Statistical Learning Theory

Doruk Çetin

September 13, 2019

Contents

1 Introduction

- Algorithm is essentially the model.
- Correctness (of an algorithm that outputs random variables) is a scientific problem.
- Information theory point of view: generalization capacity.
- We should give back to our customers not individual answers but distributions over answers. Quantifying uncertainty is important.
- Given the posterior, Bayes classifier is optimal for the classification problem. It says that one must take the class for which the probability of the class given the observation is maximal among your choices and it is not so low that you are in doubt.
- Maximum likelihood optimization will at worst give you an *atypical* solution.

Estimating the probability distribution of the data might be a much harder problem than estimating the probability distribution of the solutions given the data, because the data usually live in a much higher dimensional space than the output solutions. That's the whole purpose, we want to get rid of degrees of freedom.

Foundation of Robust Algorithm Design

- Input inconsistency and error tolerance: Algorithms should robustly process stochastic inputs and should be able to tolerate "moderate" soft error rates. Bayesian approach is robust in the sense that it robustifies solutions by iterating over models.
- Algorithm validation: Validation of cost functions and algorithms by information theory. What is generalization in this context? There are also other inference algorithms that are not gradient descent on some cost functions. If that's the case, we need a concept for validation that does not relate on cost functions.
- The goal is self improving algorithms that model reality automatically in a data-driven fashion.

How can we validate (data science) algorithms?

- Kolmogorov: Algorithms with random variables as input compute random variables as output! (How can we prove correctness of such algorithms?)
- Shannon: Algorithms must compute typical solutions! (What does this mean for algorithm design?)
- Vapnik: Algorithms have to generalize over noise/model mismatch! (What does generalization mean for algorithms?)

In Data Analysis, a big problem is to separate signal from the noise. When separating signal from noise, you can rely on the properties of the random variables. An even bigger problem is to separate a relevant signal to solve a task from the irrelevant signal which is also there but which I do not want to pay attention to. This is the key idea behind the control experiments (i.e. making $L = 2$).

Machine learning is not optimization!

- What you should do, cannot be done, since we don't know $P(\mathbf{X})$!

$$\text{Calculate } c^{\perp} \in \arg\min_{c \in \mathcal{C}} \mathbb{E}_{\mathbf{X}} R(c, \mathbf{X})$$

- What you can do, is not most relevant, since it might yield atypical solutions.

$$\text{Calculate } \hat{c}(\mathbf{X}') \in \arg\min_{c \in \mathcal{C}} R(c, \mathbf{X}') \text{ or } \hat{c}_L(\mathbf{X}') \in \arg\min_{c \in \mathcal{C}} \frac{1}{L} \sum_{l=1}^{L} R(c, \mathbf{X}^{(\mathbf{l})})$$

- Machine Learning algorithms localize solutions $c \sim P_\theta(c \mid \mathbf{X}')$. Note that we have to validate the metric of the solution space $\mathcal{C}$.

We should not do optimization but proper localization which goes hand-in-hand with approximation. Approximations are your ticket to calculating atypical solutions, if done right. Regularization, for example, is a step towards getting more stable solutions.

Zero covariance only indicates statistical independence for Gaussian random variables.

Glivenko-Cantelli is true for distributions but not for densities. There is not a corresponding result for densities, without any additional assumptions.

$$\mathbf{P}\left\{ \sup_{x \in \mathbb{R}} |F(x) - \hat{F}_n(x)| \xrightarrow{n \to \infty} 0 \right\} = 1$$

1.1 Basics

- Gaussian integral:

$$\int_{-\infty}^{\infty} e^{-a(x+b)^2} dx = \sqrt{\frac{\pi}{a}}$$

- Sum of normally distr. variables $X \sim \mathcal{N}(\mu_x, \Sigma_x)$, $Y \sim \mathcal{N}(\mu_y, \Sigma_y)$:

$$X + Y \sim \mathcal{N}(\mu_x + \mu_y, \Sigma_x + \Sigma_y)$$

- Affine transformation of multivariate normal distribution $X \sim \mathcal{N}(\mu, \Sigma)$:

$$Y = c + BX \sim \mathcal{N}(c + B\mu, B\Sigma B^T)$$

- Log of clustering posterior:

$$-\log p(c; X, \theta) = \beta R + \log \mathcal{Z}$$

2 Maximum Entropy Inference

Entropy is expected surprise. Surprise $s : [0, 1] \to [0, \infty)$ measure how informative events are. Given are events $X \in \mathcal{X}$ with probability $p(X)$, surprise can be formalized by **four axioms** (Shannon's solution of the fundamental properties of information):

- Certainty: $S(1) = 0$, no surprise for certain events
- Anti-monotonicity: $p \leq q \implies S(p) \geq S(q)$
- Continuity: $S(p)$ is a continuous function of p, no surprise jumps for infinitesimal p changes
- Additivity: $S(p \cdot q) = S(p) + S(q)$

Jaynes' philosophy: Find the density $p(x)$ which fulfils the constraints (given by expectations values) but is maximally undefined (noncommital) otherwise. In a certain sense we look for the distribution which obeys the measurement constraints but otherwise is most similar to the uniform distribution (i.e. maximizes entropy or differential entropy).

$$\textbf{Gibbs distribution: } p(x) = \frac{\exp(-\sum_j \lambda_j r_j(x))}{\sum_{x' \in \mathcal{X}} \exp(-\sum_j \lambda_j r_j(x'))}$$

For a fragile $r(x)$ you would have to use a low λ and for a very robust $r(x)$ you could probably live with a very large λ.

Maximum entropy examples [link]:

- The uniform distribution on the interval $[a, b]$ is the maximum entropy distribution among all continuous distributions which are supported in the interval $[a, b]$, and thus the probability density is 0 outside of the interval.
- Maximum entropy distribution for non-negative r.v. with mean μ (Exponential):

  $$p(x) = \frac{1}{\mu} \exp(-\frac{x}{\mu})$$

  (among all continuous distributions supported in $[0, \infty]$)
- Maximum entropy distribution with zero mean and fixed variance (Gaussian):

  $$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{x^2}{\sigma^2})$$

  (among all continuous distributions supported in $[-\infty, \infty]$)
- Laplace distribution maximizes entropy when the conditional distribution assumes a different variance in each experiment.

  $$p(x) = \frac{1}{2\Delta} \exp(-\frac{|x|}{\Delta})$$

  (among all continuous distributions supported in $[-\infty, \infty]$)

2.1 Properties

Properties of entropy:

- Definition: $H(X) = -\sum_{i=1}^{n} P(X_i) \log P(X_i)$
- Symmetry: $H_n(p_1, p_2, ...) = H_n(p_2, p_1, ...)$

- Chain rule: $H(X,Y) = H(X|Y) + H(Y) = H(Y|X) + H(X)$
- $H(X,Y) \leq H(Y) + H(X)$, equality holds when X and Y are independent
- Cross-entropy: $H(p,q) = -\sum_{x\in\mathcal{X}} p(x) \log q(x)$
- Conditional entropy: $H(x \mid y) = -\sum_{i,j} p(x_i, y_j) \log \frac{p(x_i,y_j)}{p(y_j)}$

Properties of mutual information:

- Non-negativity: $I(X;Y) \geq 0$
- Symmetry: $I(X;Y) = I(Y;Y)$
- $I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$
- $I(X;Y) = H(X,Y) - H(X|Y) - H(Y|X) = H(Y) + H(X) - H(X,Y)$
- Conditional mutual information: $I(X;Y|Z) = H(X|Z) - H(X|Y,Z)$
- Chain rule for mutual information: $I(X;Y,Z) = I(X;Z) + I(X;Y|Z)$

Properties of Kullback–Leibler divergence:

- KL-divergence (relative entropy): $D_{KL}(P(x) \parallel Q(x)) = -\sum_{x\in\mathcal{X}} P(x) \log(\frac{Q(x)}{P(x)})$
- Non-negativity: $D_{KL}(P(x) \parallel Q(x)) \geq 0$
- $I(X;Y) = D_{KL}(P(X,Y) \parallel P(X)P(Y))$
- $H(p,q) = H(p) + D_{KL}(p \parallel q)$ implies $H(p) \leq H(p,q)$

2.2 Information theory recap

- In a straightforward representation, $\log_2(n)$ bits are needed to represent a variable that can take one of n values if n is a power of 2. If these values are equally probable, the entropy (in bits) is equal to n.
- p is the target and q is the approximation distribution.

$$D_{KL}(p(x) \parallel q(x)) = \sum_x p(x) \log \frac{p(x)}{q(x)}$$

- Kullback–Leibler divergence penalizes approximations that underestimate likely events.
- Entropy of X $\approx$ uncertainty of X $\approx$ amount of information of X $\approx$ minimum expected number of questions to guess X
- In **Forward KL**, the difference between $p(x)$ and $q(x)$ is weighted by $p(x)$. During the optimization process then, whenever $p(x) = 0$, $q(x)$ would be ignored. The difference between $p(x)$ and $q(x)$ will be minimized if $p(x) > 0$. It is known as **zero avoiding**, as it is avoiding $q(x) = 0$ whenever $P(x) > 0$.

$$p_{\mu,\sigma} \text{ as a mass-seeker: } \arg\min_{p_{\mu,\sigma}(x)} D_{KL}(p \parallel p_{\mu,\sigma})$$

- In **Reverse KL**, as we switch the two distributions' position in the equation, now $q(x)$ is the weight. Here, it is better to fit just some portion of $p(x)$, as long as that approx-

imate is good. Consequently, Reverse KL will try avoid spreading the approximate. As those properties suggest, this form of KL Divergence is know as **zero forcing**, as it forces $q(x)$ to be 0 on some areas, even if $p(x) > 0$.

$$p_{\mu,\sigma} \text{ as a mode-seeker: } \arg\min_{p_{\mu,\sigma}(x)} D_{KL}(p_{\mu,\sigma} \parallel p)$$

- Non-negativity of D_{KL} from Gibbs' inequality:

$$-\sum_{i=1}^{n} p_i \log p_i \leq -\sum_{i=1}^{n} p_i \log q_i \implies \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i} \geq 0$$

3 Maximum Entropy Clustering

- There is a tradeoff between richness (informativeness) of a solution and its stability.
- The way we can build up cost functions is a rich repertoire to address particular properties of your data analysis problem. It really helps if you ask yourself what would I really like to have.

