1 Introduction to Information Theory and Data Science

Miguel R. D. Rodrigues, Stark C. Draper, Waheed U. Bajwa, and Yonina C. Eldar

Summary

The field of information theory – dating back to 1948 – is one of the landmark intellectual achievements of the twentieth century. It provides the philosophical and mathematical underpinnings of the technologies that allow accurate representation, efficient compression, and reliable communication of sources of data. A wide range of storage and transmission infrastructure technologies, including optical and wireless communication networks, the internet, and audio and video compression, have been enabled by principles illuminated by information theory. Technological breakthroughs based on information-theoretic concepts have driven the "information revolution" characterized by the anywhere and anytime availability of massive amounts of data and fueled by the ubiquitous presence of devices that can capture, store, and communicate data.

The existence and accessibility of such massive amounts of data promise immense opportunities, but also pose new challenges in terms of how to extract useful and actionable knowledge from such data streams. Emerging data-science problems are different from classical ones associated with the transmission or compression of information in which the semantics of the data was unimportant. That said, we are starting to see that information-theoretic methods and perspectives can, in a new guise, play important roles in understanding emerging data-science problems. The goal of this book is to explore such new roles for information theory and to understand better the modern interaction of information theory with other data-oriented fields such as statistics and machine learning.

The purpose of this chapter is to set the stage for the book and for the upcoming chapters. We first overview classical information-theoretic problems and solutions. We then discuss emerging applications of information-theoretic methods in various data-science problems and, where applicable, refer the reader to related chapters in the book. Throughout this chapter, we highlight the perspectives, tools, and methods that play important roles in classic information-theoretic paradigms and in emerging areas of data science. Table 1.1 provides a summary of the different topics covered in this chapter and highlights the different chapters that can be read as a follow-up to these topics.

Table 1.1. Major topics covered in this chapter and their connections to other chapters

Section(s)	Topic	Related chapter(s)
1.1–1.4	An introduction to information theory	15
1.6	Information theory and data acquisition	2–4, 6, 16
1.7	Information theory and data representation	5, 11
1.8	Information theory and data analysis/processing	6–16

1.1 Classical Information Theory: A Primer

Claude Shannon's 1948 paper "A mathematical theory of communications," *Bell Systems Technical Journal*, July/Oct. 1948, laid out a complete architecture for digital communication systems [1]. In addition, it articulated the philosophical decisions for the design choices made. *Information theory*, as Shannon's framework has come to be known, is a beautiful and elegant example of engineering science. It is all the more impressive as Shannon presented his framework decades before the first digital communication system was implemented, and at a time when digital computers were in their infancy.

Figure 1.1 presents a general schematic of a digital communication system. This figure is a reproduction of Shannon's "Figure 1" from his seminal paper. Before 1948 no one had conceived of a communication system in this way. Today nearly all digital communication systems obey this structure.

The flow of information through the system is as follows. An *information source* first produces a random message that a transmitter wants to convey to a destination. The message could be a word, a sentence, or a picture. In information theory, all information sources are modeled as being sampled from a set of possibilities according to some probability distribution. Modeling information sources as stochastic is a key aspect of Shannon's approach. It allowed him to quantify uncertainty as the lack of knowledge and reduction in uncertainty as the gaining of knowledge or "information."

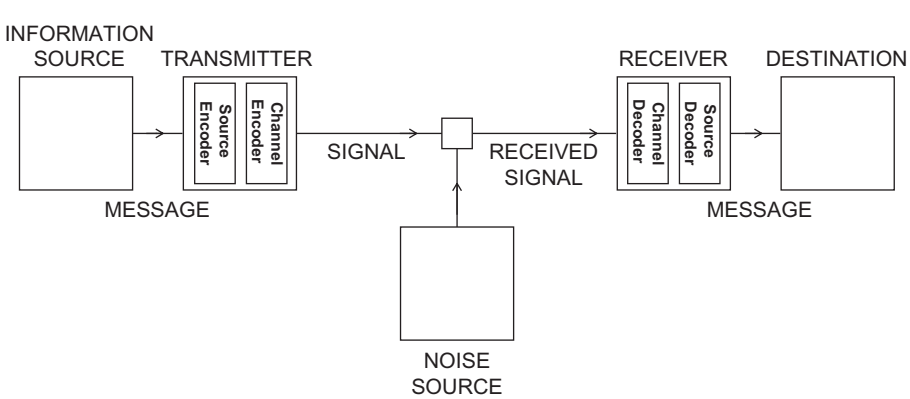

Figure 1.1 Reproduction of Shannon's Figure 1 in [1] with the addition of the source and channel encoding/decoding blocks. In Shannon's words, this is a "schematic diagram of a general communication system."

