A Fusion Algorithm for Uniform Sensor Failure

Matt Higger, Murat Akcakaya,
Deniz Erdogmus

Cognitive Systems Laboratory
Northeastern University

ADSA
10/25/2012

Results

Bad news…

- If a sensor breaks during operation, under the assumption that sensors yield independent decisions given truth, we cannot benefit from that sensor through unsupervised or semisupervised learning.
- Consequently, broken sensors need to be detected and replaced.
- In the mean time, fusion system must operate robustly with broken sensor in place.

Good news…

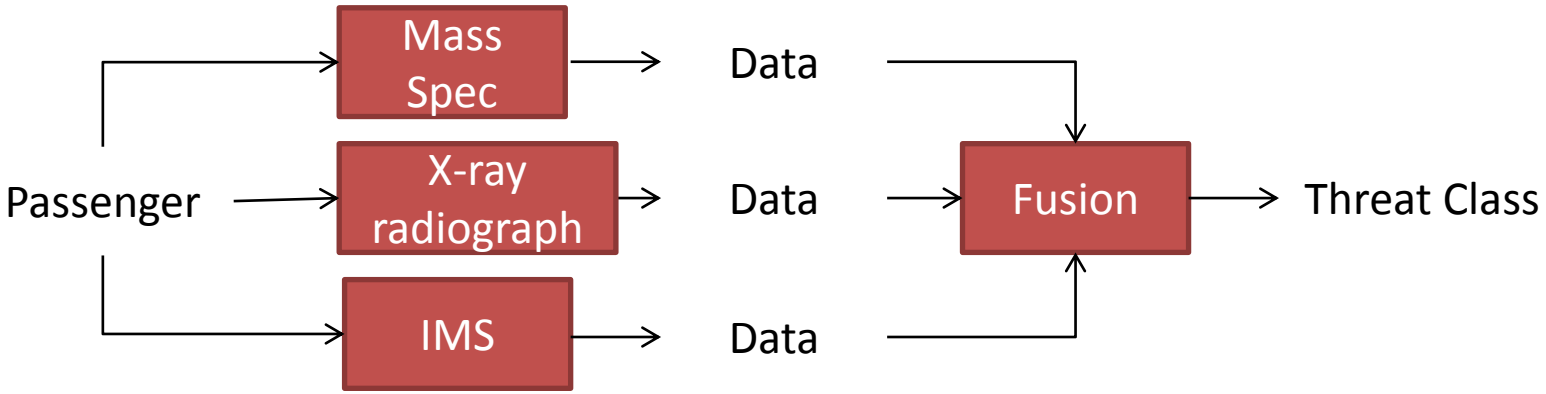
- Given a sensor failure model, we can design robust fusion rules that…
  - perform optimally with properly calibrated and correctly working sensors
  - outperform naïve fusion that assumes no sensor failure
- We can detect failed sensors from operational data, earlier in for some sensor characteristics, later for others…

Additional note…

- We reconsider the optimality objective in statistical decision theory.
- Traditional Bayes classifier remains a (rudimentary) special case.

Problem: Sensor Failure

Generic Model of Fusion:



A fusion algorithm is only as good as the sensors it relies on. A single failed sensor can introduce a large amount of risk to an otherwise operational system.

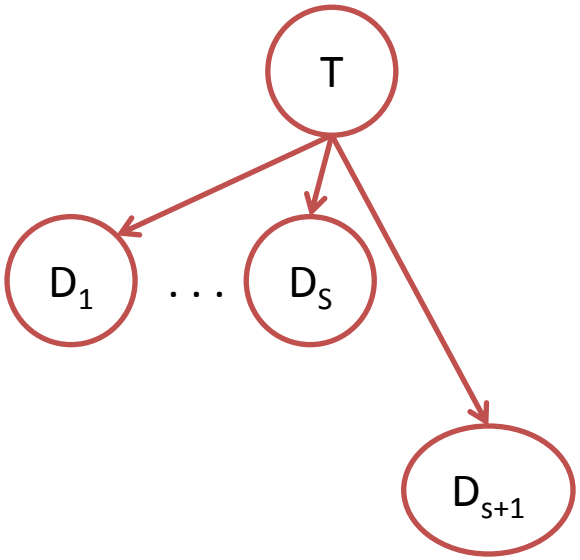
Solution:

- Learn failed sensor's characteristic online (Difficult or Impossible)
- Build a fusion rule robust to sensor failure (Current Work)
- Find and remove failed sensors from fusion (Future Work)

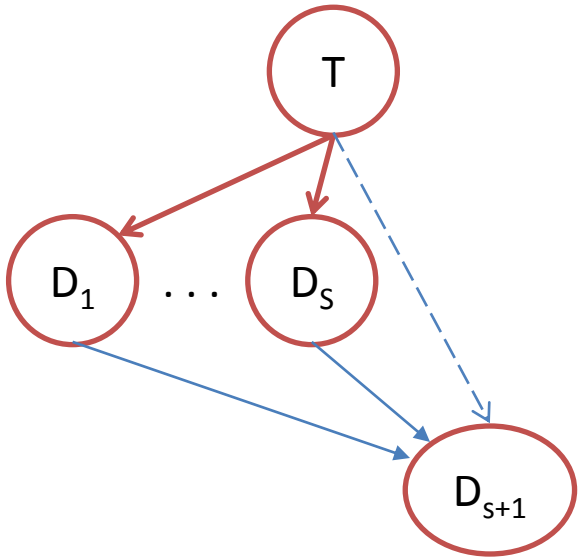
Learn Failed Sensor Characteristic Online?

For a Naïve Bayes Model …

Functional Sensors:



D_{s+1} Failing:



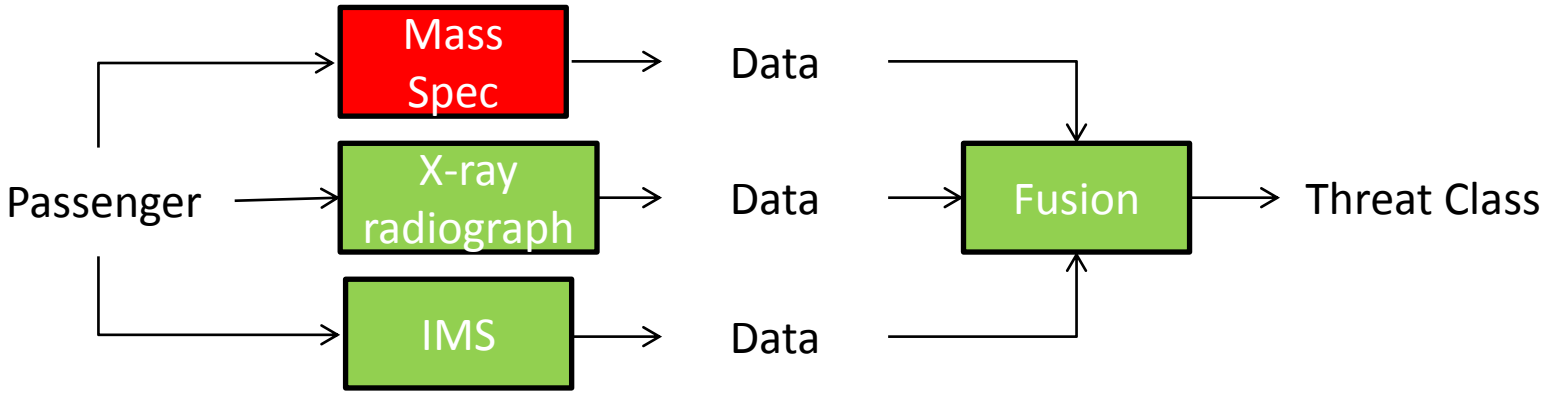
In Naïve Bayes model, adding a broken sensor's new characteristic (online) can not improve classification performance.