3.1 k-means Clustering

- k-means Clustering Costs: $\mathcal{R}^{km}(c, \mathbf{Y}, \mathbf{X}) = \sum_{i \leq n} ||x_i - y_{c(i)}||^2$
- "Optimal clustering": $(\hat{c}, \hat{y}) \in \arg\min_{c,y} \mathcal{R}^{km}(c, \mathbf{Y}, \mathbf{X})$

Initialization of the centroids may introduce randomness.

Hard assignments are problematic from a probabilistic perspective. We do not have a certainty, but the calculus we use do not represent the doubt about the assignments. Nearest neighbor rule is the culprit.

A global minimum may be fragile with respect to noise, that is, if the same signal is used with different noise realizations, the outputs may be very different, in some sense a discontinuous input-output relation. A possible solution to this problem is given by maximum entropy clustering.

Limitations: very dependent on the initial configuration and converges to a local minimum.

Independence:

- additive cost corresponds to some form of conditional independence
- given the prototypes, observations are assigned independently to their optimal clusters
- in other words: the global optimal assignment is the concatenation of the individual optimal assignment
- this is what makes alternate optimization tractable in the K-means despite its combinatorial aspect

3.2 Metropolis Sampler for Clustering

Markov chains:

- A Markov chain is said to be irreducible if it is possible to get to any state from any state.
- A state i has period k if any return to state i must occur in multiples of k time steps. If $k = 1$, then the state is said to be aperiodic. Otherwise ($k > 1$), the state is said to be periodic with period k. A Markov chain is aperiodic if every state is aperiodic. An irreducible Markov chain only needs one aperiodic state to imply all states are aperiodic.

Stationarity condition implies the condition that the centroids to be average of the objects belonging to clusters. Replacing the squared distortion measure by the absolute value yields the median as centroid in 1-dim.

MCMC sampler for clustering: it is an algorithm that accepts cost improvements with probability 1 and cost deteriorations with a smaller probability. Randomness allows the algorithm to escape local minima. Proposal distribution should reflect the search strategy.

Probability of being in a state is the Boltzmann weight for that state.

Detailed balance of an algorithm guarantees that the Gibbs distribution is a stationary distribution of the Metropolis sampling process. Detailed balance is a sufficient condition to converge towards Gibbs distribution but not a necessary one. There might be algorithms that violate detailed balance but still end up converging towards Gibbs distribution.

Gibbs sampling for clustering: You keep all the assignments the same except of object i and draw a new assignment $c(i)$.

A flat prior does not exploit the local structure, topology basically plays no role.

Areas of low cost does not mean that the solutions look similar, they are only judged to be comparable.

The idea of simulated annealing consists in letting β vary with time (through an annealing schedule). For problems where finding an approximate global optimum is more important than finding a precise local optimum in a fixed amount of time, simulated annealing may be preferable to alternatives such as gradient descent.

3.3 Maximum Entropy Clustering

Fuzzy assignment algorithm does not operate in the state of solutions as it assigns probabilistic values to multiple centroids for each point.

What we do is we replace the sampling solutions by empirical estimates of fractional assignments of objects to clusters. So we get a summary statistics of the properties of all the solutions in the set of solutions which are highly probable under the Gibbs distribution and that's what the $\mathbf{P}_{i\alpha}$'s should give us. They should tell us about all the solutions in the set of highly probable solutions given the Gibbs distribution, how many of these solutions assign objects to clusters. This summary statistics gives us the fractional $\mathbf{P}_{i\alpha}$'s. It is a way of representing a potentially exponentially large number of solutions with these probabilities. [11.03.2019, around 1:00:00]

How deterministic annealing works:

- Set T to an arbitrarily large value and gradually reduce T until it reaches a negligible value.
- Define c's distribution P using maximum entropy. Compute centroids that maximize P's entropy. Repeat.

When $T \to 0$, P concentrates on the k-means cluster assignment. So when T is sufficiently small, the optimal cluster assignment is the one that assigns each point to its closest centroid.

When $T \to \infty$, there is a unique solution where all centroids are equal to the centroid of all points and P is the uniform distribution.

As T decreases, the number of local minima increases.

Sum-product trick (in the calculation of the k-means Gibbs distribution): Switching the product and the sum is only possible for cost functions for which when you condition on the parameters all the objects become statistically independent. This depends on the fact that the cost function is linear in the individual costs i.e. there are no nonlinear "interactions". Labelling for data point i is statistically independent from the labelling of data point j, for $i \neq j$.

- Deterministic setting imposes $\nabla_\theta R = 0$
- Probabilistic setting requires $\mathbb{E}[\nabla_\theta R] = 0$

3.4 Least Angle Clustering

- Uses cosine similarity as the similarity measure in clustering.
- Its cost function has quadratic terms in M (it is non-linear in the assignment variables), so we cannot use the sum-product trick as we did in k-means.
- It gives too much weight to large clusters. This cost function is good for demonstrating the linearization trick, but not very good for actually solving a practical problem.

$$\begin{aligned} R(M,X) &= -\frac{1}{n}\sum_{\nu=1}^{k}\sum_{i=1}^{n}\sum_{j\geq i}^{n} M_{i\nu}M_{j\nu}\cos\phi_{ij} \\ &= -\frac{1}{2n}\sum_{\nu=1}^{k}\sum_{i=1}^{n}\sum_{j=1}^{n} M_{i\nu}M_{j\nu}e_i e_j + \frac{1}{n}\sum_{\nu=1}^{k}\sum_{i=1}^{n} M_{i\nu}^2 \\ &= -\frac{1}{2n}\sum_{\nu=1}^{k}\sum_{i=1}^{n}\sum_{j=1}^{n} M_{i\nu}M_{j\nu}e_i e_j + 1 \\ &= -\frac{1}{2n}\sum_{\nu=1}^{k}(\sum_{i=1}^{n} M_{i\nu}e_i)^2 + 1 \end{aligned}$$

Partition function becomes

$$\mathcal{Z} = e^{-\beta}\sum_{\{M_{i\nu}\}}\prod_{\nu=1}^{k}\exp\left(\frac{\beta}{2n}(\sum_{i=1}^{n} M_{i\nu}e_i)^2\right)$$

Now we consider one of the products and plug the result back to the partition function

$$\sqrt{\frac{\pi}{\beta n}}^d = \int_{\mathbb{R}^d} dy_\alpha \exp\left(-\frac{\beta n}{2}(y_\alpha - \frac{1}{n}\sum_i M_{i\alpha}e_i)^2\right)$$

y_α is a sufficient statistic for this problem, it decouples the problem. $M_{i\alpha}$'s become conditionally independet given y_α's.

$$\begin{aligned}\mathcal{Z} &= e^{-\beta}\sqrt{\frac{\beta n}{\pi}}^d \sum_{\{M_{i\nu}\}}\prod_{\nu=1}^{k}\int_{\mathbb{R}^d} dy_\nu e^{-\beta n y_\nu^2}\prod_{i=1}^{n} e^{\beta M_{i\nu}e_i y_\nu} \\ &= e^{-\beta}\sqrt{\frac{\beta n}{\pi}}^d \int\cdots\int dy_1\ldots dy_k \exp(-\beta n f(y,x))\end{aligned}$$

This integral is dominated by the maximal value of what's inside the bracket and that allows us in the asymptotic limit to calculate these integrals just with one value.

$$\frac{\partial}{\partial y_\alpha} = 0 \implies y_\alpha = \frac{1}{n}\sum_i e_i p_{i\alpha} \text{ and } p_{i\alpha} = \frac{\exp(\beta e_i y_\alpha)}{\sum_{\nu=1}^{k}\exp(\beta e_i y_\nu)}$$

Since it is super peaky (concentrated) integrand, it is sufficient to expand it to its second derivative through Taylor expansion. This is the standard trick to linearize (at the expense of $O(1)$ integrals) a large sum when you know how to form your interaction term as a square. Then you sum up everything and you have the Gibbs distribution, from which you can sample solutions.

In some sense, that's the essential trick which you can use in this context. If this trick doesn't work, you either have to go to an approximation or you have to resort to an algorithm which sort of sums up iteratively (like belief propagation) all the contributions. If you want to have a close form term that's the only trick I [Buhmann] know which works in these circumstances.

Linearization trick, in concise form:

$$\begin{aligned}e^{\frac{b^2}{2a^2}} &= \int_{-\infty}^{+\infty}\frac{a}{\sqrt{2\pi}}\exp\left(-\frac{a^2x^2}{2}+bx\right)dx \\ e^{-\frac{b^2}{2a^2}} &= i\int_{-i\infty}^{+i\infty}\frac{a}{\sqrt{2\pi}}\exp\left(\frac{a^2x^2}{2}-bx\right)dx\end{aligned}$$

3.5 Self-Organized Clustering (SOMs)

$$\mathcal{R}^{som}(c,\mathbf{Y},\mathbf{X}) = \sum_{1\le i\le n}\sum_{1\le\nu\le k}\mathbf{T}_{c(i),\nu}D(x_i,y_\nu)$$

where T is the transmission error and D is the quantization error.