The message is then fed into a transmission system. The transmitter itself has two main sub-components: the *source encoder* and the *channel encoder*. The source encoder converts the message into a sequence of 0s and 1s, i.e., a bit sequence. There are two classes of source encoders. *Lossless source coding* removes predictable redundancy that can later be recreated. In contrast, *lossy* source coding is an irreversible process wherein some distortion is incurred in the compression process. Lossless source coding is often referred to as *data compression* while lossy coding is often referred to as *rate-distortion* coding. Naturally, the higher the distortion the fewer the number of bits required.

The bit sequence forms the data payload that is fed into a channel encoder. The output of the channel encoder is a signal that is transmitted over a noisy communication medium. The purpose of the *channel code* is to convert the bits into a set of possible signals or *codewords* that can be reliably recovered from the noisy received signal.

The communication medium itself is referred to as the *channel*. The channel can model the physical separation of the transmitter and receiver. It can also, as in data storage, model separation in time.

The destination observes a signal that is the output of the communication channel. Similar to the transmitter, the receiver has two main components: a *channel decoder* and a *source decoder*. The former maps the received signal into a bit sequence that is, one hopes, the same as the bit sequence produced by the transmitter. The latter then maps the estimated bit sequence to an estimate of the original message.

If lossless compression is used, then an apt *measure of performance* is the probability that the message estimate at the destination is not equal to the original message at the transmitter. If lossy compression (rate distortion) is used, then other measures of goodness, such as mean-squared error, are more appropriate.

Interesting questions addressed by information theory include the following.

1. Architectures
   - What trade-offs in performance are incurred by the use of the architecture detailed in Figure 1.1?
   - When can this architecture be improved upon; when can it not?
2. Source coding: lossless data compression
   - How should the information source be modeled; as stochastic, as arbitrary but unknown, or in some other way?
   - What is the shortest bit sequence into which a given information source can be compressed?
   - What assumptions does the compressor work under?
   - What are basic compression techniques?
3. Source coding: rate-distortion theory
   - How do you convert an analog source into a digital bitstream?
   - How do you reconstruct/estimate the original source from the bitstream?
   - What is the trade-off involved between the number of bits used to describe a source and the distortion incurred in reconstruction of the source?

4. Channel coding
   - How should communication channels be modeled?
   - What throughput, measured in bits per second, at what reliability, measured in terms of probability of error, can be achieved?
   - Can we quantify fundamental limits on the realizable trade-offs between throughput and reliability for a given channel model?
   - How does one build computationally tractable channel coding systems that "saturate" the fundamental limits?
5. Multi-user information theory
   - How do we design systems that involve multiple transmitters and receivers?
   - How do many (perhaps correlated) information sources and transmission channels interact?

The decades since Shannon's first paper have seen fundamental advances in each of these areas. They have also witnessed information-theoretic perspectives and thinking impacting a number of other fields including security, quantum computing and communications, and cryptography. The basic theory and many of these developments are documented in a body of excellent texts, including [2–9]. Some recent advances in network information theory, which involves multiple sources and/or multiple destinations, are also surveyed in Chapter 15. In the next three sections, we illustrate the basics of information-theoretic thinking by focusing on simple (point-to-point) binary sources and channels. In Section 1.2, we discuss the compression of binary sources. In Section 1.3, we discuss channel coding over binary channels. Finally, in Section 1.4, we discuss computational issues, focusing on linear codes.

1.2 Source Coding: Near-Lossless Compression of Binary Sources

To gain a feel for the tools and results of classical information theory consider the following lossless source coding problem. One observes a length-n string of random coin flips, $X_1, X_2, \ldots, X_n$, each $X_i \in \{\text{heads}, \text{tails}\}$. The flips are independent and identically distributed with $\mathbb{P}(X_i = \text{heads}) = p$, where $0 \leq p \leq 1$ is a known parameter. Suppose we want to map this string into a bit sequence to store on a computer for later retrieval. Say we are going to assign a *fixed* amount of memory to store the sequence. How much memory must we allocate?

