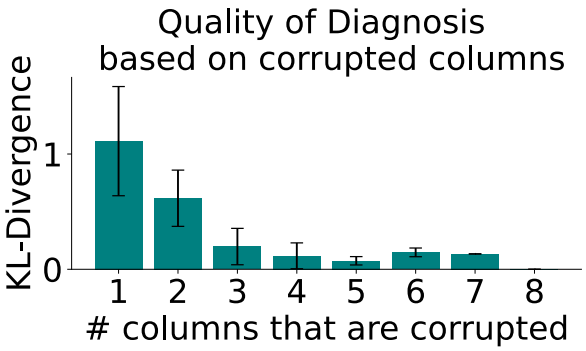


Figure 9: The qualtiy of diagnosis based on n columns. Lower is better

**Discussion**. This showcases that the more columns are corrupted, the better the predictive diagnosis becomes. For instance, once all columns are corrupted, $\mathcal{H}$-LLM outputs a uniform diagnosis because it has no information given the evidence observed. This exactly corresponds to the true corruption probability, outputting a KL of 0. We notice that the KL generally decreases with the number of corrupted columns for this reason.

**Takeaway**. Greater uncertainty results in more uniform diagnosis. However, less uncertainty can make it difficult to directly pinpoint the exact cause, causing more uncertainty.

### D.3 Adaptation experiment

This section expands on the adaptation experiments by providing more variables and values by corruption coefficient and the number of columns corrupted.

Table 7: Accuracy based on the number of corrupted columns, where 5% of values given a selected column are corrupted on a shifted dataset with a number of corrupted values. Higher is better. (with a corruption coefficient of 0.05)

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| No retraining | 0.43 ± 0.02 | 0.44 ± 0.02 | 0.44 ± 0.02 | 0.44 ± 0.02 | 0.45 ± 0.02 | 0.45 ± 0.02 | 0.45 ± 0.02 | 0.45 ± 0.02 |
| Partially Updating | 0.72 ± 0.02 | 0.71 ± 0.02 | 0.70 ± 0.02 | 0.69 ± 0.02 | 0.68 ± 0.02 | 0.67 ± 0.02 | 0.65 ± 0.02 | 0.54 ± 0.06 |
| New model training | 0.71 ± 0.02 | 0.70 ± 0.02 | 0.69 ± 0.02 | 0.69 ± 0.02 | 0.68 ± 0.02 | 0.67 ± 0.02 | 0.64 ± 0.02 | 0.50 ± 0.02 |
| Ensemble Method | 0.71 ± 0.02 | 0.70 ± 0.02 | 0.69 ± 0.02 | 0.69 ± 0.02 | 0.68 ± 0.02 | 0.67 ± 0.02 | 0.64 ± 0.02 | 0.50 ± 0.02 |
| $\mathcal{H}$-LLM | 0.95 ± 0.01 | 0.93 ± 0.01 | 0.90 ± 0.02 | 0.87 ± 0.01 | 0.84 ± 0.02 | 0.79 ± 0.02 | 0.77 ± 0.02 | 0.68 ± 0.02 |

Table 8: Accuracy based on the number of percent of corrupted value within a given column (with three corrupted columns with three corrupted columns)

| | 0.01 | 0.02 | 0.05 | 0.1 | 0.2 | 0.3 | 0.5 |
|---|---|---|---|---|---|---|---|
| No retraining | 0.43 ± 0.02 | 0.44 ± 0.02 | 0.44 ± 0.02 | 0.45 ± 0.02 | 0.46 ± 0.02 | 0.48 ± 0.02 | 0.49 ± 0.03 |
| Partially Updating | 0.74 ± 0.03 | 0.72 ± 0.02 | 0.70 ± 0.02 | 0.66 ± 0.02 | 0.62 ± 0.02 | 0.57 ± 0.02 | 0.52 ± 0.03 |
| New model training | 0.77 ± 0.02 | 0.74 ± 0.02 | 0.69 ± 0.02 | 0.66 ± 0.02 | 0.61 ± 0.02 | 0.55 ± 0.02 | 0.51 ± 0.03 |
| Ensemble Method | 0.77 ± 0.02 | 0.74 ± 0.02 | 0.69 ± 0.02 | 0.66 ± 0.02 | 0.61 ± 0.02 | 0.55 ± 0.02 | 0.51 ± 0.03 |
| $\mathcal{H}$-LLM | 0.95 ± 0.01 | 0.94 ± 0.01 | 0.90 ± 0.02 | 0.82 ± 0.02 | 0.70 ± 0.02 | 0.57 ± 0.02 | 0.52 ± 0.03 |