- SOM reduces data dimensions and displays similarities among data by applying competitive learning. They are also called Kohonen maps. (It is sort of finds confusion-robustified means.)
- The index of a code book vector can be changed by noisy communication.
- We have two sources of distortions, i.e., quantization error and code vector confusion.

3.6 Complexity Constraint Clustering

The number of clusters is not set a priori, but it is determined by jointly minimizing distortion and complexity costs.

Entropy constraint refers to Huffman coding, you have to pay a price for code lengths for heavily used indexes.

You pay logarithmic costs for coding, but the probability that you actually need this long code goes to zero - that way we can invest a centroid for only one datapoint.

When you have entropy constrained clustering, uniform quantization is the optimal solution where codebook vectors are ignorant to the data distribution.

4 Clustering Distributional Data

Invariantly from your clustering model, you can determine what is the natural distortion measure for that particular cluster when you look at how you imagine the data are generated.

4.1 Histogram Clustering

- Likelihood of the data is

$$\mathcal{L}(\mathcal{Z}) = \prod_{i \leq n} \prod_{j \leq m} P((i,j)|c(i), q(j|c(i))^{l\hat{p}(i,j)}$$

- $P((i,j)|c,q) = q(j|c(i)p(c(i))$
- $D^{KL}(\hat{p}(\cdot|i) \parallel q(\cdot|\alpha)) = \sum_j \hat{p}(j|i) \log \frac{\hat{p}(j|i)}{q(j|\alpha)}$, the cost for assigning object x_i to class α
- $\mathcal{R}^{hc}(c; \{q(\cdot|\alpha)\}) = \frac{1}{n} \sum_{i \leq n} D^{KL}(\hat{p}(\cdot|i) \parallel q(\cdot|c(i)))$
- $\hat{p}(\cdot|i)$: empirical histogram for object i, i.e. object specific histogram which should be similar to the centroid histogram
- $q(\cdot|\alpha)$: model probability distribution of features conditioned on cluster $c(i)$, i.e. centroid histogram for all objects which are assigned cluster α
- Solving stationary equations yields $q(j|\nu) = \frac{\sum_i P_{i\nu} \hat{p}(j|i)}{\sum_i P_{i\nu}}$

Data is represented by probability distributions instead of Euclidean space. We get a partition of the probability simplex rather than a partition of the Euclidean space. Essentially we would like to think about how to compare histograms.

The assignment of object i to cluster α is measured with the **Kullback-Leibler divergence** between the conditional feature distribution of the object relative to the model distribution of that cluster and that is **the natural way** of comparing histograms to a centroid in the probability simplex.

Cost function is linear in $c(i)$, again yielding a product over all objects in Gibbs distribution formulation. We again a get a mean definition, same as in the k-means. Now, we calculate the mean of all the feature distributions given the objects i when the object i

is assigned to cluster ν. You take the weighted average of the object histograms over the clusters.

The final goal is to infer the generative histograms. They can be seen as centroids in the space of histograms, which brings a flavour of k-means clustering. In order to fulfil the task we will use empirical posterior probabilities.

Equations for Gibbs distribution and centroids can be solved using a deterministic annealing procedure analogous to the k-means case. In summary, once the nature of the data has been qualified as distributional data, from the clustering perspective, the problem is analogous to that of k-means.

Lack of topology: Histograms represent categorical information while most of the feature spaces are equipped with a natural topology (feature similarity): color circle, frequency ordering, edge directions, etc. Permutation of the bin index will not change the cost in D_{KL}. Indeed we can't impose a structure: we don't know anything about pixels by themselves.

In short, KL-divergence is permutation invariant w.r.t. the features, any permutation gives exactly the same value. This is a problem when we have an ordered feature space.

Heuristic histogram smoothing is advisable in many applications, e.g., distribute a small percentage of a count to neighbouring bins.

4.2 Parametric Distributional Clustering

A single selection of Gaussian prototypes $g_\alpha(y)$ is used to create mixture densities:

$$p(y \mid \nu) = \sum_{\alpha=1}^{s} p(\alpha \mid \nu) g_\alpha(y)$$

We represent the model distribution for all the features y (could be continuous) given the cluster ν, by mixture parameters (coefficients) conditioned on the particular cluster ν and Gaussian prototypes which you mix for particular feature y and for a mode α. It is no longer permutation invariant, which is a significant improvement.

If you have a feature domain with finite support, you have a problem as Gaussian distribution has infinite support. Feature values at the boundary cannot be modelled appropriately by Gaussian distributions. We have to use model distributions which are limited to the feature domain.

Rectified Gaussians: then we have to rectify the Gaussians by cutting off the probability mass that goes outside and scaling it up. This is just another modelling assumption.

$g_\alpha(y)$ is the original Gaussians, $\widetilde{g}_\alpha(y)$ is the rectified Gaussians and $\widetilde{G}_\alpha(j)$ is the density corresponding to one feature. Then,

$$\text{Prior of assignment function } c\text{: } p(c) = \prod_{i=1}^{n} p_{c(i)}$$

$$\textbf{Data likelihood given } c\textbf{: } p(\mathcal{Y} \mid c, \theta) \prod_{i=1}^{n} \left[\prod_{j=1}^{m} \left(\sum_{\alpha=1}^{s} p(\alpha \mid c(i)) \widetilde{G}_\alpha(j) \right)^{l\hat{p}(i,j)} \right]$$

parameter vector θ summarizes all the parameters of the mixture model, e.g., mixture coefficients, centroids and covariances of the Gaussians, binning parameters.

M-step does not admit a closed form solution for $p(\alpha \mid \nu)$ or μ_α (Gaussian means) anymore, so optimization of the parameters is done using numerical methods. One way is choosing randomly pairs of Gaussians and shifting weight from one to the other.

Moral of the story: If you happen to be forced to invent a model, separate as many issues as possible. There is an issue in defining a clustering model and there is an issue to define the centroid probability distributions of your features given a particular cluster. these are independent knobs on which you can tune your model, if you mix up everything you will have a very hard time to come up with a good model. Separating issues will help localizing problems.

4.3 Information Bottleneck

We want to find groups such that we compress the information in the object space as much as possible (by minimizing mutual information between the object index and the cluster index).

We minimize the mutual information between x and c, as we want to have generic centroids. Peculiarities of the objects x should be forgotten and what is generic of all the objects which are assigned to a particular cluster should be retained - that's what we care about.

Maximally minimizing the mutual information without a counter-force is basically assigning all the objects to same cluster, which does not preserve anything. It is entropy death, there is no structure whatsoever. So we need a counterforce, which is the constraint that we force the representations to be as informative as possible relative to a feature space.

Normally we do not know $P(x, y)$ but only know the empirical observation of the true distribution.

Tradeoff is made explicit by cost functional

$$\mathcal{R}^{IB} = I(X;C) - \lambda I(C;Y), \text{ where } c_{opt} = \arg\min_c \mathcal{R}^{IB}$$

$I(C;Y)$ acts like a distortion constraint and $I(X;C)$ is an objective for quantization. Problem is designing C such that information is preserved about Y.

4.4 Rate Distortion Theory

Shannon/Kolmogorov's rate distortion theory states that the optimal representation fulfils the rate distortion function.

5 Pairwise Clustering

Clustering algorithms always impose structure on data. There is not something like model free data analysis.

Buhmann's view is clustering as partitioning, an alternative view is clustering as density estimation.

We would like to have a similarity measure which basically ignores the influence of confounding variables. Variations of such a variable define not a point but a manifold and we measure the distance to this manifold.

5.1 Similarities versus Dissimilarities

It is a very wise assumption when you want to build a model to use large numbers only when you are pretty certain that these large numbers are doing the job. This is important as large distances are difficult to measure accurately in many applications.

When you have a comparison function which is based on dissimilarities and large dissimilarities give you large costs. You want to minimize costs but you have an implicit inference bias by giving large numbers when you are particularly uncertain.

Using similarities might lead to non-convex maximization problem whereas dissimilarities in most cases gives you a convex minimization problem. So, computational reasons speak for dissimilarities strategy but model selection reasons, I believe, speak against this strategy.

You want to start in situations where you are pretty unsure about how you should compare things but have a clear view of testing for identity or high similarity, you want to go for similarities. Very large mistakes with very large distances which are very rare might still influence your estimators.

5.2 Correlation Clustering

Inter-cluster costs can be transformed into intra cluster costs:

$$\begin{aligned}
\mathcal{R}^{cc}(c;\mathcal{D}) &= -\sum_{\nu\leq k}\sum_{(i,j)\in\varepsilon_{\nu\nu}} S_{ij} + \sum_{\nu\leq k}\sum_{\mu\leq k,\mu\neq\nu}\sum_{(i,j)\in\varepsilon_{\nu\nu}} S_{ij} \\
&= -2\sum_{\nu\leq k}\sum_{(i,j)\in\varepsilon_{\nu\nu}} S_{ij} + \sum_{\nu\leq k}\sum_{\mu\leq k}\sum_{(i,j)\in\varepsilon_{\nu\nu}} S_{ij} \\
&= -2\sum_{\nu\leq k}\sum_{(i,j)\in\varepsilon_{\nu\nu}} S_{ij} + \sum_{(i,j)} S_{ij} \\
&= -2\sum_{\nu\leq k}\sum_{(i,j)\in\varepsilon_{\nu\nu}} S_{ij} + const
\end{aligned}$$

- Correlation Clustering: Has problem with the costs as large clusters have much more weight than small clusters! Cost function seems to put wrong emphasis in evaluating solutions.
- Shifted Correlation Clustering: Uses continuous similarities and sums them up relative to a threshold.
- Graph Partitioning: Partition via graph cuts using continuous dissimilarities, very similar to correlation clustering.