Since there are 2^n possible sequences, all of which could occur if p is not equal to 0 or 1, if we use n bits we can be 100% certain we could index any heads/tails sequence that we might observe. However, certain sequences, while possible, are much less likely than others. Information theory exploits such non-uniformity to develop systems that can trade off between efficiency (the storage of fewer bits) and reliability (the greater certainty that one will later be able to reconstruct the observed sequence). In the following, we accept some (arbitrarily) small probability $\epsilon > 0$ of observing a sequence that we

choose not to be able to store a description of.[1] One can think of ϵ as the probability of the system failing. Under this assumption we derive bounds on the number of bits that need to be stored.

1.2.1 Achievability: An Upper Bound on the Rate Required for Reliable Data Storage

To figure out which sequences we may choose not to store, let us think about the statistics. In expectation, we observe np heads. Of the 2^n possible heads/tails sequences there are $\binom{n}{np}$ sequences with np heads. (For the moment we ignore non-integer effects and deal with them later.) There will be some variability about this mean but, at a minimum, we must be able to store all these expected realizations since these realizations all have the same probability. While $\binom{n}{np}$ is the cardinality of the set, we prefer to develop a good approximation that is more amenable to manipulation. Further, rather than counting cardinality, we will count the log-cardinality. This is because given k bits we can index 2^k heads/tails source sequences. Hence, it is the exponent in which we are interested.

Using Stirling's approximation to the factorial, $\log_2 n! = n\log_2 n - (\log_2 e)n + O(\log_2 n)$, and ignoring the order term, we have

$$\log\binom{n}{np} \simeq n\log_2 n - n(1-p)\log_2(n(1-p)) - np\log_2(np) \tag{1.1}$$

$$= n\log_2\left(\frac{1}{1-p}\right) + np\log_2\left(\frac{1-p}{p}\right)$$

$$= n\left[(1-p)\log_2\left(\frac{1}{1-p}\right) + p\log_2\left(\frac{1}{p}\right)\right]. \tag{1.2}$$

In (1.1), the $(\log_2 e)n$ terms have canceled and the term in square brackets in (1.2) is called the *(binary) entropy*, which we denote as $H_{\mathrm{B}}(p)$, so

$$H_{\mathrm{B}}(p) = -p\log_2 p - (1-p)\log_2(1-p), \tag{1.3}$$

where $0 \le p \le 1$ and $0\log 0 = 0$. The binary entropy function is plotted in Fig. 1.2 within Section 1.3. One can compute that when $p = 0$ or $p = 1$ then $H_{\mathrm{B}}(0) = H_{\mathrm{B}}(1) = 0$. The interpretation is that, since there is only one all-tails and one all-heads sequence, and we are quantifying log-cardinality, there is only one sequence to index in each case so $\log_2(1) = 0$. In these cases, we *a priori* know the outcome (respectively, all the heads or all tails) and so do not need to store any bits to describe the realization. On the other hand, if the coin is fair then $p = 0.5$, $H_{\mathrm{B}}(0.5) = 1$, $\binom{n}{n/2} \simeq 2^n$, and we must use n bits of storage. In other words, on an exponential scale almost all binary sequences are 50% heads and 50% tails. As an intermediate value, if $p = 0.11$ then $H_{\mathrm{B}}(0.11) \simeq 0.5$.

[1] In source coding, this is termed *near-lossless* source coding as the arbitrarily small ϵ bounds the probability of system failure and thus loss of the original data. In the *variable-length* source coding paradigm, one stores a variable amount of bits per sequence, and minimizes the expected number of bits stored. We focus on the near-lossless paradigm as the concepts involved more closely parallel those in channel coding.