### D.4 Effects of Self-Healing across corruption levels

We systematically analyze how self-healing effectiveness varies with corruption levels across our five datasets (Airlines, Poker, Weather, Electricity, and Forest Type). For each dataset, we vary both the corruption value $\tau$ and the number of corrupted columns $k$, measuring accuracy with and without the self-healing mechanism. Figure 10 shows that self-healing's impact grows with corruption severity. Specifically, as either $\tau$ or $k$ increases, the gap between baseline and self-healed performance widens. This pattern holds consistently across all datasets, though with varying magnitudes. These results demonstrate that self-healing becomes more crucial as data degradation becomes more severe, providing a safety mechanism for maintaining model performance under challenging conditions.

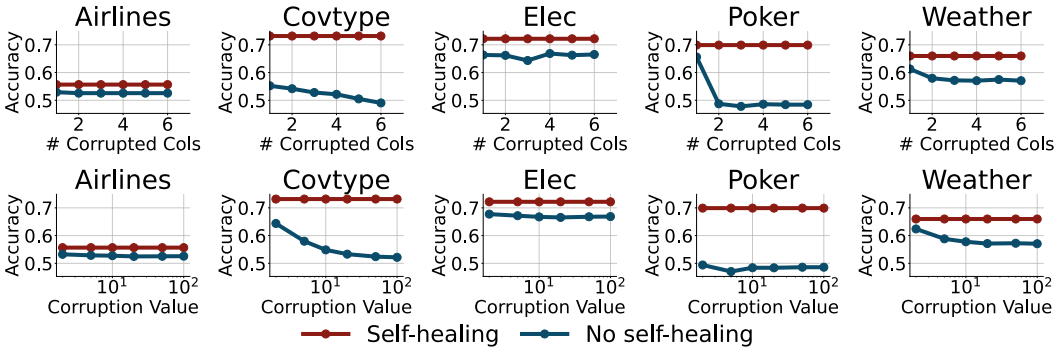


Figure 10: Effects of self-healing for five datasets as we vary the number of corrupted columns and the corruption value. Self-healing consistently identifies corrupted columns at test time. This typically becomes more important as the corruption level increases (either by corruption value or number of corrupted columns). Baseline is not implementing a self-healing mechanism upon drift detection.

### D.5 Extended benchmarks

We extend our comparison on the diabetes prediction task to include additional adaptation methods and adaptive algorithms. Table 9 presents results from ten different approaches, including standard adaptations (no retraining, partial updates, new model training, ensemble methods), streaming-specific algorithms (ADWIN Bagging, Hoeffding Tree), and our SHML approach.

| Method | Adaptations | | | | | Algorithms | | | | SHML |
|---|---|---|---|---|---|---|---|---|---|---|
| | No retraining | Partially updating | New model training | Ensemble method | Airstream | ADWIN Bagging | Hoeffding Tree | Adaptive Voting | Adaptive RF | Self-Healing ML |
| **Accuracy** | 0.52 ± 0.16 | 0.65 ± 0.13 | 0.65 ± 0.12 | 0.64 ± 0.12 | 0.59 ± 0.13 | 0.68 ± 0.10 | 0.70 ± 0.10 | 0.62 ± 0.11 | 0.69 ± 0.09 | **0.76 ± 0.08** |

Table 9: Accuracies of various adaptations on the original diabetes dataset setup in the paper.

The results show that while specialized streaming algorithms (e.g., Hoeffding Tree at 0.70 accuracy) outperform basic adaptations, they still fall short of SHML's performance (0.76 accuracy).

### D.6 Component-wise Ablation Analysis

To understand the importance of each SHML component, we conduct an ablation study by systematically removing each component and observing the impact. Table 10 shows the results of this analysis.

| Ablation | Accuracy (%) | Takeaway |
|---|---|---|
| Baseline (no self-healing) | 52 | Accuracy is worse without self-healing |
| Full (full self-healing) | 76 | Self-healing improves accuracy over baseline. |
| No monitoring | 52 | Monitoring is required to trigger the SHML system. $\mathcal{H}$-LLM was not triggered and no actions were proposed. |
| No diagnosis | 52 | Diagnosis is required for proposing sensible actions. Defaults to non-sensical actions. |
| No actions | 52 | Actions could not be implemented because they were not proposed, defaults to no behavior. |
| No testing | 62 | Actions chosen but not tested against empirical data. A suboptimal action was chosen. |

Table 10: Ablation study results for $\mathcal{H}$-LLM. We systematically remove one component of the system and inspect its outputs. The takeaway represents our qualitative evaluation.

