Bias at the Baseline: Examining Information Bias Through Entropy and Bits

Jeremy Wright

Dec,2025

Abstract

Modern information systems operate within environments of overwhelming scale, speed, and complexity. Data emerges from diverse sources at rates far exceeding the capacity of any individual observer to fully process. To manage this uncertainty, both human cognition and algorithmic machine learning systems rely on mechanisms that compress informational complexity into more predictable structures. I examine conceptual links between statistical reasoning, Bayesian inference, and lastly Shannon entropy to view how uncertainty is reduced in practice. Particular focus is given to the role of baseline assumptions, which shape inference before analysis formally begins. Bias is examined as a structural consequence of entropy reduction within current constrained systems of interpretation. As systems favor predictable patterns, information diversity declines and feedback loops emerge. The discussion refines by suggesting that maintaining some degree of high-entropy information is needed for preserving novelty, supported through proposing methods of injecting entropy into data in an adaptive way based on a defined entropy threshold for discrete data and formalizing entropy depletion as a concept.

1 Introduction: The Informational Overload Problem

There is a large pressure to greatly reduce the uncertainty found within information, which is a process that comes along with its own caveats. Information itself is foundational to modern decision making. Across disciplines such as health science, economics, social research, and computational modeling, data functions as the raw material from which patterns are extracted and predictions are formed. In principle, more information should produce clearer insights. In practice, however, the proliferation of data has created environments in which there are explosive and ever increasing rates at which data is produced, which tends to overwhelm rather than simplify. We are now immersed in informational ecosystems that generate enormous quantities of signals.

These signals may take the form of digital communication, sensor outputs, financial transactions, medical measurements, or user-generated media among others. Some signals are common, others are more rare. Each signal carries potential meaning, but interpretation requires context filtering and statistical reasoning. The challenge is bound to accumulation as well as interpretation. When information increases faster than our capacity to evaluate it, uncertainty persists even in the presence of large datasets. The modern problem is therefore a polar opposite to a scarcity of information, instead defined by the difficulty of extracting reliable structure from informational abundance. Human cognition and the systems we build to navigate information effectively refine bias into operational form. We can look at this formalization of bias in a different kind of way with the established idea of entropy given a bit further along. Information arguably is all encompassing and theories involving it can span many differing domains [1].

A useful framework for describing these conditions is the generally known concept of the three V's of data: volume, velocity, and variety. Volume refers to the sheer amount of data produced continuously across digital systems. Billions of messages, transactions, and observations are generated daily. No single individual can hope to examine such quantity. Velocity refers to the speed at which information arrives. Real-time data from financial

markets to social media feeds produce streams of data that update faster than reflective analysis can occur. Variety refers to the heterogeneity of formats through which information is expressed. Such as structured tables, unstructured text, images, audio, and mixed media coexist within the same informational ecosystem. These dimensions create a landscape in which uncertainty is unavoidable. The presence of more data does not erase ambiguity, it often amplifies it rather. The task of analysis therefore becomes one of identifying meaningful patterns within noisy, heterogeneous environments.

2 Probability and Conditional Reasoning

To understand how patterns are identified, we begin with probability, the mathematical framework for reasoning under uncertainty. Probability assigns degrees of likelihood between zero and one:

- $P(A) = 1$ if event A occurs with certainty,
- $P(A) = 0$ if impossible.

Probabilities over mutually exclusive and exhaustive outcomes sum to 1.

Most real world events fall somewhere between these extremes. A probability of 0.75 indicates that an event occurs three quarters of the time under repeated conditions. So a P of 1 would mean that outcome or observation has a 100 percent chance of occurring. All possible probabilistic outcomes examined must sum to 1 (or 100 percent) total. So, say out of three outcomes, if one outcome is .75, another is .05, then it implies the remaining outcome should be .20, (.75 + .05 + .20 = 1). Probability becomes particularly useful when events influence one another. Conditional probability describes how the likelihood of one event changes given another:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)},$$

where $P(A \cap B)$ is the joint probability.

For example, in a standard 52-card deck, let A be "drawing a heart" so $P(A) = 13/52 = 0.25$. If we know the card is red (B, $P(B) = 0.5$), then $P(A \mid B) = 0.25/0.5 = 0.5$. New information alters expected distributions. In practice, however, probability is rarely applied to perfectly controlled systems like card decks. Real-world data often involves dependencies, hidden variables, and incomplete observations. These complexities make it difficult to treat information as if it exists in isolation. This challenge leads to a deeper question related to how beliefs change when evidence accumulates. Bayesian inference provides one answer. Instead of asking only how often something occurs, Bayesian reasoning asks how strongly we should believe a hypothesis given the available evidence.

2.1 Bayesian Inference and Baseline Assumptions

Bayesian inference treats probability as a dynamic measure of belief. The central relation is Bayes' theorem:

$$P(\theta \mid D) = \frac{P(D \mid \theta) \cdot P(\theta)}{P(D)},$$

where $P(\theta)$ is the prior and the initial belief in the parameter or hypothesis, $P(D \mid \theta)$ the likelihood, or the likelihood and probability of observing the data D given the hypothesis or parameter, $P(\theta \mid D)$ the posterior, the updated belief in the hypothesis after including the observed data D, and $P(D)$ the marginal likelihood, the total probability of the data under all possible hypotheses. If components of some data in terms of probability based outcomes in total sum to under 1 or over 1, then it is likely an error has been made. The prior generally encodes baseline assumptions before observing any data. Sampling frames, model structures, or possible measurement choices are some elements that can shape expectations before analysis begins. Bias often originates here, not merely after data is seen. Bayesian inference is also an important philosophical point about knowledge. Every interpretation begins with assumptions. The prior distribution represents those assumptions explicitly.