The operational upshot of (1.2) is that if one allocates $nH_{\mathrm{B}}(p)$ bits then basically all expected sequences can be indexed. Of course, there are caveats. First, np need not be integer. Second, there will be variability about the mean. To deal with both, we allocate a few more bits, $n(H_{\mathrm{B}}(p)+\delta)$ in total. We use these bits not just to index the expected sequences, but also the *typical sequences*, those sequences with empirical entropy close to the entropy of the source.[2] In the case of coin flips, if a particular sequence consists of n_{H} heads (and $n-n_{\mathrm{H}}$ tails) then we say that the sequence is "typical" if

$$H_{\mathrm{B}}(p)-\delta \leq \left[\frac{n_{\mathrm{H}}}{n}\log_2\left(\frac{1}{p}\right)+\frac{n-n_{\mathrm{H}}}{n}\log_2\left(\frac{1}{1-p}\right)\right] \leq H_{\mathrm{B}}(p)+\delta. \tag{1.4}$$

It can be shown that the cardinality of the set of sequences that satisfies condition (1.4) is upper-bounded by $2^{n(H_{\mathrm{B}}(p)+\delta)}$. Therefore if, for instance, one lists the typical sequences lexicographically, then any typical sequence can be described using $n(H_{\mathrm{B}}(p)+\delta)$ bits. One can also show that for any $\delta>0$ the probability of the source *not* producing a typical sequence can be upper-bounded by any $\epsilon>0$ as n grows large. This follows from the law of large numbers. As n grows the distribution of the fraction of heads in the realized source sequence concentrates about its expectation. Therefore, as long as n is sufficiently large, and as long as $\delta>0$, any $\epsilon>0$ will do. The quantity $H_{\mathrm{B}}(p)+\delta$ is termed the *storage "rate" R*. For this example $R=H_{\mathrm{B}}(p)+\delta$. The rate is the amount of memory that must be made available per source symbol. In this case, there were n symbols (n coin tosses), so one normalizes $n(H_{\mathrm{B}}(p)+\delta)$ by n to get the rate $H_{\mathrm{B}}(p)+\delta$.

The above idea can immediately be extended to independent and identically distributed (i.i.d.) finite-alphabet (and more general) sources as well. The general definition of the *entropy of a finite-alphabet random variable X* with probability mass function (p.m.f.) p_X is

$$H(X)=-\sum_{x\in\mathcal{X}} p_X(x)\log_2 p_X(x), \tag{1.5}$$

where "finite-alphabet" means the sample space $\mathcal{X}$ is finite.

Regardless of the distribution (binary, non-binary, even non-i.i.d.), the simple coin-flipping example illustrates one of the central tenets of information theory. That is, to focus one's design on what is likely to happen, i.e., the typical events, rather than on worst-case events. The partition of events into typical and atypical is, in information theory, known as the *asymptotic equipartition property* (AEP). In a nutshell, the simplest form of the AEP says that for long i.i.d. sequences one can, up to some arbitrarily small probability ϵ, partition all possible outcomes into two sets: the typical set and the atypical set. The probability of observing an event in the typical set is at least $1-\epsilon$. Furthermore, on an exponential scale all typical sequences are of equal probability. Designing for typical events is a hallmark of information theory.

[2] In the literature, these are termed the "weakly" typical sequences. There are other definitions of typicality that differ in terms of their mathematical use. The overarching concept is the same.

1.2.2 Converse: A Lower Bound on the Rate Required for Reliable Data Storage

A second hallmark of information theory is the emphasis on developing bounds. The source coding scheme described above is known as an *achievability result*. Achievability results involve describing an operational system that can, in principle, be realized in practice. Such results provide *(inner) bounds* on what is possible. The performance of the best system is at least this good. In the above example, we developed a source coding technique that delivers high-reliability storage and requires a rate of $H(X)+\delta$, where both the error ϵ and the slack δ can be arbitrarily small if n is sufficiently large.

An important coupled question is how much (or whether) we can reduce the rate further, thereby improving the efficiency of the scheme. In information theory, *outer bounds* on what is possible – e.g., showing that if the encoding rate is too small one cannot guarantee a target level of reliability – are termed *converse results*.