- Normalized Graph Partitioning: To get rid of the bias of the former for very unbalanced (very small and very large) clusters.

5.3 Review on centering

$$C_n = I_n - \frac{1}{n}\mathbb{O} = I_n - \frac{1}{n}\mathbf{1}\mathbf{1}^T$$

where $\mathbb{O}$ is a matrix of all 1's. In our notation, this is $Q = I_n - \frac{1}{n}U(n)$, $P^c = QPQ$.

For an m-by-n matrix X, the multiplication $C_m X$ removes the means from each of the columns, while XC_n removes the means from each of the m rows.

For a n-by-n matrix X, multiplication $C_n X C_n$ creates a doubly centered matrix where both now and column means are equal to zero.

Scatter matrix is $S = XC_n(XC_n)^T = XC_nX$

5.4 Pairwise Data Clustering

cost function is invariant to shifting off-diagonal elements, therefore we change the spectrum so that all eigenvalues are positive. This means we can interpret it as a scalar product in a high dimensional space and that this cost function effectively operates like k-means in such kernel space.

6 Mean-field Approximation

The **problem** is usually the evaluation of the partition function that involves exponentially many terms and is analytically intractable. This often happens when the cost functions is not linear in the individual costs, or includes non trivial structures terms that make that the cost are not independent from one other. Cost functions for which the Gibbs distribution can be calculated exactly are rare, e.g., K-means clustering or histogram clustering conditioned on centroids. **Strategy** we will pursue is fitting statistically independent models to models where you have a lot of statistical dependency.

6.1 Proposal distribution

Approximate

$$p(c \mid x) = e^{-\beta(\mathcal{R}(c,x)-\mathcal{F}(x))}, \text{ where } \mathcal{F}(x) = -\frac{1}{\beta}\log\sum_c e^{-\beta\mathcal{R}(c,x)}$$

by a factorial distribution

$$Q(c,\theta^0) = \prod_{i=1}^{n} q_i(c(i)), \text{ where } q_i(\alpha) = \frac{exp(-\beta h_{i,\alpha})}{\sum_{\nu=1}^{k}\exp(-\beta h_{i,\nu})} \in [0,1]$$

$h_{i,\alpha}$ are parameters that denote the partial costs of assigning object i to cluster α. As an example, for k-means $h_{i,\alpha} = \|x_i - y_\alpha\|^2$

Notice that the probability of a configuration c is given by the probability of the individual assignments $q_i(c(i))$ and that there are no correlations between the assignments of different objects $c(i), c(j), i \neq j$ in this model. We disregard statistical dependencies between assignment variables while approximating the Gibbs distribution.

6.2 Approximation strategy

Since Kullback-Leibler divergence is always non-negative,

$$
\begin{aligned}
0 &\leq \frac{1}{\beta} D_{KL}(Q(c,\theta^0) \parallel p(c \mid x)) \\
&= \frac{1}{\beta} \sum_{c \in \mathcal{C}} Q(c,\theta^0) \log \frac{Q(c,\theta^0)}{\exp[-\beta(\mathcal{R}(c,x) - \mathcal{F}(x)]} \\
&= \frac{1}{\beta} \sum_{c} Q(c,\theta^0) \log \prod_{i=1}^{n} q_i(c(i)) - \frac{1}{\beta} \sum_{c} Q(c,\theta^0)(-\beta \mathcal{R}(c,x) + \beta \mathcal{F}(x)) \\
&= \frac{1}{\beta} \sum_{c} \prod_{j=1}^{n} q_j(c(j)) \sum_{i=1}^{n} \log q_i(c(i)) + \mathbb{E}_Q[\mathcal{R}(c,x)] - \mathcal{F} \\
&\overset{*}{=} \frac{1}{\beta} \sum_{i=1}^{n} \sum_{c(i)=\{1,\ldots,k\}} q_i(c(i)) \log q_i(c(i)) + \mathbb{E}_Q[\mathcal{R}(c,x)] - \mathcal{F} \\
&= \frac{1}{\beta} \sum_{i=1}^{n} \sum_{\nu=1}^{k} q_i(\nu) \log q_i(\nu) + \mathbb{E}_Q[\mathcal{R}(c,x)] - \mathcal{F} \\
\mathcal{F}(x) &\leq \mathbb{E}_Q[\mathcal{R}(c,x)] + \frac{1}{\beta} \sum_{i} \sum_{\nu} q_i(\nu) \log q_i(\nu) =: \mathcal{B}
\end{aligned}
$$

- Strategy is to minimize $\mathcal{B}$ w.r.t. $q_u(\alpha)$, i.e. $h_{i,\alpha}$ parameters
- $\frac{1}{\beta} \sum_i \sum_\nu q_i(\nu) \log q_i(\nu)$ is the negative entropy of the clustering solution
- *: script equations 4.10-4.15, lecture slide 5: the summation $\sum_{c \in \mathcal{C}}$ reduces to $\sum_{c(i)=\{1,\ldots,k\}}$ due to normalization. Summations of $c(j), j \neq i$ sum up to 1.

The bound $\mathcal{B}(\{q_i(\nu)\})$ comprises the expected costs and the temperature scaled negative entropy of the approximating probability distribution. Therefore, the bound defines an approximating free energy. It is important to note that the method requires an approximating distribution for which the KL-divergence can be efficiently evaluated. The factorial structure is sufficient for this condition but not necessary.

Trick above, illustrated:

- $\frac{1}{\beta}\sum_c \prod_{j=1}^n q_j(c(j)) \sum_{i=1}^n \log q_i(c(i))$

$$\begin{aligned}
&= f_{11}f_{21}\dots f_{n1}[\log f_{11} + \log f_{21} \cdots + \log f_{n1}] \\
&+ f_{11}f_{21}\dots f_{n2}[\log f_{11} + \log f_{21} \cdots + \log f_{n2}] \\
&+ \dots \\
&+ f_{1k}f_{2k}\dots f_{nk}[\log f_{1k} + \log f_{2k} \cdots + \log f_{nk}]
\end{aligned}$$

- $\frac{1}{\beta}\sum_{i=1}^n \sum_{c(i)=\{1,\dots,k\}} q_i(c(i)) \log q_i(c(i))$

$$= f_{11}\log f_{11} + f_{12}\log f_{12} + \cdots + f_{nk}\log f_{nk}$$

6.3 Stationarity condition

We write $\mathcal{B}$ explicitly and differentiate w.r.t. $q_u(\alpha)$ for the stationarity condition:

$$\begin{aligned}
0 &= \frac{\partial}{\partial q_u(\alpha)}\left(\mathcal{B} + \sum_{i=1}^n \lambda_i \left(\sum_{\nu=1}^k q_i(\nu) - 1\right)\right) \\
&= \frac{\partial}{\partial q_u(\alpha)}\frac{1}{\beta}\sum_{s\le n}\sum_{\nu\le k} q_s(\nu)\log q_s(\nu) + \frac{\partial}{\partial q_u(\alpha)}\mathbb{E}_Q[\mathcal{R}(c)] + \lambda_u \\
&= \frac{1}{\beta}(1 + \log q_u(\alpha)) + \frac{\partial}{\partial q_u(\alpha)}\sum_{c\in\mathcal{C}}\prod_{i\le n} q_i(c(i))\mathcal{R}(c) + \lambda_u \\
&= \frac{1}{\beta}(1 + \log q_u(\alpha)) + \sum_{c\in\mathcal{C}}\prod_{i\ne u\le n} q_i(c(i))\mathbb{I}_{\{c(u)=\alpha\}}\mathcal{R}(c) + \lambda_u
\end{aligned}$$

We define $h_u(\alpha)$ is the expected cost of $\mathcal{R}(c)$ under the constrain that object u is assigned to cluster α:

$$h_u(\alpha) = \sum_{c\in\mathcal{C}}\prod_{i\ne u\le n} q_i(c(i))\mathbb{I}_{\{c(u)=\alpha\}}\mathcal{R}(c)$$

Remark: $h_u(\bullet)$ is independent on $q_u(\bullet)$. It depends on all other $q_i(\bullet)$ for $i \ne u$.

6.4 Optimization

Rewriting the equations above we get

$$0 = \frac{1}{\beta}(\log q_u(\alpha) + 1) + h_{u,\alpha} + \lambda_u$$

Assignment probabilities become (right: upon normalization)

$$q_u(\alpha) = \exp(-1 - \beta(h_{u,\alpha} + \lambda_u)) = \frac{e^{-\beta(h_{u,\alpha})}}{\sum_{\nu\le k} e^{-\beta(h_{u,\nu})}}$$

Second variations are positive (we determine a minimum in $q_u(\alpha)$ direction)

$$\frac{\partial^2}{\partial q_u(\alpha)^2}\mathcal{B} = (\beta q_u(\alpha))^{-1} > 0$$

This implies an **asynchronous updating scheme** (Meanfield equations):

- $$q_{s(t)}^{New} = \frac{e^{-\beta(h_{s(t),\alpha})}}{\sum_{\nu \leq k} e^{-\beta(h_{s(t),\nu})}}$$
- $$h_{s(t),\alpha} = \mathbb{E}_{Q^{Old}_{s(t)\to\alpha}}\{\mathcal{R}(c)\}$$

where $s(t)$ is an arbitrary site visitation scheme. This will converge to a local minimum of the approximate free energy in the space of factorial distributions. $\mathbb{E}_{Q^{Old}_{s(t)\to\alpha}}$ denotes an expectation over all configurations under the constraint that object $u$is assigned to cluster α.