2.2 Example Prior and Update

In Bayesian inference, we evaluate a hidden parameter θ (the state of being cold) based on observed data D (the presence of a jacket). We aim to calculate the posterior probability $P(\theta \mid D)$. The model is defined by the following probability distributions:

- Prior $P(\theta) = 0.20$: The initial baseline probability that the environment is in state θ.
- Likelihood $P(D \mid \theta) = 0.90$: The conditional probability of observing data D given the hypothesis θ is true.
- Alternative Likelihood $P(D \mid \neg\theta) = 0.10$: The conditional probability of observing data D given the hypothesis θ is false.

2.3 Calculation of the Marginal Likelihood

The evidence $P(D)$ represents the total probability of the data occurring across the entire parameter space:

$$P(D) = [P(D \mid \theta) \cdot P(\theta)] + [P(D \mid \neg\theta) \cdot P(\neg\theta)]$$

$$P(D) = (0.90 \cdot 0.20) + (0.10 \cdot 0.80)$$

$$P(D) = 0.18 + 0.08 = 0.26$$

2.4 Posterior Update

According to Bayes' Theorem, the posterior probability is the product of the likelihood and the prior, normalized by the evidence:

$$P(\theta \mid D) = \frac{P(D \mid \theta) \cdot P(\theta)}{P(D)}$$

Substituting the established values:

$$P(\theta \mid D) = \frac{0.18}{0.26} \approx 0.692$$

The integration of data D results in a significant shift in the probability mass. The shift from a prior of 0.20 to a posterior of approximately 0.692 demonstrates the inferential power of even a single high-likelihood observation.

Many discussions frame bias as post-analysis error (biased datasets or interpretations). Though, the baseline itself reflects structural assumptions. Sampling decisions, measurement techniques, or model selection all constrain what is observable or expressible. These choices are rarely malicious but do inevitably shape results. Methodological conventions (balanced sampling, randomization, double-blind studies) manage but do not eliminate bias. They define collectively tolerable limits. A dataset may be biased, a model may produce biased estimates, or a researcher may interpret results in a biased way, among numerous other possibilities. Acceptable evidence may be understood as the most stable configuration of shared baseline biases a community can sustain. They function as symbolic thresholds in some cases, designating a point beyond which we agree to treat an idea as sufficiently unbiased to act upon or accept.

Essentially no individual is likely to opt for framings or models or functions that they do not need at a given moment and will utilize that which most efficiently yields an answer or some form of statistical evidence. In that way, there is always going to be some dimension of the given data or information at hand that gets suppressed or overlooked [2] in favor of some desired outcome or threshold of evidence, such as a p-value in the case of evidence for or against a hypothesis in statistics. There may be a broad preference for a particular framing of data for instance, such as how a frequentist view generally dominates standard discourse in statistics on average compared to a Bayesian view.

3 Frequentist vs. Bayesian Views and Heuristics

Traditional statistics education emphasizes frequentist approaches, primarily involving hypothesis testing with p-values (generally < 0.05 to denote statistical significance and a value > 0.05 for non significance). These methods evaluate hypotheses by examining how frequent observations are under repeated sampling. Hypothesis testing commonly produces binary outcomes in that the results are either statistically significant or they are not. The decision is often guided by the above p-values which measure how unlikely an observation would be if the null hypothesis were true. While powerful, this framework can encourage simplified interpretations.

A result may be treated as evidence for or against a hypothesis even though uncertainty remains substantial. Frequentist methods typically avoid explicit discussion of prior beliefs. Data is analyzed as though it exists independently of historical context. In practice, however, interpretation always occurs within a conceptual framework shaped by prior expectations. Our thinking naturally gravitates toward approximated and direct conclusions. Probabilities rarely reach absolute values of zero or one, yet decisions often require categorical judgments. This connection between probabilistic reality and decisional necessity contributes to many decisions made without considering information in its broader context, but instead constrains it to what is simply needed to produce a result efficiently and quickly.

Human reasoning evolved under time and energy constraints and relies on heuristics, or in other terms, efficient shortcuts. These work well in familiar settings but introduce systematic errors like confirmation bias. In this way, as Griffiths frames, cognition and reasoning systems operate under Bayesian and predictive conditions [3]. Individuals tend to favor information that supports existing beliefs while discounting contradictory evidence. When prior beliefs are strong, new evidence may be interpreted selectively. Consider a financial analyst with a strong prior that "Forward Finance Co" is inherently stable. A minor earnings dip may be reframed as a "healthy correction," while competitor warnings are dismissed as outliers. The same 5% market drop yields different conclusions depending on the prior ("Inherent

Stability" vs. "High Risk").

Baseline assumptions filter evidence and potentially slow belief updating. This evidence can lead to different conclusions in a way that can also encourage one to develop or engineer solutions that select for or emphasize qualities or data that affirms these set notions. This could apply to many more contexts such as within research or economics and beyond. In essence, we are taking information or data (that we may or may not have a prior notion about that can impact how we interpret the evidence this data is producing) and making an effort to take a high degree of uncertainty and move towards lower degrees of uncertainty. In many cases, with the goal being to create similar and predictable outcomes that reflect very specific or desired parameters.

