UNIVERSIDAD DE GRANADA

MUTUAL INFORMATION IN UNSUPERVISED MACHINE LEARNING

FRANCISCO JAVIER SÁEZ MALDONADO

Bachelor's Thesis

Computer Science and Mathematics

Tutor

Nicolás Pérez de la Blanca Capilla

FACULTY OF SCIENCE

H.T.S. OF COMPUTER ENGINEER AND TELECOMMUNICATIONS

Granada, Tuesday 29th June, 2021

ABSTRACT

Abstract

CONTENTS

Part I

BASIC CONCEPTS

1 PROBABILITY

Underneath each experiment involving any grade of uncertainty there is a *random variable*. This is no more than a *measurable* function between two *measurable spaces*. A probability space is composed by three elements: $(\Omega, \mathscr{A}, \mathcal{P})$. We will define those concepts one by one.

1.1 BASIC NOTIONS

Definition 1. Let Ω be a non empty sample space. $\mathscr{A}$ is a *$\sigma-$algebra* over Ω if it is a family of subsets of Ω that verify that the emptyset is in $\mathscr{A}$, and it is closed under complementation and countable unions. That is:

- $\emptyset \in \mathscr{A}$.
- If $A \in \mathscr{A}$, then $\Omega \backslash A \in \mathscr{A}$.
- If $\{A_i\}_{i \in \mathbb{N}} \in A$ is a numerable family of $\mathscr{A}$ subsets, then $\cup_{i \in \mathbb{N}} A_i \in \mathscr{A}$.

The pair $(\Omega, \mathscr{A})$ is called a *measurable space* To get to our probability space, we need to define a *measure* on the *measurable space*.

Definition 2. Given $(\Omega, \mathscr{A})$ a measurable space, a *measure* $\mathcal{P}$ is a countable additive, non-negative set function on this space. That is: $\mathcal{P} : \mathscr{A} \to \mathbb{R}_0^+$ satisfying:

- $\mathcal{P}(A) \geq \mathcal{P}(\emptyset) = 0$ for all $A \in \mathscr{A}$,
- $P(\cup_n A_n) = \sum_n P(A_n)$ for any countable collection of disjoint sets $A_n \in \mathscr{A}$.

If $\mathcal{P}(\Omega) = 1$, $\mathcal{P}$ is a *probability measure* or simply a *probability*. With the concepts that have just been explained, we get to the following definition:

Definition 3. A *measure space* is the tuple $(\Omega, \mathscr{A}, \mathcal{P})$ where $\mathcal{P}$ is a *measure* on $(\Omega, \mathscr{A})$. If $\mathcal{P}$ is a *probability measure* $(\Omega, \mathscr{A}, \mathcal{P})$ will be called a *probability space*.

Throughout this work, we will be always in the case where $\mathcal{P}$ is a probability measure, so we will always be talking about probability spaces. Some notation for these measures must be introduced. Let A and B be two events. The notation $P(A, B)$ reffers to the probability of the intersection of the events A and B, that is: $P(A, B) := P(A \cap B)$. It is clear that since $A \cap B = B \cap A$, then $P(A, B) = P(B, A)$. We remark the next definition since it will be important.

Definition 4. Let A, B be two events in Ω. The *conditional probability* of B given A is defined as:

$$P(B|A) = \frac{P(A,B)}{P(A)}.$$

There is an alternative way to state the definition that we have just made.

Theorem 1 (Bayes' Theorem)**.** *Let A, B be two events in Ω, given that $P(B) \neq 0$. Then*

$$P(B|A) = \frac{P(A|B)P(A)}{P(B)}.$$

Proof. Straight from the definition of the conditional probability we obtain that:

$$P(A,B) = P(A|B)P(B).$$

We also see from the definition that

$$P(B,A) = P(B|A)P(A).$$

Hence, since $P(A,B) = P(B,A)$,

$$P(A|B)P(B) = P(B|A)P(A) \implies P(A|B) = \frac{P(B|A)P(A)}{P(B)}.$$

$\square$

However, events might not give any information about another event occurring. When this happens, we call those events to be *independent*. Mathematically, if A,B are independent events:

$$P(A,B) = P(A)P(B)$$

and as a consequence of this, the conditional probabilty of those events is $P(A|B) = P(A)$. For a finite set of events $\{A_i\}_{i=1}^n$, we say that they are mutually independent if, and only if, every event is independet of any intersection of the other events. That is, if $\{B_i\} \subset \{A_i\}$, then

$$P\left(\cap_{i=1}^k B_i\right) = \prod_{i=1}^k P(B_i) \qquad \text{for all } k \leq n.$$

Random variables (RV) can now be introduced. Their first property is that they are measurable functions. This kind of functions are defined as it follows:

Definition 5. Let $(\Omega_1, \mathscr{A}), (\Omega_2, \mathcal{B})$ be measurable spaces. A function $f : \Omega_1 \to \Omega_2$ is said to be *measurable* if, $f^{-1}(B) \in \mathscr{A}$ for every $B \in \mathcal{B}$.