6.5 Comments

First of all, the factorial form of $\mathbf{Q}$ simplifies the summation over exponentially many assignment configurations k^n of n objects into k clusters, to kn summations and an optimization problem that can be tackled with an Expectation-Minimization approach. Second, the factorial form of $\mathbf{Q}$ is sufficient and convenient to perform a mean field treatment of a problem, but it is not necessary, it is sometimes possible to use distributions that contain correlations, tree like structures for example, that also allows for efficient computations. (See structured MF vs Naive MF)

On D_{KL} ordering: we use this version and not the other way around because in this version at least we average, we take the expectation, w.r.t. a distribution which we can evaluate efficiently like the factorial distribution. Then the gibbs distribution is inside of a logarithm and that helps us to bring down the cost function from exponent by taking the logarithm.

Mean-field (EM) convergence tells you that you have found an extremal point, it does not tell you that you have found a minimum of the upper bound.

6.6 Smooth k-means clustering

Add a regularizer to the k-means cost function to favor homogeneous assignments in the local neighborhood $\mathcal{N}(i)$ (we decompose the cost function into contributions of object u and costs, which are not affected by u):

$$\begin{aligned}\mathcal{R}^{skm}(c) &= \sum_{i\leq n} \left\|x_i - y_{c(i)}\right\|^2 + \frac{\lambda}{2}\sum_{i\leq n}\sum_{j\in\mathcal{N}(i)} \mathbb{I}_{c(i)\neq c(j)} \\ &= \left\|x_u - y_{c(u)}\right\|^2 + \lambda \sum_{j\in\mathcal{N}(u)} \mathbb{I}_{\{c(j)\neq c(u)\}} + \text{costs independent of object } u\end{aligned}$$

Expectations over assignments with $c(u) = \alpha$:

$$\begin{aligned} h_{u,\alpha} &= \mathbb{E}_{\mathbf{Q}_{u \to c(u)}} \left[\left\| x_u - y_{c(u)} \right\|^2 \right] + \lambda \sum_{j \in \mathcal{N}(u)} \mathbb{E}_{\mathbf{Q}_{u \to \alpha}} \left[\mathbb{I}_{\{c(j) \neq c(u)\}} \right] + \text{const} \\ &= \left\| x_u - y_\alpha \right\|^2 + \lambda \sum_{j \in \mathcal{N}(u)} \mathbb{E}_{\mathbf{Q}} \left[\mathbb{I}_{\{c(j) \neq \alpha\}} \right] + \text{const} \\ &= \left\| x_u - y_\alpha \right\|^2 + \lambda \sum_{j \in \mathcal{N}(u)} \sum_{\nu \neq \alpha \leq k} \mathbb{E}_{\mathbf{Q}} \left[\mathbb{I}_{\{c(j) = \nu\}} \right] + \text{const} \\ &= \left\| x_u - y_\alpha \right\|^2 + \lambda \sum_{j \in \mathcal{N}(u)} \sum_{\nu \neq \alpha \leq k} q_j(\nu) + \text{const} \end{aligned}$$

The constant is independent of object u and does not influence the assignment of object u to any cluster α.

7 Model Selection for Clustering

- Cluster structure can be invalid in two ways: inappropriate model type and/or inappropriate model order
- Validation methods can be external (= comparison with ground-truth) or internal.
- We want to use convergence guarantees for sums (e.g. law of large numbers) so we convert the big product term to a big sum over the exponent.
- Keep in mind every method you use for validation induces some kind of a bias, it separates essential things from inessential things.
- You may be grossly misled by naively counting the number of parameters. (cf. discussion in AML on VC-dimension of the sine wave)

General approach when selecting model order is to measure quality for different k. It is a problem of comparing incomparable objects: with more k we gain more bits, so we should have a discount for our model when we have more information. That is, negative log-likelihood usually decreases with increasing model complexity as more parameters support a better fit! What is the right discount (complexity penalty) between model quality and the number of bits we get with the solution?

7.1 MDL and BIC

$$\text{BIC} = -\log(p(\mathbf{X} \mid \theta)) + \frac{1}{2} k' \log n$$

- Occam's razor: choose the model that provides the shortest description of the data. It is the idea formalized by MDL and BIC.
- MDL minimizes the overall description length of the data
- k' in BIC slides refers to number of clusters (k) times the number of dimensions for k-means.

- MDL and BIC are consistent as a model and asymptotically equivalent.
- Penalty term in BIC comes from the Gaussian of a quadratic expansion in the Laplace approximation.

Justification for the BIC: we do nothing but exploiting the finite dimension of the parameter space, the growing number of samples n (classical limit of statistics where the parameters are fixed and the samples go to infinity) and we obtain the two terms. The only problem is that it's a limit we understand very well but which doesn't seem to be appropriate for the world of today where data are precious and the models grow with the number of samples. In other words, problem with this analysis is that it is not interesting for the modern statistics because in modern statistics the more samples you have the more degrees of freedom you usually have.

Statistics have to completely reinvent it, because it was always parametric statistics and that's exactly not what the user wants to have. If the data is precious and just not coming for free we want to get most mileage per sample out of the data. So we should expect the models to get more complex the more samples we actually gather, as a natural reasoning. We need a new type of statistical reasoning.

If the number of parameters grow with the number of samples, our argument is no longer valid. [This matrix] ceases to be $O(1)$, instead it grows with n.

MDL and BIC are well-motivated model selection schemes. However, likelihood optimization is the hard problem because the data space might be much larger than the hypothesis class. That means if you'd do model selection based on posterior we might have an easier job. Because we won't have to estimate the density of the input. MDL-based validation is not generally applicable because of this difference between likelihood validation and posterior validation.

7.2 Gap Statistic

Idea is to find "kink" in costs. It uses maximum discrepancy between actual data cost and that of unstructured reference data (i.e. which cannot be clustered, so-called "null model"). If the data are too noisy, the method may fail!

$$\text{gap}_n(k) \coloneqq \mathbb{E}_n^*[\log(W_k)] - \log(W_k), \text{ where } W_k = \sum_{1 \le \nu \le k} \frac{1}{2n_\nu} \sum_{(i,j) \in \mathcal{E}_{\nu\nu}} D_{ij}, \text{ with } D_{ij} = \|x_i - x_j\|^2$$

- W_k above is called within-class dispersion.
- "The idea of our approach is to standardize the graph of $\log(W_k)$ by comparing it with its expectation under an appropriate null reference distribution of the data."
- $\mathbb{E}_n^*$ is the expectation value w.r.t sample of size n from the null model as a reference distribution.
- In practice approximate $\mathbb{E}_n^*$ by bootstrapping.

Comments:

- The Gap Statistic works often for spherically distributed data.

- It is not model-free, as it contains a structural bias. (The gap statistic assumes compact or spherically distributed clusters)
- For k-means-like criteria, it is a fast heuristic.
- Many similar methods have been proposed in the literature.

7.3 Stability Based Validation

Idea is that solutions on two data sets from the same source should be similar (**stability**). Signal in both datasets should be the same but the fluctuations should be different.

A stable solution can be transferred to a second data set drawn from the same distribution at minimal model misfit. Model mismatch produces unstable clustering solutions. This behavior can be detected and used for model order selection.

In practical approaches you only have one dataset and resample from that available dataset (for i.i.d. data).

Naively comparing labelings leads to a huge discrepancy because cost function to guide your search four clusterings is permutation invariant w.r.t. the cluster index. Stability should be taken not blindly with a naive comparison but w.r.t. the structure of the solution. One solution would be to match one permutation with another through Hungarian method.

The **type of classifier** largely influences the stability measurement and therefore has to be selected with care. Effect of wrong predictor: using the training data with path-based clustering, a nearest centroid classifier cannot reproduce the target labeling. In this case, the disagreement between two solutions is artificially increased by the inappropriate predictor.

The data should be split into **two disjoint subsets** because their overlap could otherwise already determine the group structure. This statistical dependence would lead to an undesirable, artificially induced stability. Hence, the use of bootstrapping can be dangerous as well in that it can lead to artificially low disagreement between solutions. Furthermore, the data sets should have (approximately) **equal size** so that an algorithm can find similar structure in both data sets. If there are too few samples in one of the two sets, the group structure might no longer be visible for a clustering algorithm.