4 Shannon Entropy: Quantifying Uncertainty

To understand these processes more specifically, we need to quantify uncertainty. Claude Shannon introduced entropy [4] as a measure or degree of uncertainty within a probability distribution. For a discrete variable with possible outcomes. The overarching idea is that to quantify the degree of uncertainty within information we look to information based entropy. In discrete settings (countable outcomes like categories or yes/no events), entropy measures the average unpredictability of a probability distribution. Shannon entropy measures average unpredictability in a discrete distribution:

$$H(X) = -\sum_i p_i \log_2 p_i,$$

expressed in bits (log base 2).

For a fair six-sided die ($p_i = 1/6$):

$$H(X) = -6 \times \left(\frac{1}{6} \log_2 \frac{1}{6}\right) = \log_2 6 \approx 2.585 \text{ bits}.$$

Every roll delivers a lot of new information because the outcome is quite unpredictable. Now if we are to consider low entropy, then only one or a few outcomes dominate, which gives low uncertainty and produces low average surprise. An example pertaining to this can be given by a heavily biased coin For a heavily biased coin ($p(\text{heads}) = 0.99$, $p(\text{tails}) = 0.01$):

$$H(X) \approx -(0.99 \log_2 0.99 + 0.01 \log_2 0.01) \approx 0.081 \text{ bits.}$$

Almost every flip is heads, very predictable, delivering almost no new information. In data, raw inputs shaped by the previously mentioned issues with volume, velocity, and variety appear to start with high entropy, to say that the outcomes feel random and surprising. Some systems (algorithms, recommendation engines, predictive models) often work hard to reduce this entropy, pushing distributions toward concentration and predictability. If probabilities become concentrated around certain outcomes, entropy decreases. A biased coin that lands heads 99 percent of the time has very low entropy because the outcome is highly predictable. Entropy therefore measures the degree of unpredictability present within a system. If we were to consider a distribution of data, for example, a uniform distribution of all observations being equally likely would provide the most entropy, whereas a data distribution that is very tightly clustered around a narrow range of outcomes gives comparatively low entropy.

Raw data often starts high-entropy (surprising, diverse). Learning reduces entropy by concentrating probability on consistent patterns. The result is expressed in bits: a basic unit of information in computing. One bit corresponds to the amount of uncertainty resolved by a single fair yes/no question, for example. The log base 2 is so that the result can be given as bits. If all outcomes are equally likely, entropy is maximized and maximum uncertainty means there is maximum average surprise per outcome. When new information reveals consistent patterns, probability mass shifts toward the most likely outcomes.

Both human cognition and machine learning algorithms more or less act in reducing entropy in this way. Patterns are extracted from complex environments by emphasizing

regularities and suppressing unwanted noise. However, entropy reduction also introduces asymmetry. Certain outcomes receive more representational weight than others. When this concentration becomes persistent, bias emerges. In modern contexts, this reduction often takes the form of filtering mechanisms. Search engines rank results according to predicted relevance. Recommendation systems prioritize content that aligns with known preferences. Social media feeds display posts that are most likely to generate engagement. This then lowers cognitive effort for users. Instead of encountering random information, individuals see content tailored to their behavioral patterns.

The broader efficiency of such systems depends on what I refer to as entropy depletion. Predictable patterns are amplified while unexpected information becomes less visible, until we eradicate the unexpected. Although this process improves usability, it also alters the informational landscape. When entropy reduction occurs repeatedly within the same system, feedback loops can develop. At each iteration, the system favors patterns that previously generated strong responses. As distributions narrow, novelty declines. Users encounter fewer unfamiliar perspectives and information becomes more predictable but less diverse. Present findings convey that we can still find model efficiency and success even with the elements of randomness and unpredictability [5].

4.1 Entropy Depletion and Systemic Bias

As stated, modern systems (search engines, recommenders) reduce entropy to deliver predictable content. Repeated depletion creates feedback loops: familiar patterns quickly amplify, novelty sharply declines and can give rise to filter bubbles [6] and echo chambers via engagement optimization. The concept of entropy is usually discussed in other contexts like within physical energy systems (such as physics and chemistry), where entropy typically increases over time and there is no process to reverse or undo it. Energy disperses, and disorder grows. In a slight contrast, informational systems operate differently. While the raw generation of data may increase global entropy, interpretive systems actively reduce

entropy to produce coherent patterns. In information systems, there does not seem to be an understood "gradual increase" in entropy in the same way, but with all the information produced daily, it would not seem illogical to assume information based entropy is also in a state of continuous increase.

There is a deliberate goal to engineer decreasing entropy in information systems. We want order, predictability, and concentration of probability. While low entropy brings efficiency and clarity, over-optimizing for predictability sacrifices discovery. The novelty that high entropy provides can get lost in that efficiency and lead to prioritizing only the familiar and not the unfamiliar. Thus one goal would be to make more individuals aware of the value of uncertainty and high entropy and to adopt additional ways of introducing more entropy, if information has become too redundant. So generally, while physical entropy increases, interpretive information systems deliberately decrease it.

Systems that compress information too aggressively risk becoming compromised. They become highly efficient within familiar conditions but struggle to respond to unfamiliar ones. This can in effect be applied to the concept of overfitting, where algorithm or model training becomes so efficient so as to capture highly specific parameters about a given data set but cannot effectively generalize beyond that. In that sense, I also argue for more consistent and intentional design choices that preserve some level of informational diversity. Before presenting a personally proposed method of "injecting" entropy into data, I will present sample outputs from both Python and R to illustrate how both the features and shape of data can lead to certain observations or outcomes to be far more likely and how to observe when entropy is either being gained or lost in the process. With the R programming language, there is a special focus on statistics. I will apply it for the goal of showing how distribution shape conveys information entropy in a direct way.