Build a Fusion Rule Robust to Sensor Failure

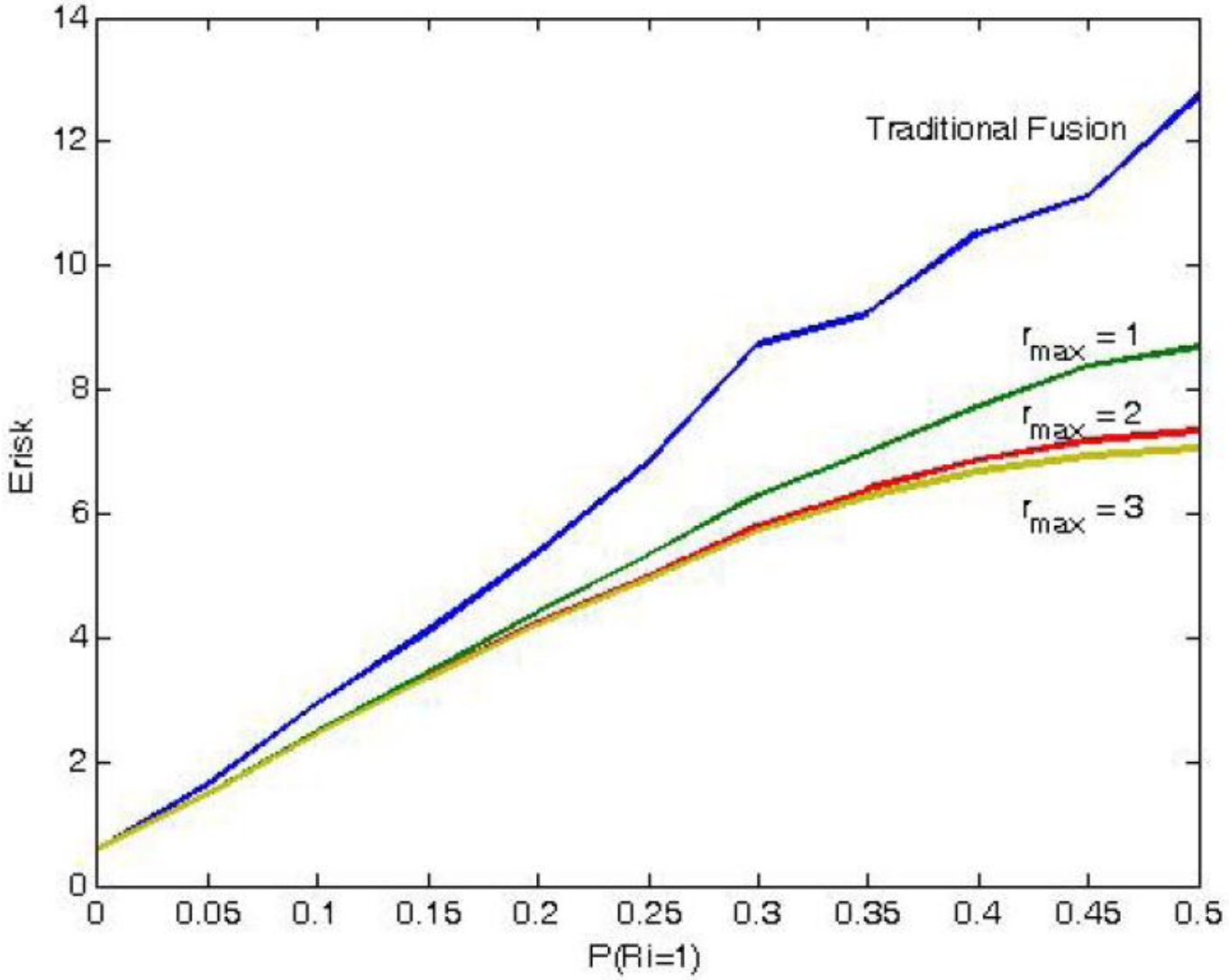
Given a uniform model of sensor failure, we can minimize risk by deciding:

$$F(d_{1:S}) = 1 \quad \text{when} \quad \log \frac{r_{1|0} P_T(1)}{r_{0|1} P_T(0)} + \sum_{D_i} log \frac{P_{D_i|T}(d_i|1)}{P_{D_i|T}(d_i|0)} + log \frac{\sum_{r=0}^{r=S} C_1(d_{1:S}, r)}{\sum_{r=0}^{r=S} C_0(d_{1:S}, r)} < 0$$