One of the key lemmas used in converse results is *Fano's inequality* [7], named for Robert Fano. The statement of the inequality is as follows: For any pair of random variables $(U,V) \in \mathcal{U}\times\mathcal{V}$ jointly distributed according to $p_{U,V}(\cdot,\cdot)$ and for any estimator $G:\mathcal{U}\to\mathcal{V}$ with probability of error $P_{\mathrm{e}} = \Pr[G(U)\neq V]$,

$$H(V|U) \leq H_{\mathrm{B}}(P_{\mathrm{e}}) + P_{\mathrm{e}}\log_2(|\mathcal{V}|-1). \tag{1.6}$$

On the left-hand side of (1.6) we encounter the *conditional entropy* $H(V|U)$ of the joint p.m.f. $p_{U,V}(\cdot,\cdot)$. We use the notation $H(V|U=u)$ to denote the entropy in V when the realization of the random variable U is set to $U=u$. Let us name this the "pointwise" conditional entropy, the value of which can be computed by applying our formula for entropy (1.5) to the p.m.f. $p_{V|U}(\cdot|u)$. The conditional entropy is the expected pointwise conditional entropy:

$$H(V|U) = \sum_{u\in\mathcal{U}} p_U(u)H(V|U=u) = \sum_{u\in\mathcal{U}} p_U(u)\left[\sum_{v\in\mathcal{V}} p_{V|U}(v|u)\log_2\left(\frac{1}{p_{V|U}(v|u)}\right)\right]. \tag{1.7}$$

Fano's inequality (1.6) can be interpreted as a bound on the ability of any hypothesis test function G to make a (single) correct guess of the realization of V on the basis of its observation of U. As the desired error probability $P_{\mathrm{e}}\to 0$, both terms on the right-hand side go to zero, implying that the conditional entropy must be small. Conversely, if the left-hand side is not too small, that asserts a non-zero lower bound on P_{e}. A simple explicit bound is achieved by upper-bounding $H_{\mathrm{B}}(P_{\mathrm{e}})$ as $H_{\mathrm{B}}(P_{\mathrm{e}})\leq 1$ and rearranging to find that $P_{\mathrm{e}} \geq (H(V|U)-1)/\log_2(|\mathcal{V}|-1)$.

The usefulness of Fano's inequality stems, in part, from the weak assumptions it makes. One can apply Fano's inequality to any joint distribution. Often identification of an applicable joint distribution is part of the creativity in the use of Fano's inequality. For instance in the source coding example above, one takes V to be the stored data sequence, so $|\mathcal{V}| = 2^{n(H_{\mathrm{B}}(p)+\delta)}$, and U to be the original source sequence, i.e., $U=X^n$. While we do not provide the derivation herein, the result is that to achieve an error probability of at most P_{e} the storage rate R is lower-bounded by $R \geq H(X) - P_{\mathrm{e}}\log_2|\mathcal{X}| - H_{\mathrm{B}}(P_{\mathrm{e}})/n$,

where $|\mathcal{X}|$ is the source alphabet size; for the binary example $|\mathcal{X}| = 2$. As we let $P_e \to 0$ we see that the lower bound on the achievable rate is $H(X)$ which, letting $\delta \to 0$, is also our upper bound. Hence we have developed an operational approach to data compression where the rate we achieve matches the converse bound.

We now discuss the interaction between achievability and converse results. As long as the compression rate $R > H(X)$ then, due to concentration in measure, in the achievability case the failure probability $\epsilon > 0$ and rate slack $\delta > 0$ can both be chosen to be arbitrarily small. Concentration of measure occurs as the *blocklength* n becomes large. In parallel with n getting large, the total number of bits stored nR also grows.

The entropy $H(X)$ thus specifies a boundary between two regimes of operation. When the rate R is larger than $H(X)$, achievability results tell us that arbitrarily reliable storage is possible. When R is smaller than $H(X)$, converse results imply that reliable storage is not possible. In particular, rearranging the converse expression and once again noting that $H_B(P_e) \leq 1$, the error probability can be lower-bounded as

$$P_e \geq \frac{H(X) - R - 1/n}{\log_2 |\mathcal{X}|}. \tag{1.8}$$

If $R < H(X)$, then for n sufficiently large P_e is bounded away from zero.

The entropy $H(X)$ thus characterizes a *phase transition* between one state, the possibility of reliable data storage, and another, the impossibility. Such sharp information-theoretic phase transitions also characterize classical information-theoretic results on data transmission which we discuss in the next section, and applications of information-theoretic tools in the data sciences which we turn to later in the chapter.