The ablation reveals that each component is crucial for effective self-healing. Removing monitoring (52% accuracy) prevents the system from triggering adaptation. Without diagnosis, the system proposes non-sensical actions, leading to baseline performance. Removing action generation or testing similarly degrades performance to baseline levels, though testing removal shows slightly better performance (62%) as some reasonable actions are still attempted, albeit without proper validation.

This analysis empirically validates our framework's design, showing that effective self-healing requires all four components working in concert.

### D.7 Model agnosticism

We evaluate SHML's effectiveness across ten different ML models to demonstrate its model-agnostic nature. Table 11 shows results for models ranging from simple (e.g., Decision Trees) to complex (e.g., XGBoost), comparing various adaptation strategies. SHML consistently outperforms baseline approaches across all model types, with improvements ranging from 11 percentage points (Naive Bayes) to 31 percentage points (LDA). This consistent improvement demonstrates that SHML's benefits are not tied to any particular model architecture but rather stem from its ability to reason about and address degradation causes.

| *Method* | DecisionTree | KNN | LDA | LogisticRegression | MLP | NaiveBayes | Perceptron | RandomForest | SGD | XGBoost |
|---|---|---|---|---|---|---|---|---|---|---|
| Baseline (No retraining) | 0.63 ± 0.05 | 0.51 ± 0.03 | 0.47 ± 0.03 | 0.49 ± 0.02 | 0.63 ± 0.04 | 0.51 ± 0.03 | 0.49 ± 0.01 | 0.63 ± 0.05 | 0.47 ± 0.03 | 0.67 ± 0.05 |
| Sliding Window | 0.63 ± 0.05 | 0.51 ± 0.03 | 0.47 ± 0.03 | 0.49 ± 0.02 | 0.66 ± 0.03 | 0.51 ± 0.03 | 0.49 ± 0.01 | 0.70 ± 0.05 | 0.47 ± 0.03 | 0.67 ± 0.05 |
| Drift Detection (DDM) | 0.63 ± 0.05 | 0.51 ± 0.03 | 0.47 ± 0.03 | 0.49 ± 0.02 | 0.64 ± 0.04 | 0.51 ± 0.03 | 0.49 ± 0.01 | 0.66 ± 0.05 | 0.47 ± 0.03 | 0.67 ± 0.05 |
| Ensemble with DDM | 0.63 ± 0.05 | 0.51 ± 0.03 | 0.47 ± 0.03 | 0.49 ± 0.02 | 0.65 ± 0.05 | 0.51 ± 0.03 | 0.49 ± 0.01 | 0.65 ± 0.07 | 0.47 ± 0.03 | 0.67 ± 0.05 |
| **$\mathcal{H}$-LLM** | **0.70 ± 0.04** | **0.73 ± 0.05** | **0.77 ± 0.04** | **0.76 ± 0.04** | **0.78 ± 0.05** | **0.62 ± 0.02** | **0.68 ± 0.09** | **0.72 ± 0.04** | **0.75 ± 0.04** | **0.71 ± 0.04** |

Table 11: Comparison of various methods across different ML models on the weather dataset (setup above), where features are corrupted at test time. Results show mean accuracy ± standard deviation.

## E Optimal diagnosis

Here, we prove that under the stated assumptions, the optimal diagnosis has zero entropy.

**Proposition 3.** *Under Assumption 1, the optimal diagnosis $\zeta^*$ has a zero entropy, i.e., $\mathbb{H}(\zeta^*) = 0$.*

*Proof.* By Definition 2,

$$\zeta^* = \arg\min_{\zeta \in \Delta(\mathcal{Z})} \mathbb{E}_{a\sim\pi(\cdot|\zeta)}[R(a)] \tag{12}$$

As $\mathcal{A}$ is finite, we write the expected value as follows.

$$\mathbb{E}_{a\sim\pi(\cdot|\zeta)}[R(a)] = \sum_{a\in\mathcal{A}} R(a)\pi(a|\zeta) \tag{13}$$

By Assumption 1, this can be rewritten as:

$$\sum_{a\in\mathcal{A}} R(a)\left(\sum_{z\in\mathcal{Z}} \pi(a|z^\dagger)\zeta(z)\right) \tag{14}$$

We change the order of summation to arrive at the following.

$$\sum_{z\in\mathcal{Z}} \zeta(z) \sum_{a\in\mathcal{A}} R(a)\pi(a|z^\dagger) \tag{15}$$

The inner sum can now be rewritten as an expectation.