As a disclaimer, Shannon entropy specifically pertains to discrete observations rather than continuous observations, and is appropriate for application within machine learning. This involves countable outcomes that are specific and cannot take on a given range of

intermediate values, which would relate to a metric like temperature. While there is a specific method to look at entropy in a continuous sense, it moves beyond the goals presented thus far and will not be utilized. I use a basic assumption of simple random sampling for a given A/B classification set up, so as to illustrate how minority classes in data get omitted often at random and how we can apply a different look at how we might use Shannon entropy to see when we are gaining no information from the minority class, in the code below, denoted as D.

Class imbalance naturally does not often align well with random sampling, however in this case it is illustrative. In this set up, the data is heavily biased in favor of the dominant class, M 95 percent of the time and only in favor of class D 5 percent of the time. In this context, it is very likely to get a sampling output that yields no minority (rare) cases. I apply 5,000 trials of a sample size n=10. It could very well be that in such a case one would see 10 of the dominant class and 0 of the minority class out of that sample size. This means that class features are essentially erased and a model gains nothing out of it whatsoever. There are ways to "make" an example of all classes appearing in an output through stratified sampling though the goal here is to show how random observations interact with entropy.

4.2 Illustrative Simulations: (Python) Minority Class Loss and (R) Distribution

Shannon entropy can be applied as a formula in a program such as python through using it as given prior, with its log base 2 component (discreet observations). Consider a population 95% majority class M, 5% minority D. Simple random sampling ($n = 10$, 5000 trials) often yields zero D instances erasing minority information and driving observed entropy near zero. For example, a case of 10 observations of M and 0 observations of D would produce no entropy for that sampling outcome because M would be observed with total certainty.

Feature dropout is a regularization technique where input features are randomly removed during training so that a model does not become overly dependent on any single piece of

information. I argue it can be used with respect to entropy to set a parameter for when a feature is dropped. This is typically implemented by generating a binary mask, a vector of 0s and 1s modeled from a Bernoulli distribution, and multiplying it element wise with the feature vector. A value of 1 keeps the feature, while 0 removes it for that training step. When integrating this idea with an entropy based control mechanism, entropy can be used as a measure of uncertainty or diversity in the feature distribution. If the entropy of the system falls below a chosen threshold indicating the model may be relying too heavily on a narrow set of predictable features, the function can activate dropout by applying the binary mask to randomly remove some features. If entropy remains above the threshold, dropout can be skipped so the full feature set is preserved. In this way, the binary mask acts as the operational mechanism for feature removal, while entropy serves as the decision parameter that determines when dropout should be applied to maintain informational diversity.

A Bernoulli distribution describes a random process with only two possible outcomes, usually coded as 1 (success) and 0 (failure), where the probability of success is p and the probability of failure is $(1 - p)$. In the context of feature dropout, a Bernoulli distribution is used to generate a binary mask: for each feature, a random draw is made that outputs 1 (keep the feature) with probability p, or 0 (drop the feature) with probability $(1 - p)$. This is relevant to Shannon entropy because the Bernoulli distribution is the simplest system whose uncertainty can be measured with entropy. When the probabilities of the two outcomes are similar, entropy is higher (more uncertainty); when one outcome dominates, entropy is lower (more predictability). The sample code I have made here connects to these concepts.

4.3 A Basic Python Demonstration (One Sample):

```
1 import numpy as np
2 import random
3 import matplotlib.pyplot as plt
4 from collections import Counter
```

```

random.seed(42)
np.random.seed(42)

data = ['M'] * 70 + ['D'] * 30  # 70/30 split for illustration

sample = random.sample(data, 10)
```

Listing 1: I define a Shannon Entropy function to return bits

```
def shannon_entropy(probabilities):
    p = np.array(probabilities)
    p = p[p > 0]
    return -np.sum(p * np.log2(p))  # bits
```

Listing 2: Define function for feature dropout

```
def apply_feature_dropout(sample_size=10, rate=0.3):
    mask = np.random.binomial(1, 1-rate, sample_size)
    return mask # This gives a binary mask, so if rate=0.3, there's a
        30% chance of dropping (mask=0) and 70% chance of keeping (mask
        =1)
```

Listing 3: Parameters for experiement

```
# Set Parameters
trials = 5000
n = 10

# If entropy < 0.4 (context specific), the sample is considered
    strongly biased toward the dominant class (M) relative to the
    minority class (D).
```

```
entropy_threshold = 0.4

all_entropies = []
interventions = 0

# Running the Experiment
for _ in range(trials):

    s = random.sample(data, n)

    # Calculate empirical probabilities
    p_M = s.count('M') / n
    p_D = s.count('D') / n

    # Compute Shannon entropy for this sample
    h = shannon_entropy([p_M, p_D])
    all_entropies.append(h)

    # Trigger intervention if entropy falls below threshold
    if h < entropy_threshold:
        interventions += 1

        # In a full model this would apply dropout to the features
           associated with these observations
        _ = apply_feature_dropout(n)
```

Listing 4: Entropy-controlled summary statistics

```
Average Entropy across 5000 trials: 0.8140 bits
Number of interventions triggered: 118 out of 5000 trials
```

```
Intervention Rate: 2.36%
```

Here we just see an entropy "injection" threshold where if entropy from one of the trial outputs falls at or below it, then dropoff gets used to remove it to keep instances wherein there is very high bias for the dominant class M over the minority class D. We can see that of all trials, 118 fell into this threshold or 2.36 percent of all trials. The parameter of 0.4 could be increased for a less stringent threshold depending on how much bias towards one class is going to be tolerated. I emhasize that there are methods of regularization that can manage class imbalances, however they generally do not appear to emphasize using calculated entropy as the primary target, this this is again, illustrative of that interest. This will now be elaborated upon briefly.