Procedure:

1. **Transfer solution via prediction:** Construct a classifier ϕ trained on $(\mathbf{X}, \mathbf{Y})$, where $\mathbf{Y}$ is the clustering solution to $\mathbf{X}$ as $\mathbf{Y} \coloneqq \mathcal{A}_k(\mathbf{X})$. We consider the predicted labeling $\phi(\mathbf{X}') \coloneqq (\phi(X_i'))_{i \leq n}$ as the extension of the clustering solution $\mathbf{Y}$ on data set $\mathbf{X}$ to the data set $\mathbf{X}'$. These predicted labels can be compared to those generated by the clustering algorithm, that is, with $\mathcal{A}_k(\mathbf{X}')$.
2. **Compare solutions:** A very natural distance measure for comparing these two labeling vectors $\phi(\mathbf{X}'), \mathbf{Y}'$ is to consider their normalized Hamming distance (i.e. 0-1 loss). This can be interpreted as the empirical misclassification risk of ϕ with regard to the test set.
3. **Permute solutions:** To overcome the nonuniqueness of representation, we modify the dissimilarity such that the label indices in one solution are optimally permuted to maximize the agreement between the two solutions under comparison. Technically, the minimization can be performed in time $O(n + k^3)$ by using the Hungarian method

for minimum weighted bipartite matching, which is guaranteed to find the globally optimal solution.

$$\text{Dissimilarity: } d_{\mathfrak{S}_k}(\phi(\mathbf{X}'), \mathbf{Y}') \coloneqq \min_{\pi \in \mathfrak{S}_k} \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}\{\pi(\phi(X_i')) \neq Y_i'\}$$

$$\text{Stability: } S(\mathcal{A}_k) \coloneqq \mathbb{E}_{\mathbf{X},\mathbf{X}'}\, d_{\mathfrak{S}_k}(\phi(\mathbf{X}'), \mathbf{Y}')$$

We call S the stability index of the clustering algorithm $\mathcal{A}_k$. It is the average dissimilarity of clustering solutions with regard to the distribution of the data. Hence, the smaller the values of the index $S(\mathcal{A}_k) \in [0, 1]$, the more stable are the solutions.

4. **Calculate relative stability:** We calculate relative stability w.r.t. the stability achieved by random label guessing as $\bar{S}(\mathcal{A}_k) \coloneqq \frac{S(\mathcal{A}_k)}{S(\mathcal{R}_k)}$, where $\mathcal{R}_k$ is a labeling algorithm that assigns an object to a class with probability $1/k$. Note that the stability measure is not defined for $k = 1$.

Summary:

- Stability Principle: Solutions for two data sets drawn from the same source should be similar. No additional assumptions about the structure of solutions. Good performance on experimental data sets.
- Stability measure worked quite well except it had systematic bias towards too simplistic solutions.
- In summary, stability is a principled approach but with a deficit. It only tells you how reliable under repeated experimentation your algorithm would react, but it does not take into account whether your solution should be informative or not.
- In other words, the stability method does not take the complexity of a solution appropriately into account. The trade-off between informativeness and stability of solutions should be quantitatively evaluated. Such a goal requires a theory for validation of algorithms!
- Buhmann's another paper: Stability is, however, only one aspect of statistical modeling, e.g., for unsupervised learning. The other aspect of the modeling tradeoff is characterized by the informativeness of the extracted patterns. A tolerable decrease in the stability of inferred patterns in the data (data model) might be compensated by a substantial increase of their information content

8 Model Validation by Information Theory

Maximal information per symbol is the channel capacity $C = \max_{P(X)} I(X;Y)$ with channel noise $P(Y \mid X)$ given. For a discrete memoryless channel, all rates below capacity C are achievable. Specifically, for every rate $R < C$, there exists a sequence of $(2^{nR}, n)$ codes with maximum probability of error $\lambda^{(n)} \to 0$.

We need robustness as algorithms overfit. If $X', X'' \sim P(X)$, then $P(c \mid X') \approx P(c \mid X'')$. Robustness by posterior sampling: replace "solution" $c^{\perp}$ (algorithm $\mathcal{A}$) by a distribution of

solutions, i.e. with a sampling procedure that draws solutions from the posterior $P^{\mathcal{A}}(c \mid X)$.

$$P^{\mathcal{A}}(c^{\perp}) = \int_{\mathcal{X}} P(X)\delta(c^{\perp} - \mathcal{A}(X))dX$$

Approximation weights is the answer to the question "How should we construct a posterior?".

8.1 Typical sets

Asymptotic equipartition property (AEP): The normalized logarithm of the probability of a sequence converges towards its entropy per symbol (proof by law of large numbers):

$$-\frac{1}{n}\log P(X_1, \ldots, X_n) \overset{\text{in prob.}}{\to} H(X), \text{ for } i.i.d. X_1, \ldots, X_n \sim P(x)$$

A **typical set** $A_{\epsilon}^{(n)}$ w.r.t. $P(x)$ is the set of sequences $(x_1, \ldots, x_n) \in \mathcal{X}^n$ with the property

$$2^{-n(H(x)+\epsilon)} \leq P(X_1, \ldots, X_n) \leq 2^{-n(H(x)-\epsilon)}$$

- If $(x_1, \ldots, x_n) \in A_{\epsilon}^{(n)}$, then $H(x) - \epsilon \leq -\frac{1}{n}\log P(x_1, \ldots, x_n) \leq H(x) + \epsilon$
- $P\{A_{\epsilon}^{(n)}\} > 1 - \epsilon$ for n sufficiently large
- $|A_{\epsilon}^{(n)}| \leq 2^{n(H(X)+\epsilon)}$, where $|A_{\epsilon}^{(n)}|$ denotes the cardinality of the typical set.
- $|A_{\epsilon}^{(n)}| \geq (1-\epsilon)2^{n(H(x)-\epsilon)}$ for n sufficiently large

The typical set $|A_{\epsilon}^{(n)}|$ has probability almost 1, all elements of the typical set are nearly equiprobable, and the number of typical solutions (elements of the typical set) is nearly 2^{nH}.

8.2 Approximation weights

$$w : \mathcal{C} \times \mathcal{X} \times \mathbb{R}_+ \to [0, 1], \text{ such that } (c, X, \beta) \mapsto w_{\beta}(c, X)$$

Weights are non-negative and the maximal weight is allocated to the global minimizer $c^{\perp}$ and it is normalized to one ($w_{\beta}(c^{\perp}, X) = 1$). Semantically, weights quantify the quality of a solution w.r.t. the global minimizer. Solutions with large approximation weights ($w_{\beta}(c, X) \geq 1 - \epsilon, \epsilon \ll 1$) can be accepted as substitutes of the global minimizers. Posterior distribution becomes

$$P(c \mid X) = \frac{w_{\beta}(c \mid X)}{\sum_{c'} w_{\beta}(c' \mid X)}$$

We require that weights fulfil the inverse order constraints compared to the costs

$$R(c, X) \leq R(\tilde{c}, X) \iff w_{\beta}(c, X) \geq w_{\beta}(\tilde{c}, X)$$

Example (unnormalized) weights:

- Boltzmann weights: $w_{\beta}(c, X) = \exp(-\beta R(c, X))$
- Fermi weights: $w_{\beta,\gamma}(c, X) = (1 + \exp(-\beta(R(c, X) - \gamma)))^{-1}$

- Approximation weights: $w_\gamma(c, X) = \begin{cases} 1 & \text{if } R(c, X) \leq R(c^\perp, X) + \gamma \\ 0 & \text{otherwise} \end{cases}$
  (also known as microcanonical partition function or binary weight function)

Normalized Boltzmann weights:

$$w_\beta(c, X) \coloneqq \exp(-\beta \Delta R(c, X)), \text{ with } \Delta R(c, X) \coloneqq R(c, X) - R(c^\perp, X)$$

- $\beta = 0$: all weights $w_\beta(c, X) = 1$ independent of the costs. $Z_q = |c(X)|$ indicates the size of the hypothesis space.
- high β: all weights are small compared to $w_\beta(c^\perp, X)$ global minimizer. Z_q essentially counts the number of globally optimal solutions.
- intermediate β: Z_q is the effective number of patterns that approximately fit the dataset X, where β defines the precision of this approximation. Noise in the measurements X reduces this resolution and thus coarsens the hypothesis class.

In the absence of noise in the data, error-free communication works even for $\beta \to \infty$.

Weight sum: measures the total weight of hypotheses with low costs. It is also known as the partition function in statistical physics when we use Boltzmann weights.

8.3 Equivariance transformations

Idea is to transform the data such that posterior is shifted. We assume a set of transformations $\mathbb{T}$ such that

$$\forall \tau', \tau'' \in \mathbb{T}, \|P(c \mid \tau' \circ X) - P(c \mid \tau'' \circ X)\|_1 > 0$$

This condition excludes transformations that leave the posterior invariant. It also implies $|\mathbb{T}| \leq |\mathcal{C}|$ for discrete hypothesis spaces.

$$\sum_{\tau \in \mathbb{T}} P(c \mid \tau \circ X) \in \left[\frac{|\mathbb{T}|}{|\mathcal{C}|}(1 - \rho), \frac{|\mathbb{T}|}{|\mathcal{C}|} \right]$$

where ρ measures the inhomogeneity of transformations.

Transformations correspond to the equivariance class of the algorithm ($\tau \circ c(X) = c(\tau \circ X)$) that should be considered as as statistical estimator. Transformations for specific estimation problems can be found in the table in the slide 17.

Note: It is essential that the experimental measurements of X are not changes, e.g., noise should not be added to the experimental values. Otherwise, the data analysis algorithm might miss the essential signal for optimal estimation.

8.4 Communication scenario

Codebook generation: Sender and receiver both receive an instance X' from problem generator, calculate posterior $P(c \mid X')$ and agree on a set of M randomly drawn transformations

$T = \{\tau_1, \ldots, \tau_M\}$ with $P(\tau) = |\mathbb{T}|^{-1}$. Here, posteriors $P_\theta(c \mid \tau_j \circ X')$ play the role of codewords in Shannon's random coding theory.

