Chapter 1: Introduction to Lossless Compression

Satyadev Nandakumar

August 2, 2017

We focus on properties of lossless compression, which is typically used to compress text. In the course of this discussion, we will use a finite input alphabet Σ, and a finite output alphabet Γ, which may be different from the input alphabet. We start by looking at a few basic compression techniques and analyse them before proceeding to a more formal treatment of sophisticated techniques. A few preliminary notations and definitions are in order.

We use Σ^* to denote the set of finite strings drawn from the alphabet, and Σ^∞, to denote the set of infinite sequences. For a finite string w, the notation $|w|$ denotes the length of the string. If A is a finite set, then $|A|$ denotes the number of elements of A. For a non-negative integer n, the set Σ^n consists of $|\Sigma|^n$ strings of length n. For most of this course, we will adopt $\Sigma = \Gamma = \{0, 1\}$, the binary alphabet.

We start with a very broad definition of what constitutes a lossless compressor - somewhat counter-intuitively, we do not insist that the compressor's output must be shorter than the input. Immediately after the definition, we justify this.

Definition 1. A *lossless compressor* is a 1-1 function $C : \Sigma^* \to \Sigma^*$.

An immediate consequence of this definition is that any lossless compressor leaves most strings uncompressed - *i.e.* for most strings w, we have $|C(w)| \geq |w|$. The following counting argument shows a weaker version of this statement.

Lemma 2. *Let n and k be positive integers, where $k < n$, $\Sigma = \{0, 1\}$ and $C : \Sigma^* \to \Sigma^*$ be a lossless compressor. Then*

$$|\{x \in \Sigma^n \mid |C(x)| \leq n - k\}| \;\leq\; 2^{n-k+1} - 1.$$

Thus an additive loss in the compression length incurs a multiplicative loss in the number of strings which can be compressed.

Proof. The set of codewords of length at most $n - k$ is $\Sigma^{\leq n-k} \cap C(\Sigma^*)$. This has at most 2^{n-k+1} strings. Since C is 1-1, this set has at most 2^{n-k+1} preimage strings. In particular, there are at most 2^{n-k+1} strings of length n which have compressed forms with length $\leq n - k$. □

The above lemma shows immediately that there must be strings x such that $|x| \leq |C(x)|$. We have the following corollary.

Corollary 3. *Let $\Sigma = \{0, 1\}$, and $C : \Sigma^* \to \Sigma^*$ be a lossless compressor. Then for all lengths $n \geq 1$, there are strings $x \in \Sigma^n$ such that $|x| \leq |C(x)|$.*

Proof. Let $n \geq 1$. Then

$$|\{z \in \Sigma^n \mid |C(z)| \leq n - 1\}| \leq \left|\Sigma^{\leq n-1}\right| = 2^{n-1+1} - 1 = 2^n - 1,$$

hence there is a string $x \in \Sigma^n$ with $|C(x)| \geq n$. □

Thus it is senseless to examine the “worst-case” performance of a data compressor. A lossless data-compressor has infinitely many strings which it cannot compress at all, and may in fact, expand an input to a greater length. What we study is the expected-codelength with respect to various probability distributions on the input.

Even though the above lemma says that a significant fraction of strings cannot be compressed much, lossless compressors may still be useful, as we describe now.

It is possible that a specific lossless compressor may compress a small but useful fraction of all possible strings.

1. First, we may heuristically say that all “meaningful” strings, *e.g.* English text, `C` source code etc. can be compressed by some specific compressor we have in hand.

2. Another way is to define a probability distribution, prove that “meaningful” strings have high probability under this distribution, and then show that the expected codelength according to this distribution is very small.

The first method is practically useful, but does not provide any *quantitative bounds* on the performance of the compressor. In order to obtain such bounds, we usually resort to, at least attempt, the second approach. Computing the expected code length is often tedious for sophisticated compressors and arbitrary distributions. Nevertheless, we now illustrate this approach on a very simple compression technique.