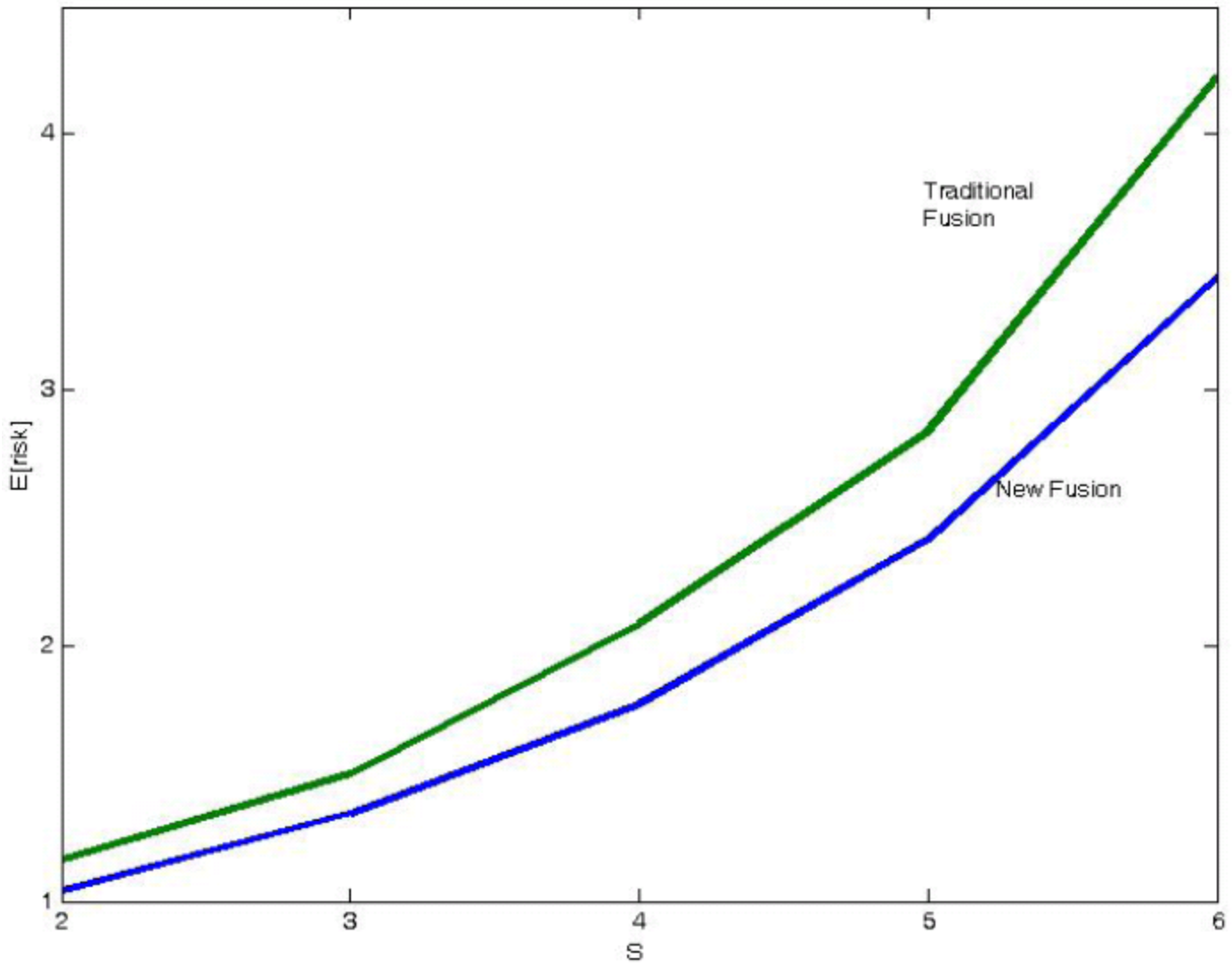
$$= 0 \quad \text{otherwise}$$



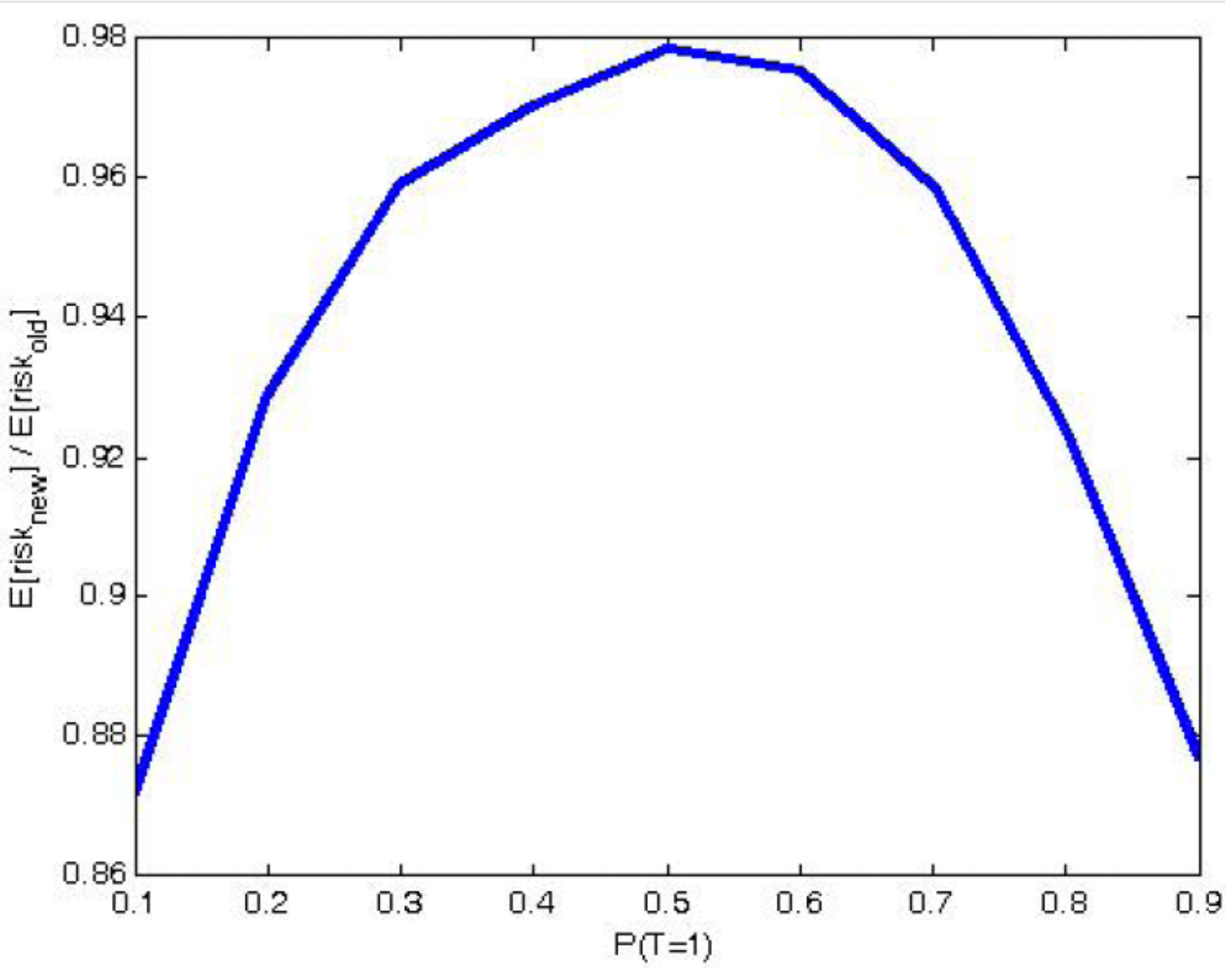
Quantitatively (1 of 3)



Quantitatively (2 of 3)



Quantitatively (3 of 3)



Summary of Robust Fusion Results

We created a novel algorithm which minimizes risk when any number of sensors have failed.

Advantages: Never outperformed by Naïve Bayes fusion & can reduce risk by up to 30%

- Reduces more risk as number of sensors, S, increases
- Reduces more risk as probability of sensor failure increases
- Reduces more risk as magnitude of threshold term is greater
- Reduces more risk for particular sensor characteristics (Not Understood)