Communication protocol: Sender selects a transformation $\tau_s \in T$ as message and sends it to the problem generator. Problem generator generates new instance $X'' \sim P(X)$ and applies transformation τ_s, which yields $\tilde{X} = \tau_s \circ X''$. Problem generator sends $\tilde{X}$ to receiver without revealing either τ_s or X''. So receiver lacks both the knowledge of τ_s and suffers from the stochastic variability of X. Then receiver calculates the expected posterior $P_\theta(c \mid \tilde{X})$ and decodes the message $\hat{\tau}$:

$$\hat{\tau} \in \arg\max_{\tau \in T} \mathbb{E}_{c \sim P_\theta(c \mid \tau \circ X')} P_\theta(c \mid \tilde{X})$$

We introduce a kernel function to be maximized in decoding

$$k_{\tau_j, \tau_s}(X', X'') = \mathbb{E}_{c \sim P_\theta(c \mid \tau_j \circ X')} P_\theta(c \mid \tilde{X}) = \sum_{c \in \mathcal{C}} P_\theta(c \mid \tau_j \circ X') P_\theta(c \mid \tau_s \circ X'') \in [0, 1]$$

Posterior agreement kernel for $\tau_j = \tau_s$:

$$k(X', X'') = \sum_{c \in \mathcal{C}} P_\theta(c \mid X') P_\theta(c \mid X'')$$

measures the similarity of X' and X'' that is induced by the posterior distribution of width θ. Essentially, the posterior specifies a sampling procedure how to choose hypotheses c that are highly likely given data X.

8.5 Error analysis

Probability of decoding error:

$$\begin{aligned}
P(\hat{\tau} \neq \tau_s \mid \tau_s) &= P\left(\max_{j \neq s} k_{\tau_j, \tau_s}(X', X'') \geq k(X', X'') \mid \tau_s\right) \\
&\leq \sum_{j \neq s} P\left(k_{\tau_j, \tau_s}(X', X'') \geq k(X', X'') \mid \tau_s\right) && \text{(Union bound)} \\
&= \sum_{j \neq s} \mathbb{E}_{X', X''}\left[P\left(k_{\tau_j, \tau_s}(X', X'') \geq k(X', X'') \mid \tau_s, X', X''\right)\right] \\
&\leq \sum_{j \neq s} \mathbb{E}_{X', X''} \frac{\mathbb{E}_{\tau_j} k_{\tau_j, \tau_s}(X', X'')}{k(X', X'')} && \text{(Markov inequality)}
\end{aligned}$$

Markov inequality applies since since $k(X', X'')$ is a non-negative number under the conditioning.

9 Appendix

Jensen's inequality: if X is a random variable and φ is a convex function, then

$$\varphi(\mathbb{E}[X]) \leq \mathbb{E}[\varphi(X)]$$

An example would be upper-bounding the entropy:

$$H(X) = \mathbb{E}\left[\log_b \frac{1}{P(X)}\right] \leq \log_b \mathbb{E}\left[\frac{1}{P(X)}\right] = \log_b n$$

Parametric statistics is a branch of statistics which assumes that sample data comes from a population that can be adequately modelled by a probability distribution that has a fixed set of parameters. Conversely a **non-parametric model** differs precisely in that the parameter set (or feature set in machine learning) is not fixed and can increase, or even decrease, if new relevant information is collected.

Why convex optimization is easy: a local minimum is global, the local tangent hyperplane is a global lower bound. **The non-convex case is hard**: multiple minima, no local to global inference, NP-hard in some cases.

Sufficient statistic: In statistics, a statistic is sufficient with respect to a statistical model and its associated unknown parameter if "no other statistic that can be calculated from the same sample provides any additional information as to the value of the parameter". In particular, a statistic is sufficient for a family of probability distributions if the sample from which it is calculated gives no additional information than does the statistic, as to which of those probability distributions is that of the population from which the sample was taken.

9.1 Ising model for image de-noising

Given a noisy, binary image $y = (y_1, \ldots, y_n)$ where $y_i \in \{\pm 1\}$ is the value of the i-th pixel, the Ising model tries to find a denoised image $x = (x_1, \ldots, x_n)$ by minimizing the following energy function:

$$E(x|y) = -\frac{\beta}{2} \sum_i \frac{x_i}{|N_i|} \sum_{j \in N_i} x_j - \mu \sum_i x_i y_i$$

where N_i is the set of neighbors of pixel i. For a 2D image the pixel neighbourhood N_i usually involves $|N_i| = 4$ or 8 surrounding pixels.

9.2 Markov Chains

Markov Chain is a set of transition probabilities $p(x \to y)$ with $x, y \in \mathcal{X}^N$, that satisfy

1. **Irreducubility**: for all $x, y \in \mathcal{X}^N$ there is a path $x_0, \ldots, x_n$ of length n, connecting x to y with non-zero probability, i.e. $x = x_0$, $y = x_n$ and $p(x_i \to x_{i+1}) > 0$.

2. **Aperiodicity**: for all x, y there is an integer $n(x, y)$ such that, for any $n > n(x, y)$ there is a path of length n connecting x to y with non-zero probability.
3. **Stationarity** with respect to distribution $p(x)$:

$$\sum_x p(x)p(x \to y) = p(y)$$

Detailed balance implies stationarity

$$p(x)p(x \to y) = p(y)p(y \to x), \text{ or concisely } \pi_i p_{ij} = \pi_j p_{ji}$$

If the Markov chain is a time-homogeneous Markov chain, so that the process is described by a single, time-independent matrix p_{ij}, then the vector π is called a stationary distribution (or invariant measure) if $\forall j \in S$ it satisfies

$$0 \leq \pi_j \leq 1, \ \sum_{j \in S} \pi_j = 1, \ \pi_j = \sum_{i \in S} \pi_i p_{ij}$$

A stationary distribution π is a (row) vector, whose entries are non-negative and sum to 1, is unchanged by the operation of transition matrix $\mathbf{P}$ on it and so is defined by

$$\pi \mathbf{P} = \pi$$

By comparing this definition with that of an eigenvector we see that the two concepts are related and that

$$\pi = \frac{e}{\sum_i e_i}$$

is a normalized multiple of a left eigenvector $\mathbf{e}$ of the transition matrix $\mathbf{P}$ with an eigenvalue of 1.

9.3 Gibbs free energy

Consider a system with configuration space X and a real-valued energy function $E(x)$. The Gibbs Free Energy (which considers the goal of simultaneously "minimizing cost + maximizing entropy") is then defined as a functional over the space of probability distributions $p(x)$ on X

$$G(p) = \mathbb{E}_p[E] - \frac{1}{\beta} H(p) = \sum_{x \in X} p(x)E(x) + \frac{1}{\beta} \sum_{x \in X} p(x) \log p(x)$$

We define Gibbs distribution as

$$p_\beta(x) = \exp[-\beta(E(x) - F(\beta))] \left(= \frac{e^{-\beta E(x)}}{\sum_{x' \in X} e^{-\beta E(x')}} \right)$$

Using this definition, we can write the Gibbs free energy alternatively as

$$G(p) = \frac{1}{\beta} D_{KL}(p \parallel p_\beta) + F(\beta) \geq F(\beta)$$

We can see that $G(p)$ is minimal at $p^* = p_\beta$ with value $G(p^*) = 0$

The free energy is a lower bound on the Gibbs free energy, and the Gibbs distribution is the (ideal) distribution which attains this bound.

$$F(\beta) = -\frac{1}{\beta}\log Z(\beta), \text{ where } Z(\beta) = \sum_{x\in X} e^{-\beta E(x)}$$

Free energy in its explicit form can be written as

$$F(\beta) = -\frac{1}{\beta}\log\left[\sum_{x\in X} e^{-\beta E(x)}\right], \text{ using constraint } \sum_{x\in X} p_\beta(x) = 1$$

9.4 Locally Linear Embedding

Idea: So if our data come from a manifold, we should be able to do a local linear approximation around every part of the manifold, and then smoothly interpolate them together into a single global system. To do dimensionality reduction — to learn the manifold — we want to find these global low-dimensional coordinates. The trick is to do a different linear dimensionality reduction at each point (because locally a manifold looks linear) and then combine these with minimal discrepancy.

Using k-nearest neighborhoods means we take a fine-grained view where there is a lot of data, and a coarse-grained view where there is little data.

LLE tends to handle non-uniform sample densities poorly because there is no fixed unit to prevent the weights from drifting as various regions differ in sample densities.

9.5 Sum-product trick

For $\mathcal{X} = \{x_1, x_2, ..., x_N\}$ and $\mathcal{C} = \{y_1, y_2, ..., y_k\}$ the denominator can be written as the sum

$$Z = (f_{1,1}f_{2,1}...f_{n,1}) + (f_{1,2}f_{2,1}...f_{n,1}) + ... + (f_{1,k}f_{2,k}...f_{n,k})$$

where $f_{i,\nu} = \exp(-||x_i - y_\nu||^2/T)$. This is clearly $O(Nk^N)$ as there are k^N different assignment configurations and computing cost of one assignment is $O(N)$. Sum-product trick transforms this summation into the product

$$Z = (f_{1,1} + f_{1,2} + ... + f_{1,k})(f_{2,1} + f_{2,2} + ... + f_{2,k})...(f_{n,1} + f_{n,2} + ... + f_{n,k})$$

where the complexity drops to $O(NK)$. This trick can only be applied where the cost function is linear in the individual costs, so the assignment of one point to a cluster does not affect the assignments of other data points.