$$\sum_{z\in\mathcal{Z}} \zeta(z)\mathbb{E}_{a\sim\pi(\cdot|z^\dagger)}[R(a)] \tag{16}$$

Thus we can rewrite the minimization problem as follows.

$$\zeta^* = \arg\min_{\zeta\in\Delta(\mathcal{Z})} \sum_{z\in\mathcal{Z}} \zeta(z)\mathbb{E}_{a\sim\pi(\cdot|z^\dagger)}[R(a)] \tag{17}$$

Let $z^* \in \mathcal{Z}$ such that

$$z^* \in \arg\min_{z\in\mathcal{Z}} \mathbb{E}_{a\sim\pi(\cdot|z^\dagger)}[R(a)] \tag{18}$$

Then

$$\begin{aligned}\sum_{z\in\mathcal{Z}} \zeta(z)\mathbb{E}_{a\sim\pi(\cdot|z^\dagger)}[R(a)] &\geq \sum_{z\in\mathcal{Z}} \zeta(z)\mathbb{E}_{a\sim\pi(\cdot|(z^*)^\dagger)}[R(a)] \\ &= \mathbb{E}_{a\sim\pi(\cdot|(z^*)^\dagger)}[R(a)] \\ &= \sum_{z\in\mathcal{Z}} (z^*)^\dagger(z)\mathbb{E}_{a\sim\pi(\cdot|z^\dagger)}[R(a)]\end{aligned} \tag{19}$$

Therefore

$$\zeta^* = (z^*)^\dagger \tag{20}$$

and by the definition of entropy and $(z^*)^\dagger$ we get

$$\mathbb{H}(\zeta^*) = 0 \tag{21}$$

□

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The paper presents a novel paradigm called self-healing machine learning. This is established in Section 3.3 and Sec. 4, where we develop the theoretical underpinnings of this field. We also make significant practical contributions by proposing the first-ever self-healing algorithm presented in Sec. 5. We discuss the positive impacts of this technology in the introduction and the discussion section, where we argue it can have transformative effects in a variety of real-world situations. We clearly show the viability of self-healing machine learning in Sec. 6.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Self-healing machine learning inherently assumes that diagnoses can be performed well and that actions can be proposed on the basis of these diagnoses. This poses challenges we discuss in Sec. 5.1. We attempt to overcome these limitations by using language models and incorporating their unique properties into $\mathcal{H}$-LLM. We discuss further challenges of self-healing systems in the discussion section (Sec. 7)

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: In order to establish the relationship between diagnosis and the actions, we define the certainty diagnosis (Def. 1) and the optimal diagnosis (Def. 2). Furthermore, we assume independent actions (Assumption 1). Under this, we establish two propositions about the entropy of an optimal diagnosis (Proposition 1) and its existence (Proposition 2). One of the proofs is in the main paper and the other one can be found in Appendix E.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide all the required information to reproduce our algorithm $\mathcal{H}$-LLM (Appendix. B) as well as the full detail on experiments, including dataset generation, parameters, etc. (Appendix. C). We provide an additional appendix section with supplementary experiments that can aid the reviewers where we discuss the setups within each experiment (Appendix. D). The code can be found at: `https://github.com/pauliusrauba/Self_Healing_ML` or `https://github.com/vanderschaarlab/Self_Healing_ML`

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide full experimental details, where all access to data and code is fully available. Code can be found at: `https://github.com/pauliusrauba/Self_Healing_ML` or `https://github.com/vanderschaarlab/Self_Healing_ML`.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All details are provided in Appendix. C. The experimental setting is presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them. Full details provided in the appendix.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All appropriate experiments report error bars.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Yes, all relevant information is provided.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Yes. We discuss this in Sec. 7. To reiterate, we believe our work can have significant positive effects on multiple areas within AI and have substantial practical implications, as discussed in the Discussion section. This includes having immediate practical benefits in industries where model degradation is common, such as medicine, finance, predictive policing, IoT data streams, and more. We further hope our work spurs substantial developments in self-healing theory. We also discuss that the unique improvements in systems that can employ self-healing could also be misused by agents for other purposes, such as using self-healing for surveillance systems or other ethically ambiguous technologies. The broader impact also subsumes future work in this area. We hope that a potential direction for future work is building theory around optimal diagnoses, optimal adaptation strategies, as well as scaling larger algorithms.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We do not release any data or models with a high risk for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer:

    Justification: We cite and refer to all appropriate codebases that are employed as benchmarks.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The core assets, including the framework and the algorithm, are clearly described.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.