1.3 Channel Coding: Transmission over the Binary Symmetric Channel

Shannon applied the same mix of ideas (typicality, entropy, conditional entropy) to solve the, perhaps at first seemingly quite distinct, problem of reliable and efficient digital communications. This is typically referred to as Shannon's "channel coding" problem in contrast to the "source coding" problem already discussed.

To gain a sense of the problem we return to the simple binary setting. Suppose our source coding system has yielded a length-k string of "information bits." For simplicity we assume these bits are randomly distributed as before, i.i.d. along the sequence, but are now fair; i.e., each is equally likely to be "0" or a "1." The objective is to convey this sequence over a communications channel to a friend. Importantly we note that, since the bits are uniformly distributed, our result on source coding tells us that no further compression is possible. Thus, uniformity of message bits is a worst-case assumption.

The channel we consider is the *binary symmetric channel* (BSC). We can transmit binary symbols over a BSC. Each input symbol is conveyed to the destination, but not entirely accurately. The binary symmetric channel "flips" each channel input symbol ($0 \to 1$ or $1 \to 0$) with probability p, $0 \leq p \leq 1$. Flips occur independently. The challenge is for the destination to deduce, one hopes with high accuracy, the k information bits

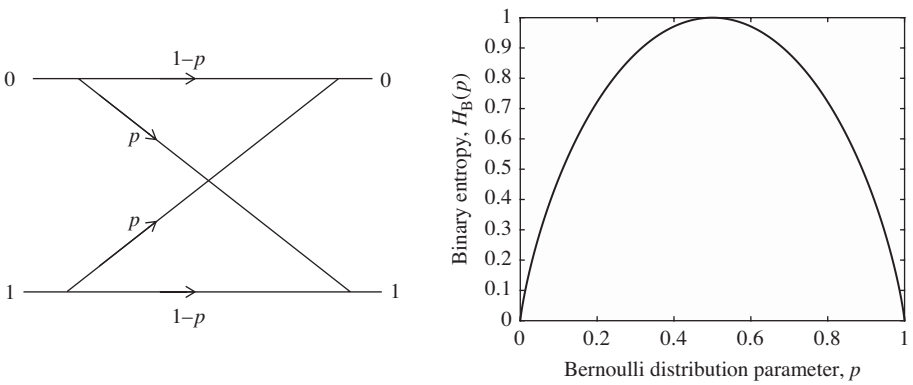

Figure 1.2 On the left we present a graphical description of the binary symmetric channel (BSC). Each transmitted binary symbol is represented as a 0 or 1 input on the left. Each received binary observation is represented by a 0 or 1 output on the right. The stochastic relationship between inputs and outputs is represented by the connectivity of the graph where the probability of transitioning each edge is represented by the edge label p or $1-p$. The channel is "symmetric" due to the symmetries in these transition probabilities. On the right we plot the binary entropy function $H_B(p)$ as a function of p, $0 \le p \le 1$. The capacity of the BSC is $C_{BSC} = 1 - H_B(p)$.

transmitted. Owing to the symbol flipping noise, we get some slack; we transmit $n \ge k$ binary channel symbols. For efficiency's sake, we want n to be as close to k as possible, while meeting the requirement of high reliability. The ratio k/n is termed the "rate" of communication. The length-n binary sequence transmitted is termed the "codeword." This "bit flipping" channel can be used, e.g., to model data storage errors in a computer memory. A graphical representation of the BSC is depicted in Fig. 1.2.

1.3.1 Achievability: A Lower Bound on the Rate of Reliable Data Communication

The idea of channel coding is analogous to human-evolved language. The length-k string of information bits is analogous to what we think, i.e., the concept we want to impart to the destination. The length-n codeword string of binary channel symbols is analogous to what we say (the sentence). There is redundancy in spoken language that makes it possible for spoken language to be understood in noisy (albeit not too noisy) situations. We analogously engineer redundancy into what a computer transmits in order to be able to combat the expected (the typical!) noise events. For the BSC those would be the expected bit-flip sequences.