4.4 Considering a Partially Pooled Entropy Management in Python

Below is an illustrative manual example of considering partial pooling logic with respect to entropy outcomes between groups. In partial pooling settings, the group with a smaller n size is impacted more by this process than the group with a larger n size. This effect is relative, and could also potentially "inject" entropy into a smaller group. The concept being that you can see what happens if you amplify the entropy signal to a portion of your overall data. A global metric such as population mean guides this process normally, though here the goal is to pull towards probability distributions and then examine entropy. As prior, I already defined the entropy function, however pre-existing packages in Python, such as in scipy, can have a ready to use entropy calculation without computing in manually.

```

# Partial pooling of probability distributions
def partial_entropy_normalization(p_group, p_global, n_group, k=10):
    w = n_group / (n_group + k)
    p_new = w * np.array(p_group) + (1 - w) * np.array(p_global)
    return p_new
```

```
7
8  # Example distributions
9  p_global = [0.5, 0.3, 0.2]
10 p_small  = [0.9, 0.1, 0.0]  # small sample group
11 p_large  = [0.9, 0.1, 0.0]  # same distribution but large n
12
13 # Apply partial pooling logic given entropy
14 p_small_new = partial_entropy_normalization(p_small, p_global, n_group
       =5)
15 p_large_new = partial_entropy_normalization(p_large, p_global, n_group
       =100)
16
17 # Entropy comparison
18
19
20 # Group differences np array
21 differences_small = np.array(p_small_new) - np.array(p_small)
22 differences_large = np.array(p_large_new) - np.array(p_large)
23
24 # I check the difference in bits within both groups by subtracting the
       original entropy from the new entropy after partial pooling.
25 entropy_difference_small = entropy(p_small_new) - entropy(p_small)
26 entropy_difference_large = entropy(p_large_new) - entropy(p_large)
```

Listing 5: Above is the partial pooling logic and below is the bit outputs

The outcome between the groups before and after is as such:

Small group entropy (before): 0.4690 bits

Small group entropy (after): 1.2948 bits

Large group entropy (before): 0.4690 bits

Large group entropy (after): 0.6519 bits

Entropy difference for small group: 0.8258 bits

Entropy difference for large group: 0.1829 bits

We can see the smaller group has a comparatively larger increase while larger group has smaller change in entropy, reflecting partial pooling logic where smaller groups are more influenced by the global distribution.

4.5 Example R Output Distribution Shape

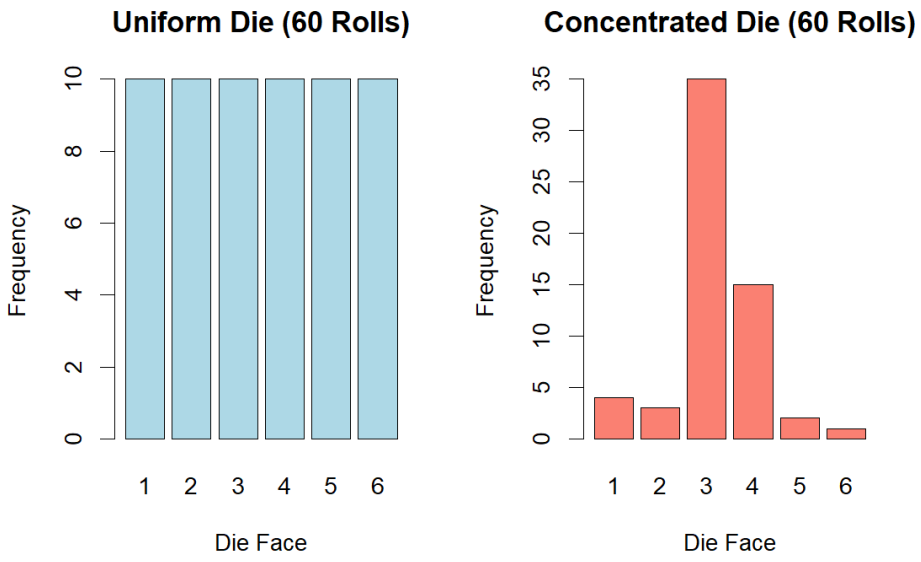

Figure 1: Distribution Shape Conveys Entropy.

It can be briefly shown here how distribution shape can naturally convey something about the present entropy. Consider a basic experiment of tossing normal die with 6 faces a

total of 60 times. A uniform deistribution of the uniform die example, has each outcome be equiprobable and therefore produces the maximum possible entropy of that configuration as a result. We also potray the reduction of entropy through looking at the concentrated die visual. The results are far more tightly clustered around a dominant outcome and therefore observations are more certain, meaning that entropy has collapsed rather than increased. Kurtosis can show us what to expect about entropy as well, as a negative kurtosis value relates to platykurtic distribution shape (flatter) meaning results are spread more evenly. If positive, then a leptokurtic (tall) shape tightens the results. This is to show obvious changes in entropy as a mesokurtic and normal distribution may not convey this as well.

Respective kurtosis values for the uniform and concentrated examples: -1.325 and 1.551.