Disadvantage: Computationally Expensive: O(S!) terms

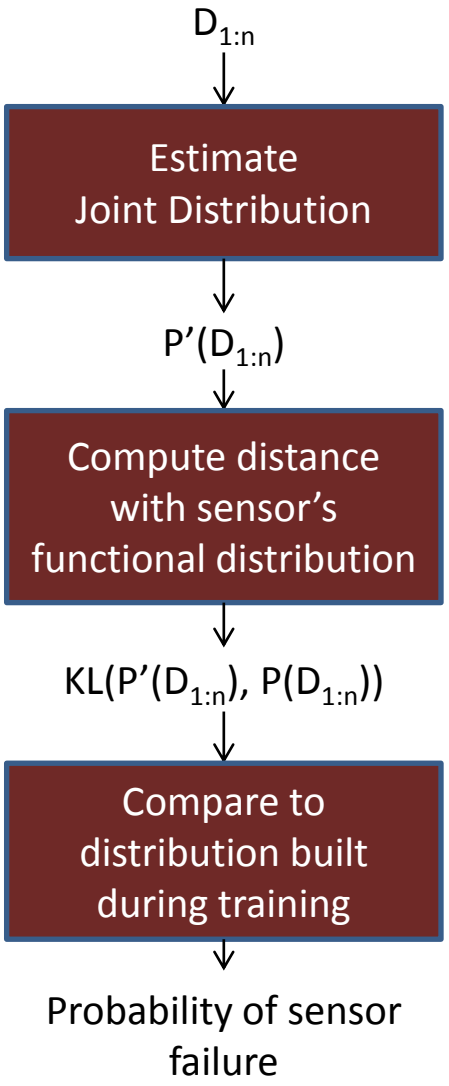
- Estimations of new fusion function are shown to fuse with little or no cost to risk
- Dynamic Programming Implementation is possible
- Implementation which takes advantage of redundancy will lower computational cost

Failed Sensor Detection

During operation, algorithm has no access to actual threat class of passengers:

- We cannot distinguish between a failure in Pd or PFA … there is no way to tell the difference
- The best, and only, way to determine sensor failure is by comparing output of sensors to each other during operation.

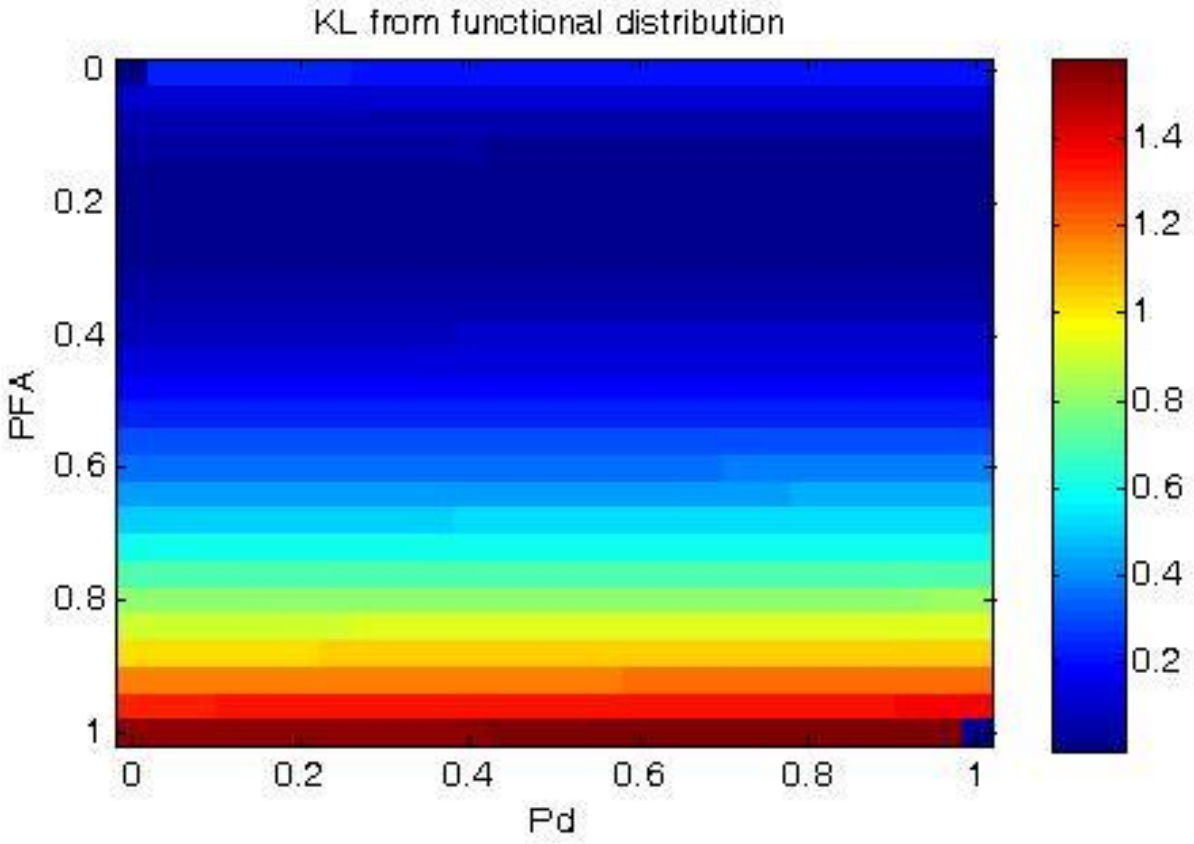
We've built a method which can detect sensor failure, relying only on relationships of sensors to each other.