9.6 Simulating procedures

Metropolis-Hastings: Metropolis-Hastings (MH) is a Markov chain Monte Carlo (MCMC) method for sampling probability distributions that are hard to directly sample. Monte Carlo

is an umbrella term defining methods that use random sampling to solve problems, whereas a Markov chain is a sequential model that satisfies Markov property. That property results in past and future being independent, given the present. In MH we first choose an arbitrary sample x_0 and generate new samples x_t by conditioning on the sample from the preceding step of the iteration, x_{t-1}, and build a Markov chain. At every iteration we generate a candidate for the new sample by picking an x' from a proposal distribution $q(x'|x_t)$. We then calculate the acceptance probability for the candidate sample as $A(x_t, x') = min\{1, \frac{p(x')q(x_t|x')}{p(x_t)q(x'|x_t)}\}$. We set $x_{t+1} = x'$ with probability $A(x_t, x')$ and leave it as $x_{t+1} = x_t$ otherwise. Choice of the proposal distribution does not alter the convergence guarantee of the MH algorithm, yet it is very crucial in the sense that it directly affects the speed in which the accepted samples are generated. Algorithm works best when the assumed proposal distribution matches the desired target distribution.

Metropolis: Metropolis algorithm is the name of the original algorithm which was later generalized by Hastings as the MH algorithm. Metropolis algorithm requires a symmetric proposal distribution $q(x_t|x') = q(x'|x_t)$, which simplifies the acceptance probability for x' to $A(x_t, x') = min\{1, \frac{p(x')}{p(x_t)}\}$. An example process is random walk that uses the Gaussian distribution as the proposal.

Simulated Annealing: Simulated annealing is a variant of the MH algorithm that uses a non-homogeneous Markov chain to generate samples. That means the invariant distributions at each iteration does not equate to $p(x)$ but rather to $p_t(x) \propto p^{1/T_t}(x)$, where T_t is the so-called "temperature" at iteration t, where $T_0 = 1$. Similarly, replacing the invariant probabilities in the above, general formula yields an acceptance probability $A(x_t, x') = min\{1, \frac{p^{1/T_t}(x')q(x_t|x')}{p^{1/T_t}(x_t)q(x'|x_t)}\}$. There exists many "cooling schedules" to update the temperature at each iteration. For our purposes exponential schedule produced satisfactory results and other explored schedules failed to deliver better results, so we only illustrate the exponential schedule below. Simulated annealing can be very useful for the global optimization tasks, at it may escape being stuck in local optima. The overall procedure is based on the annealing process in material sciences where the subject material is cooled carefully according to a schedule to produce a structure with desired atttributes.

9.7 Combinatorial optimization

Combinatorial optimization is a topic that consists of finding an optimal object from a finite set of objects. In many such problems, exhaustive search is not tractable. It operates on the domain of those optimization problems, in which the set of feasible solutions is discrete or can be reduced to discrete, and in which the goal is to find the best solution. Some common problems involving combinatorial optimization are the travelling salesman problem ("TSP") and the minimum spanning tree problem ("MST").

Combinatorial optimization deals with efficiently finding a provably strong solution among a finite set of options.

Many problems are NP hard and one therefore relies on heuristics or specialized approaches:

- Relaxation methods

- Branch and bound methods
- Stochastic approaches (e.g. simulated annealing, genetic algorithms)

Alternate optimization converges to a local minimum.

9.8 Simulated vs deterministic annealing

Simulated annealing is stochastic, while deterministic one is, well, deterministic. More precisely, simulated annealing employs a time-dependent Markov Chain (stochastic) to find the minimum, while deterministic annealing has EM procedure (deterministic).

Simulated annealing:

- uses Metropolis Hasting MCMC to sample the Gibbs distribution
- easy to implement but needs a very slow cooling

Deterministic annealing:

- direct minimization of the free energy
- computes expectations of global quantities with respect to the Gibbs distribution
- aggressive cooling is possible, but Z_T computation must be tractable

Deterministic annealing works mainly because of the solution following strategy (homotopy): do not solve a difficult problem from scratch, but rather starting from a good guess of the solution

Trade-off in annealing strategy:

- slow annealing: long running time but no transition is missed
- fast annealing: quicker but with higher risk to miss a transition

Simulated annealing is a stochastic relaxation method that treats an objective function as a system energy, and by analogy with the annealing process of solids, searches for its minimum with decreasing the system temperature. SA searches randomly at a high temperature, but more deterministically at a low temperature.

While decreasing the temperature, SA searches the minimum stochastically at each temperature and thus requires a very long time to find an optimal solution. Hence, though theoretically, it is guaranteed to find the optimal solution, SA is practically an approximation method.

On the contrary, DA consumes less computational time because it searches the local minimum deterministically at each temperature. Furthermore, it should be noticed that, in case that multiple local minima exist at some temperature, DA might not be able to find the minimum. For this reason, even theoretically, DA is not guaranteed to find the optimal solution.

Wikipedia:

- For problems where finding an approximate global optimum is more important than finding a precise local optimum in a fixed amount of time, simulated annealing may be preferable to alternatives such as gradient descent.

- This notion of slow cooling implemented in the simulated annealing algorithm is interpreted as a slow decrease in the probability of accepting worse solutions as the solution space is explored. Accepting worse solutions is a fundamental property of metaheuristics because it allows for a more extensive search for the global optimal solution.

9.9 Deterministic Annealing

Certain chemical systems can be driven to their low-energy states by annealing, which is a gradual reduction of temperature, spending a long time at the vicinity of the phase transition points. In the corresponding probabilistic framework, a Gibbs distribution is defined over the set of all possible configurations which assigns higher probability to configurations of lower energy. This distribution is parametrized by the temperature, and as the temperature is lowered it becomes more discriminating (concentrating most of the probability in a smaller subset of low-energy configurations). At the limit of low temperature it assigns nonzero probability only to global minimum configurations.

Jaynes's maximum entropy principle: of all the probability distributions that satisfy a given set of constraints, choose the one that maximizes the entropy. The informal justification is that while this choice agrees with what is known (the given constraints), it maintains maximum uncertainty with respect to everything else. Had we chosen another distribution satisfying the constraints, we would have reduced the uncertainty and would have therefore implicitly made some extra restrictive assumption.

At infinite, these are uniform distributions, i.e., each input vector is equally associated with all clusters. These are extremely fuzzy associations. As is lowered, the distributions become more discriminating and the associations less fuzzy. At zero temperature the classification is hard with each input sample assigned to the nearest codevector with probability one.

In summary, the DA method performs annealing as it maintains the free energy at its minimum (thermal equilibrium) while gradually lowering the temperature; and it is deterministic because it minimizes the free energy directly rather than via stochastic simulation of the system dynamics.

It should also be noted that at the limit of low temperature ($T = 0$), both the unconstrained DA method and the mass-constrained DA method, converge to the same descent process, namely, GLA. This is because the association probabilities of the two DA methods are identical at the limit, and they assign each data point to the nearest codevector with probability one. The difference between the two is in their behavior at intermediate , where the mass-constrained clustering method takes the cluster populations into account.

9.10 Constant Shift Embedding

1. Get D (assume 0 diagonal)
2. Symmetrize D
3. Centralize D: $D^c \leftarrow QDQ$
4. Get similarities: $S^c \leftarrow -\frac{1}{2}D^c$

5. Shift similarities: $\tilde{S} \leftarrow S^c - \lambda_n(S^c)I_n$
   where λ_n denotes the minimal eigenvalue
6. $\tilde{D}_{(i,j)} \leftarrow \tilde{S}_{(i,i)} + \tilde{S}_{(j,j)} - 2\tilde{S}_{(i,j)}$
7. $\tilde{S}^c \leftarrow Q\tilde{S}Q$
8. $\tilde{D}^c \leftarrow (-2)\tilde{S}^c$
9. Assert $\tilde{D}^c == Q\tilde{D}Q$

Shifted D contains squared Euclidean distances in the high dimensional space.

9.11 Shannon capacity etc.

The Shannon theorem states that given a noisy channel with channel capacity C and information transmitted at a rate R, then if $R < C$ there exist codes that allow the probability of error at the receiver to be made arbitrarily small. This means that, theoretically, it is possible to transmit information nearly without error at any rate below a limiting rate, C.

The converse is also important. If $R > C$, an arbitrarily small probability of error is not achievable. All codes will have a probability of error greater than a certain positive minimal level, and this level increases as the rate increases. So, information cannot be guaranteed to be transmitted reliably across a channel at rates beyond the channel capacity. The theorem does not address the rare situation in which rate and capacity are equal.

Simple schemes such as "send the message 3 times and use a best 2 out of 3 voting scheme if the copies differ" are inefficient error-correction methods, unable to asymptotically guarantee that a block of data can be communicated free of error. Advanced techniques such as Reed–Solomon codes and, more recently, low-density parity-check (LDPC) codes and turbo codes, come much closer to reaching the theoretical Shannon limit, but at a cost of high computational complexity.

Source coding theorem: N i.i.d. random variables each with entropy $H(X)$ can be compressed into more than $NH(X)$ bits with negligible risk of information loss, as $N \to \infty$; but conversely, if they are compressed into fewer than $NH(X)$ bits it is virtually certain that information will be lost.

Coin toss example: Counter-intuitively, the most likely sequence is often not a member of the typical set. For example, suppose that X is an i.i.d Bernoulli random variable with $p(0) = 0.1$ and $p(1) = 0.9$. In n independent trials, since $p(1) > p(0)$, the most likely sequence of outcome is the sequence of all 1's, $(1, 1, ..., 1)$. Here the entropy of X is $H(X) = 0.469$, while

$$-\frac{1}{n}\log_2 p\left(x^{(n)} = (1, 1, \ldots, 1)\right) = -\frac{1}{n}\log_2(0.9^n) = 0.152$$

So this sequence is not in the typical set because its average logarithmic probability cannot come arbitrarily close to the entropy of the random variable X no matter how large we take the value of n.