5 Entropy Thresholds

I would consider that the core distinction between this entropy driven approach and established methods like SMOTE lies in the signal preservation goal. For SMOTE as a regularization too, (Synthetic Minority Over Sampling Technique) when a dataset is lopsided [7]. For instance, some medical data having thousands of examples of healthy cells but only five examples of cancerous ones, a computer model will likely just ignore the rare cases often. To fix this, you can look at those five rare points on a graph and find the ones that are most similar to each other. Instead of just making exact copies of those five points (which doesn't teach the computer anything new), this technique draws imaginary lines between them. It then drops brand-new, synthetic data points anywhere along those lines.

By creating these "in-between" versions of what you already have, you're essentially magnifying the minority group. This forces the computer to pay attention to the rare cases by making that part of the map look much more crowded and detailed than it actually was. While SMOTE is designed to mitigate class imbalance through the interpolation of points effectively inventing samples to satisfy a model's need for data volume, my consideration of

this entropy injection mechanism functions as an information-theoretic control. As stated prior, the challenge in modern systems is a collapse of informational diversity during the process of uncertainty (entropy) reduction.

The value of the 0.4 bit entropy threshold serves as a context-specific intervention rather than an arbitrary heuristic. In a binary classification system, the maximum possible entropy is exactly 1 bit, which occurs when the class distribution is perfectly balanced (50–50). Setting the threshold at 0.4 bits therefore establishes a normalized diversity floor, which could be presented as:

$$H_{\text{threshold}} = 0.4 \times H_{\max},$$

this implies that any batch or sample where the informational "surprise" falls below 40% of its theoretical maximum, corresponding to a strongly skewed class distribution, is considered too compressed to provide a meaningful learning signal. In such cases, the system may trigger an intervention, such as feature dropout, in order to reintroduce variability.

Split (M/D)	Probability of M (p)	Entropy (bits)
10 / 0	1.0	0.000
9 / 1	0.9	0.469
8 / 2	0.8	0.722
7 / 3	0.7	0.881
6 / 4	0.6	0.971
5 / 5	0.5	1.000

Table 1: Possible entropy values for all class splits when the sample size is $n = 10$.

Table 1 shows the possible entropy values for all class distributions when the sample size is $n = 10$. Because the sample size is small, the class probabilities can only change in increments of 0.1, meaning that the entropy values are discrete rather than continuous. In this setting, the entropy of a 9/1 split is approximately 0.469 bits, which is already greater than the intervention threshold of 0.4 bits. As a result, the only configuration whose entropy falls below the 0.4 threshold is the extreme case of 10/0, where one class completely dominates and the entropy collapses to 0. Consequently, across the 5000 simulated trials, the intervention

is triggered only when the sampled batch contains no observations from the minority class. This outcome reflects a property of the small sample size rather than a limitation of the entropy criterion itself. Given that I want to only remove the most biased extreme case here, this is not problematic in this context.

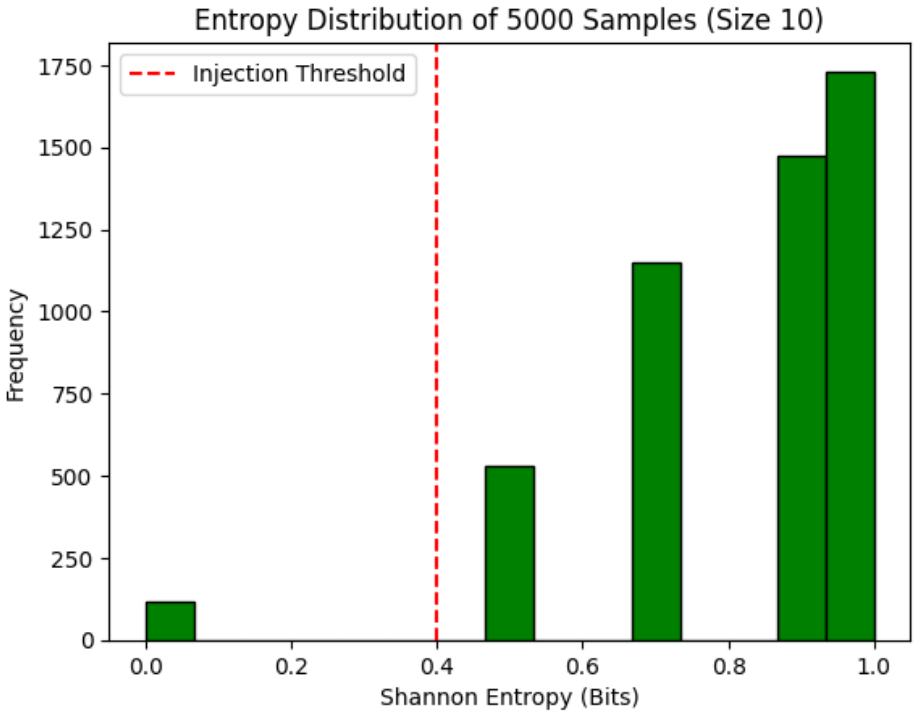

Figure 2: Visual of Previous Entropy Cutoff (0.4).

5.1 Scaling the Entropy Threshold to Multi-Class Systems

The entropy threshold used for intervention can be generalized to systems with more than two outcomes by scaling it relative to the maximum possible entropy of the system. For a categorical variable with k possible outcomes, the maximum entropy occurs when all outcomes are equally likely and is given by

$$H_{\max} = \log_2(k).$$

An intervention threshold can then be defined as a fraction of this maximum entropy:

$$H_{\text{threshold}} = \alpha H_{\max}, \quad 0 < \alpha < 1.$$

This formulation allows the threshold to maintain consistentcy across different problem sizes. For example, when $\alpha = 0.4$, the intervention is triggered whenever the observed entropy falls below 40% of the system's theoretical maximum informational diversity.