Challenge: Failed Sensor Detection

Sensor failures which result in a change in PFA are easily detected (many examples).

Sensor failures which result in a change in Pd are very difficult to detect (only tested when a passenger walks through with a weapon or bomb)



Error Dependent Risk Minimization for Detection

Traditional minimum risk based "Bayesian" classifier design assumes a constant cost for each type of truth-decision pair.

The well known Neyman-Pearson approach and the likelihood ratio tests have been the resulting “optimal” detectors based on the traditional minimum risk.

Traditional minimum risk may not capture what we always care about as an optimization objective.

The risk/cost associated with a particular error outcome may depend on the circumstances at the time this error occurs.

Our sensitivity to errors might be a function of the error rate itself, hence non-constant risks must be considered to capture the inherent risk assessment values of humans that utilize and rely on these systems to make decisions or policies

Extended Expected Risk Definition

The proposed extended definition of the expected risk for an M-ary test is

$$\text{Risk}(Z) = \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} p_j \, f_{ij}(Ps) \, p_{i|j}$$

where

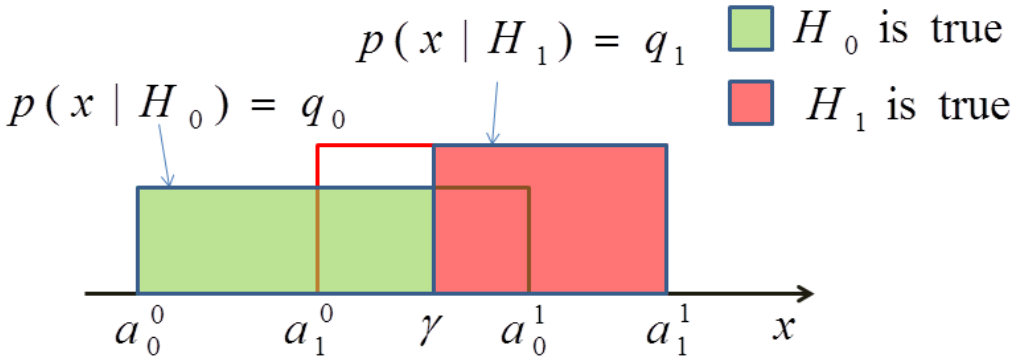
- H_i is the i^{th} hypothesis
- p_i is the a priori probability of the i^{th} hypothesis
- $p_{i|j} = \Pr(\text{decide } H_i \mid H_j \text{ is true})$ is the probability of deciding H_i while H_j is true
- $Z = \bigcup_{i=0}^{M-1} Z_i$ is the observation space
- $Ps = \{Ps_i \mid i = 0,\ldots,M-1\}$ with $Ps_i = \{\Pr(\text{decide } H_i \mid H_j \text{ is true}) \mid j = 0,\ldots,M-1\}$

Note that if $f_{ij}(Ps) = c_{ij}$ is a constant above definition reduces to the traditional expected risk.

1D Binary Hypothesis Testing Example

We demonstrate a one dimensional toy example considering a uniformly distributed class densities: $p(x|H_j)=U[a_j^0,a_j^1]=q_j$. We assume that the costs for true decisions are zero and $f_{ij}(p_{i|j})=p_{i|j}$

Solution



- Partitions of the common support: $\text{Reg}^0(\gamma)=[a_1^0,\gamma]$ and $\text{Reg}^1(\gamma)=[\gamma,a_0^1]$
- The proposed $\text{Risk}(\gamma)=p_0q_0^2(a_0^1-\gamma)^2+p_1q_1^2(\gamma-a_1^0)^2$
- The optimum $\gamma=(p_0q_0^2a_0^1+p_1q_1^2a_1^0)/(p_0q_0^2+p_1q_1^2)$
- The traditional $\text{TRisk}(\gamma)=p_0c_{10}q_0(a_0^1-\gamma)+p_1c_{01}q_1(\gamma-a_1^0)$ is a linear function of the threshold, and hence the optimum threshold can only be on the boundaries.