# Investigation of error tolerant nature of machine learning algorithms

Hyungmin Cho
endroit@stanford.edu

## 1. Introduction

Soft computing differs from conventional (hard) computing in that, unlike hard computing, it is tolerant of imprecision, uncertainty, partial truth, and approximation. This definition implies two important things. The computation result from those soft computations can be considered to be valid if it is within acceptable range even with some inaccuracy, and those applications are less sensitive to erroneous computations and invalid or corrupted data values. These observations made the basis of building a computer architecture which is specially designed to such error tolerance soft computations. In this computer architecture, we will have relaxed requirements for correct computations and by doing that, we can expect huge savings in computation costs. Many of machine learning algorithms are consisted of soft computations. In this paper, we will explore various kinds of machine learning algorithms to find out how much inaccuracy those algorithms can tolerate by injecting artificial inaccuracy during the computation process. This will be used for building the application aware computer architecture that will exploit the error tolerant nature of the machine learning algorithms.

## 2. System Architecture for Application Aware Resilience

Computing system we will use for soft computing machine learning applications is Error Resilient System Architecture (ERSA), which is designed to execute error resilient applications with relaxed reliability hardware. The underlying architecture of ERSA is based on multi core architecture with asymmetric reliability level. There is little number of reliable computing cores in the system, which is responsible for executing error intolerant or critical computations, and the rest of the computing system is consisted of inexpensive, relaxed reliability cores which will execute the error tolerant computation. Reliable computing cores, which is called super reliable core (SRC), has similar reliability requirement like modern computing system, which means the target error rate is near zero. Therefore, there can be huge amount of cost overhead to ensure this level of reliability. Enough time slack in circuit level, voltage band gap or extra error detection/correction mechanisms are the sources of such overhead. In ERSA, the portion of computing components which needs such a high reliability is limited to very little, to minimize overall overhead. Relaxed reliability cores (RRCs) are allowed to generate some amount of computation errors. Besides of eliminating overhead for reliability, RRCs can have more radical approach to reduce computation costs like reduce CMOS feature size or reduce the operation voltage.

### 2.1. Computation model

When we execute machine learning algorithms on ERSA, we will execute the main execution path on the SRC, and execute individual computation tasks on RRCs. Because the computations tasks on RRCs can be crashed during execution because of the underlying hardware errors, ERSA has runtime software to manage re-execution of crashed tasks.

## 3. Machine learning algorithms

Not all machine learning applications can be executed on ERSA and still have high quality acceptable results. Following is the list of characteristics of algorithms that can be executed on ERSA.

- Parallel computations

  ERSA has multi-core based architecture which assumes the application is highly parallelizable. If the ML algorithm is not parallelizable, then it cannot have benefits from the ERSA architecture.

- Iterative computations

  If the computation is done in iterative fashion, the overall convergence process can tolerate errors in intermediate computation state, because the consecutive computations can recover from the previous incorrect computations.

- Individual training data computations

  If the algorithm doesn't have iterative computations, at least its computation tasks can be divided into individual computations from different data instances. By ensuring those criteria, erroneous computations from one computation task can be prevented to be propagated to other computation tasks.

Following is the selected machine learning applications to be investigated in this paper

- Least Mean Square
  We used Newton's method to optimize the parameters of LMS algorithm. Errors will be injected to gradient and hessian computations in learning phase.

- Logistic Regression
  We used Newton's method for Logistic regression also.

- Support Vector Machine
  Simplified SMO algorithm was used for optimization algorithm. Errors will be injected to kernel computations in learning phase.

- K-Means Clustering
  We applied K-Means Clustering algorithm for image clustering based on a pixel's RGB value. Errors will be injected when finding the closest centroid. When an error is injected, the id of closest centroid for each data point will be changed randomly.

- Bayesian Network with Loopy Belief Propagation
  Bayesian Network structure was applied to learn and detect object images based on spatial context. The core algorithm used to get posterior probabilities over the Bayesian network is loopy belief propagation. Error will change the value of message between clusters.

### 3.1. Dataset

We used input data from the UCI machine learning data base and the input data provided for CS229 and CS228 homework assignments.