We now consider the noise process. For any chosen length-n codeword there are about $\binom{n}{np}$ typical noise patterns which, using the same logic as in our discussion of source compression, is a set of roughly $2^{nH_B(p)}$ patterns. If we call X^n the codeword and E^n the noise sequence, then what the receiver measures is $Y^n = X^n + E^n$. Here addition is vector addition over $\mathbb{F}_2$, i.e., coordinate-wise, where the addition of two binary symbols is implemented using the *XOR* operator. The problem faced by the receiver is to identify the transmitted codeword. One can imagine that if the possible codewords are far apart in the sense that they differ in many entries (i.e., their *Hamming distance* is large) then the

receiver will be less likely to make an error when deciding on the transmitted codeword. Once such a codeword estimate has been made it can then be mapped back to the length-k information bit sequence. A natural decoding rule, in fact the *maximum-likelihood* rule, is for the decoder to pick the codeword closest to Y^n in terms of Hamming distance.

The design of the codebook (analogous to the choice of grammatically correct – and thus allowable – sentences in a spoken language) is a type of probabilistic packing problem. The question is, how do we select the set of codewords so that the probability of a decoding error is small? We can develop a simple upper bound on how large the set of reliably decodable codewords can be. There are 2^n possible binary output sequences. For any codeword selected there are roughly $2^{nH_{\mathrm{B}}(p)}$ typical output sequences, each associated with a typical noise sequence, that form a *noise ball* centered on the codeword. If we were simply able to divide up the output space into disjoint sets of cardinality $2^{nH_{\mathrm{B}}(p)}$, we would end up with $2^n/2^{nH_{\mathrm{B}}(p)} = 2^{n(1-H_{\mathrm{B}}(p))}$ distinct sets. This *sphere-packing* argument tells us that the best we could hope to do would be to transmit this number of distinct codewords reliably. Thus, the number of information bits k would equal $n(1 - H_{\mathrm{B}}(p))$. Once we normalize by the number n of channels uses we get a transmission rate of $1 - H_{\mathrm{B}}(p)$.

Perhaps quite surprisingly, as n gets large, $1 - H_{\mathrm{B}}(p)$ is the supremum of achievable rates at (arbitrarily) high reliability. This is the Shannon capacity $C_{\mathrm{BSC}} = 1 - H_{\mathrm{B}}(p)$. The result follows from the law of large numbers, which can be used to show that the typical noise balls concentrate. Shannon's proof that one can actually find a configuration of codewords while keeping the probability of decoding error small was an early use of the *probabilistic method*. For any rate $R = C_{\mathrm{BSC}} - \delta$, where $\delta > 0$ is arbitrarily small, a randomized choice of the positioning of each codeword will with high probability, yield a code with a small probability of decoding error. To see the plausibility of this statement we revisit the sphere-packing argument. At rate $R = C_{\mathrm{BSC}} - \delta$ the 2^{nR} codewords are each associated with a typical noise ball of $2^{nH_{\mathrm{B}}(p)}$ sequences. If the noise balls were all (in the worst case) disjointed, this would be a total of $2^{nR}2^{nH_{\mathrm{B}}(p)} = 2^{n(1-H_{\mathrm{B}}(p)-\delta)+nH_{\mathrm{B}}(p)} = 2^{n(1-\delta)}$ sequences. As there are 2^n binary sequences, the fraction of the output space taken up by the union of typical noise spheres associated with the codewords is $2^{n(1-\delta)}/2^n = 2^{-n\delta}$. So, for any $\delta > 0$ fixed, as the blocklength $n \to \infty$, only an exponentially disappearing fraction of the output space is taken up by the noise balls. By choosing the codewords independently at random, each uniformly chosen over all length-n binary sequences, one can show that the expected (over the choice of codewords and channel noise realization) average probability of error is small. Hence, at least one codebook exists that performs at least as well as this expectation.

While Shannon showed the existence of such a code (actually a sequence of codes as $n \to \infty$), it took another half-century for researchers in error-correction coding to find asymptotically optimal code designs and associated decoding algorithms that were computationally tractable and therefore implementable in practice. We discuss this computational problem and some of these recent code designs in Section 1.4.

While the above example is set in the context of a binary-input and binary-output channel model, the result is a prototype of the result that holds for *discrete memoryless channels*. A discrete memoryless channel is described by the conditional distribution