5.2 Example: Six-Sided Die

Consider again a six-sided die, where $k = 6$. The maximum entropy is therefore

$$H_{\max} = \log_2(6) \approx 2.585 \text{ bits}.$$

Using $\alpha = 0.4$, the intervention threshold becomes

$$H_{\text{threshold}} = 0.4 \times 2.585 \approx 1.034 \text{ bits}.$$

This means that if the observed distribution of outcomes becomes concentrated such that the entropy drops below approximately 1.03 bits, the system may trigger an intervention (e.g., feature dropout) to restore informational diversity.

5.3 Illustrative Entropy Values

Outcome Distribution	Dominant Probability	Entropy (bits)
Uniform (1/6 each)	0.167	2.585
Two faces dominant (0.45, 0.45, others small)	0.45	∼1.47
One face dominant (0.70, others small)	0.70	∼1.16
One face highly dominant (0.85, others small)	0.85	∼0.80
Single outcome only	1.00	0.000

Table 2: Example entropy values for different outcome concentrations in a six-sided system.

Defining the threshold as a fraction of $H_{\max}$ allows the intervention rule to scale naturally across binary, multi-class, or higher-dimensional categorical systems while maintaining a

consistent interpretation in terms of informational diversity.

6 Entropy Depletion and Iterative Bias Growth

In complex information systems entropy is the reliable measure of uncertainty. Systems process information iteratively (repeated sampling or learning steps), patterns are extracted, probabilities concentrate, and entropy is gradually consumed. When entropy is lost, a system becomes more predictable, and bias toward dominant outcomes grows. Below is a basic formalization of my concept of entropy depletion, described earlier. I introduce **baseline bias** as the fraction of a distribution already skewed toward the dominant outcome before any iterations. For example, if one outcome occurs 40% of the time initially, the baseline bias is $B_0 = 0.4$.

6.1 Baseline Bias

Let

$$B_0 \in [0, 1]$$

be the baseline bias, the fraction of the distribution favoring the dominant outcome at the start. This is important to establish given that the reference to bayesian priors motivates the idea of having an established beleif or bias about some given or possible outcome.

6.2 Entropy as a Depletable Resource

Let H_0 be the maximum possible entropy for the system, and H_i the remaining entropy after iteration i. Entropy decays with each iteration according to an exponential depletion process:

$$H_i = H_0\, e^{-\lambda i}, \quad i = 0, 1, 2, \ldots$$

where $\lambda > 0$ controls the rate of entropy loss per iteration. At $i = 0$, H_0 is fully available, and as i increases, $H_i \to 0$.

6.3 Bias Growth Due to Entropy Depletion

As entropy is consumed, bias toward the dominant outcome grows. Let the additional bias be proportional to the entropy lost:

$$B_{\text{add}}(i) = \gamma \left(H_0 - H_i \right)$$

where γ is the *bias amplification factor*, describing how strongly lost entropy translates into increased dominance. The total bias at iteration i becomes:

$$B_i = \min \left(1,\ B_0 + \gamma H_0 \big(1 - e^{-\lambda i} \big) \right)$$

At $i = 0$, B_0 represents the starting skew. As $i \to \infty$, bias approaches $B_0 + \gamma H_0$, capped at 1.

6.4 Illustrative Example: Six-Sided Die

Consider once more a six-sided die with initially equal probabilities. Maximum possible entropy is:

$$H_0 = \log_2(6) \approx 2.585 \text{ bits}$$

Assume the following parameters:

- Baseline bias: $B_0 = 0.4$ (one face initially favored 40%)
- Entropy depletion rate: $\lambda = 0.5$ per iteration
- Bias amplification factor: $\gamma = 0.3$

After iteration $i = 2$:

$$H_2 = 2.585 \cdot e^{-0.5 \cdot 2} \approx 0.951$$

$$B_2 = 0.4 + 0.3 \cdot (2.585 - 0.951) \approx 0.4 + 0.490 = 0.89$$

After two iterations of processing, 89% of the distribution is biased toward the dominant face, illustrating how entropy depletion amplifies existing skew.

6.5 Iterative Example Table

Using the parameters from the six-sided die example:

$$H_0 = 2.585, \quad B_0 = 0.4, \quad \lambda = 0.5, \quad \gamma = 0.3$$

The entropy and bias evolve over iterations as follows:

Iteration i	Remaining Entropy H_i	Additional Bias $B_{\text{add}}(i)$	Total Bias B_i
0	2.585	0.0	0.40
1	$2.585\, e^{-0.5 \cdot 1} \approx 1.567$	$0.3 \cdot (2.585 - 1.567) \approx 0.305$	0.705
2	$2.585\, e^{-0.5 \cdot 2} \approx 0.951$	$0.3 \cdot (2.585 - 0.951) \approx 0.490$	0.89
3	$2.585\, e^{-0.5 \cdot 3} \approx 0.577$	$0.3 \cdot (2.585 - 0.577) \approx 0.603$	1.0 (capped)
4	$2.585\, e^{-0.5 \cdot 4} \approx 0.350$	$0.3 \cdot (2.585 - 0.350) \approx 0.682$	1.0 (capped)

Notes:

- H_i follows exponential depletion: $H_i = H_0 e^{-\lambda i}$
- Additional bias $B_{\text{add}}(i) = \gamma(H_0 - H_i)$
- Total bias $B_i = \min(1, B_0 + B_{\text{add}}(i))$
- Bias saturates at 1 when cumulative skew reaches the maximum possible.