## 4. Error model

Actual computation error an application will face will come from the physical characteristics of underlying hardware. Therefore, it will be very hard to build a concrete error model without the actual implementation of the computer system. In this stage, therefore, we assumed there will be random errors during the computation tasks on the RRCs. We will set a certain error rate, and select individual computation tasks randomly during the computation and add disturbance value to the original value. The disturbance value will also be chosen randomly from Gaussian distribution.

$$X_e = X + \Psi$$
$$\Psi \sim N(0, X)$$

If the computation result is not a real or integer value, such as category ID, then we will change the original value with a random number within possible range.

$$X_e = random(\min(X)..\max(X))$$

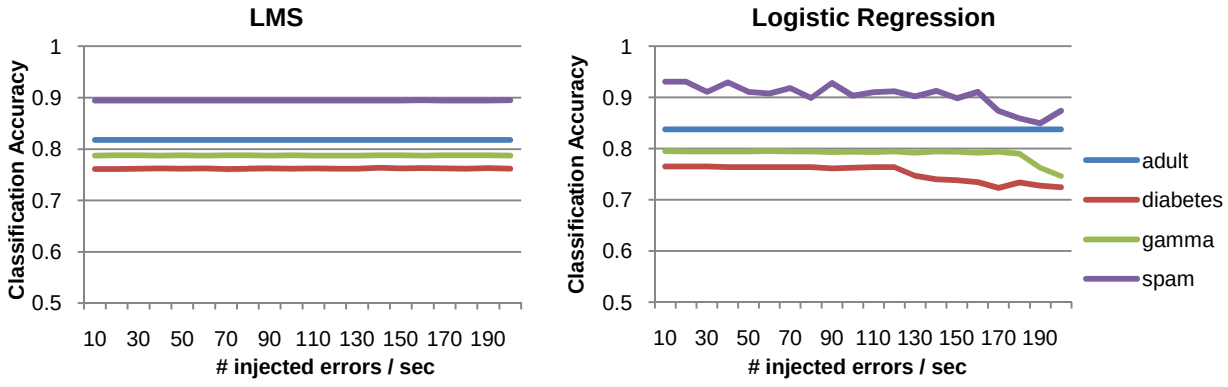
To find out error tolerance level, we will do the experiment for different error injection rates and will compare the output quality.

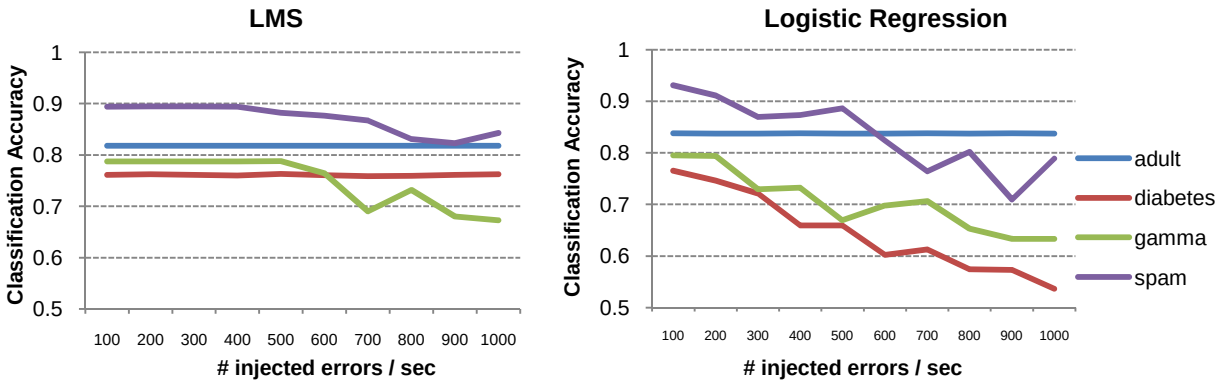
## 5. Experiment result

According to the error model we artificially inject errors into the original program and measured how much quality degradation happens as the error rate increases. The basic quality measurement is percentage of correct classification. We will compare the correct classification percentage with the original error free execution.

For K-means clustering, using the number of miss-clustered points as the measure of the quality is not a good idea, because according to the initial starting point, the clustered result can be different. Instead, we measured the output quality of the clustering by the average diameter of the cluster. The purpose of the K-means clustering iteration is achieving the tightest clustering by finding best cluster centers. Thus, if the resulting cluster diameter is larger than the diameter from error free execution, then we can use the looseness of the diameter as the measure of quality degradation.