6.6 Choosing Bias Amplification Factor

The bias amplification factor, γ, determines how strongly lost entropy translates into additional bias toward the dominant outcome. Conceptually, it captures the sensitivity of the system to entropy depletion: a larger γ implies that even small entropy losses rapidly increase bias, whereas a smaller γ produces a more gradual growth in skew. The choice of γ can be guided by several considerations. Theoretically, it reflects how rigid or flexible the system is expected to be. Empirically, it can be calibrated through simulations to achieve realistic or interpretable dynamics. Additionally, γ may be normalized to prevent the total bias from exceeding its maximum, e.g., $\gamma \leq \frac{1-B_0}{H_0}$ ensures that $B_i \leq 1$ even when entropy is fully depleted. In practice, adjusting γ allows the modeler to explore systems that are either robust to entropy loss or highly sensitive to initial skews.

6.7 Example Application of γ

For instance, with reference to the die with a baseline bias of $B_0 = 0.4$ and maximum entropy $H_0 \approx 2.585$ bits. Choosing a relatively high bias amplification factor, $\gamma = 0.3$, means that after just two iterations of entropy depletion (with $\lambda = 0.5$ per iteration), the additional bias grows substantially:

$$B_2 = \min\left(1, B_0 + \gamma H_0(1 - e^{-\lambda \cdot 2})\right) \approx 0.89.$$

In contrast, selecting a smaller $\gamma = 0.1$ would produce a more gradual increase, yielding $B_2 \approx 0.63$. This expresses how γ directly modulates the sensitivity of the system to entropy loss: higher values rapidly amplify dominance, while lower values allow skew to accumulate more slowly. One can adjust γ depending on whether they want to explore scenarios where existing biases become pronounced quickly or more resiliently over multiple iterations.

6.8 Coded Example

```
import numpy as np

# Parameters
sides = 6
iterations = 8
gamma = 0.3
lam = 0.5
B0 = 0.4

H0 = np.log2(sides) # Theoretical maximum entropy for a fair die


for i in range(iterations):

    Hi = H0 * np.exp(-lam * i) # Entropy depletion over iterations

    B_add = gamma * (H0 - Hi) # Bias amplification based on entropy
        depletion

    bias = min(1.0, B0 + B_add)
```

The print instructions were truncated out for clarity and the output mirrors what was shown above, that with the same parameters we quickly deplete our entropy and cap out at a maximum bias towards a dominant probability meaning entropy collapses to 0 at the instance of this occuring.

The output:

```
iter | H_i(bits) | dominant_prob
-----------------------------------
   0 |     2.585 | 0.400
   1 |     1.568 | 0.705
   2 |     0.951 | 0.890
   3 |     0.577 | 1.000
   4 |     0.350 | 1.000
   5 |     0.212 | 1.000
   6 |     0.129 | 1.000
   7 |     0.078 | 1.000
```

7 Implications, Challenges and Conclusion

The central claim of this paper was that systemic information bias can be understood as an emergent property of repeated entropy depletion within interpretive systems. Looking forward, I would suggest that the broader utility of this framework lies in the development of additional entropy-centric regularization tools. This is supported through the synthesis of established ideas from outside works of research in an effort to promote entropy as a baseline for how we discuss, frame, or model information and bias.

If we consider that regularization is a fight against overfitting (or entropy collapse), then explicitly coupling dropout rates or weight decay to real time entropy measurements could offer a more homeostatic approach to training. In constraining the focus to structured data for the moment, we can more clearly observe how this diversity floor connects to calculated entropy. Primarily, this suggests that for a system to remain capable of discovery, it must be intentionally designed to reject excessive predictability, ensuring that the unexpected is never entirely eradicated by the prevailing efforts in reducing entropy across countless information

or data sources. This can also in a number of ways, encourage less of a cognitive prefernce for certainty in information and encourage more of a preference towards maintaining novel or suprising signals from data. The transparency of these signals matters in terms of how they are derived or constrained so as to limit black-box or obsure outcomes [8] so they may be understood or replicated.

Additionally, having to compute entropy with respect to other parameters or features could in some cases be more computationally expensive than it otherwise needs to be. There are already numerous techniques or regularization tools that can properly handle atypical distributions or feature erasure effectively. Though, I again state that entropy being the guiding metric for how we approach bias should be given more emphasis, being considered alongside features that engineer solutions tailored to minimizing entropy on a widely adopted scale. This synthesis of concepts extended by bounding entropy to bias and further proposed applications of this, and should ideally serve to motivate additional solutions or discussion of how to best balance bias and certainty versus uncertainty in our information landscape.

References

[1] D. J. C. MacKay. *Information theory, inference, and learning algorithms*. Cambridge University Press, 2003.

[2] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan. A survey on bias and fairness in machine learning. *ACM Computing Surveys*, 54(6):1–35, 2021.

[3] T. L. Griffiths. Bayesian models of cognition. In M. C. Frank and A. Majid, editors, *Open Encyclopedia of Cognitive Science*. MIT Press, 2024.

[4] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:623–656, 1948.

[5] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

[6] E. Pariser. *The filter bubble: What the Internet is hiding from you*. Penguin Press, 2011.

[7] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[8] R. Shwartz-Ziv and N. Tishby. Opening the black box of deep neural networks via information. *arXiv*, 2017.