### 5.1. Comparison of LMS and Logistic Regression



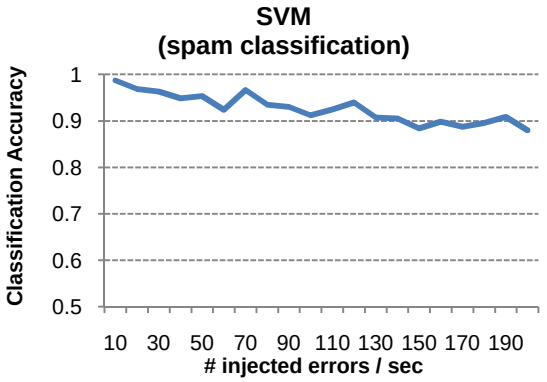



LMS and Logistic regression is very similar algorithm, so we could apply same dataset for two classification algorithms. It turns out that in the original error free execution, logistic regression is better in terms of correct classification percentage, but as we increase the error injection rate, LMS is more robust and sustained its result quality with higher error rate. Possible explanation is the gradient of LMS is linear while that of Logistic Regression isn't, it can suppress the unstable behavior according to the injected disturbance values. This discovery gave us an insight that we can have one more options to choose machine learning algorithm according to the application's purpose. If the application wants low cost classification algorithm, we can choose LMS instead of Logistic regression and execute it on error prone, unreliable hardware which yields low cost computation.

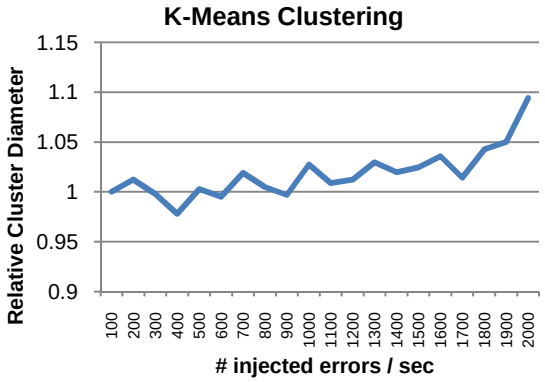
## 5.2. Various ML algorithms

We also did same experiments on different ML algorithms. However, we failed to collect sufficient amount of dataset that can be applied to all algorithms, following algorithms are tested with only one dataset. It is not sufficient to characterize the details error tolerant nature of the algorithms, but we can see certain level of error resiliency in each algorithm.

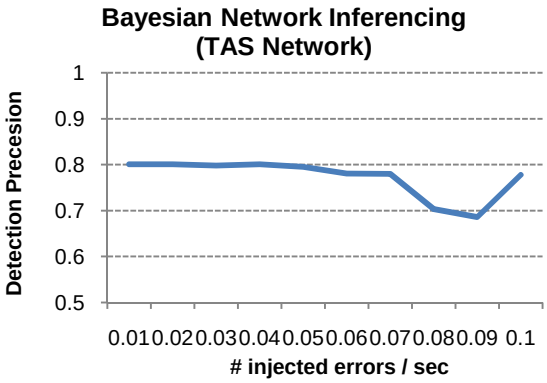
### 5.2.1. Support Vector Machine




### 5.2.2. K-means Clustering




#### 5.2.3. Bayesian Network Inference with Loopy Belief Propagation




## 6. Conclusion

In this paper, we explored several kinds of machine learning algorithms and figured out how much amount of errors they can tolerate with small impact on result quality. We also have concluded that the selection of right algorithm according to the purpose of application is crucial to design a system with desired level of reliability. This result will be used to actual implementation of ERSA system to determine which level of unreliability it should have. Also, based on this result, we can extend the research to find what we can have to increase the resiliency of given machine learning algorithm.

## 7. References

[1] Y. Jin. (2006, Jan.) Soft Computing. [Online]. http://www.soft-computing.de/def.html

[2] UC Irvine Machine Learning Repository. [Online]. http://archive.ics.uci.edu/ml/

[3] G. Heitz and D. Koller, "Learning Spatial Context: Using Stuff to Find Things," in *European Conference on Computer Vision (ECCV)*, 2008.

[4] A. Ng. CS 229 Notes. [Online]. http://www.stanford.edu/class/cs229/materials.html