As a quick note, we can affirm that if f, g are real-valued measurable functions, and $k \in \mathbb{R}$, it is true that kf, $f+g$, fg and f/g (if g is not the identically zero function) are also *measurable functions*.

We are now ready to define one of the concepts that will lead us to the main objective of this thesis.

Definition 6. Let $(\Omega, \mathscr{A}, \mathcal{P})$ be a probability space, and $(E, \mathcal{B})$ be a measurable space. A *random variable* is a measurable function $X : \Omega \to E$, from the probability space to the measurable space. This means: for every subset $B \in (E, \mathcal{B})$, its preimage

$$X^{-1}(B) = \{\omega : X(\omega) \in B\} \in \mathscr{A}.$$

Using that sums, products and quotients of measurable functions are measurable functions, we obtain that *sums, products and quotients of random variables are random variables*.

Let now X be a R.V. The *probability* of X taking a concrete value on a measurable set contained in E, say, $S \in E$, is written as:

$$P_X(S) = P(X \in S) = P(\{a \in \Omega : X(a) \in S\}).$$

A very simple example of random variable is the following:

Example 1. Consider tossing a coin. The possible outcomes of this experiment are *Heads or Tails*. Those are our random events. We can give our random events a possible value. For instance, let *Heads* be 1 and *Tails* be 0. Then, our random variable looks like this:

$$X = \begin{cases} 1, & \text{if we obtain heads,} \\ 0, & \text{if we obtain tails.} \end{cases}$$

In the last example, our random variable is *discrete*, since the set $\{X(\omega) : \omega \in \Omega\}$ is finite. A random variable can also be *continuous*, if it can take any value within an interval.

1.2 EXPECTATION OF A RANDOM VARIABLE

Definition 7. The *cumulative distribution function* F_X of a real-valued random variable X is its probability of taking value below or equal to x. That is:

$$F_X(x) = P(X \leq x) = P(\{\omega : X(\omega) \leq x\}) = P_X((-\infty, x]) \quad \text{for all } x \in \mathbb{R}.$$

We can difference between certain types of random variables. If the image, $\mathcal{X}$, of X is countable, we call it a *discrete* random variable. Its *probability mass function* p gives the probability of the R.V. being equal to a certain value:

$$p(x) = P(X = x).$$

If the cumulative distribution function of our random variable X is continuous everywhere, then X is a *continuous* random variable. In this case there might exist a non-negative Lebesgue-integrable function f such that:

$$F_X(x) = \int_{\infty}^{x} f(t)dt,$$

called the *probability density function* of X.

During this document, distributions $P(x)$ will be adressed many times. We will use $P(x)$ to denote the probability mass function of a discrete random variable or the probability density function of a continuous random variable.

We are now ready to introduce the *expectation* of a random variable. Imagine observing a wide number of outcomes from our random variable, and taking the average of these random values. The expectation is the value of this average when we take *infinite* outcomes of our random variable.

Definition 8. Let X be a non negative random variable on a probability space $(\Omega, \mathscr{A}, \mathcal{P})$. The *expectation* $E[X]$ of X is defined as:

$$E[X] = \int_{\Omega} X(\omega)\, dP(\omega).$$

Sometimes we might be referring to multiple random variables. In these cases, in order to make reference to the variable (or distribution function, that will be presented later) for which we calculate the expectation, we will denote it as E_X (or E_P, in the case that we are adressing a distribution).

The expectation of a random variable will be also denoted as μ. Now, if X is generic $R.V$, the expectation is defined as:

$$E[X] = E[X^+] - E[X^-],$$

where X^+, X^- are defined as it follows:

$$X^+(\omega) = \max(X(\omega), 0), \qquad X^-(\omega) = \min(X(\omega), 0).$$

The expectation $E[X]$ of a random variable is a linear operation. That is, if Y is another random variable, and $\alpha, \beta \in \mathbb{R}$, then

$$E[\alpha X + \beta \mathcal{Y}] = \alpha E[X] + \beta E[\mathcal{Y}].$$

This is a trivial consequence of the linearity of the *Lebesgue integral*.

As a note, if X is a *discrete* random variable and $\mathcal{X}$ is its image, its expectation can be computed as:

$$E[X] = \sum_{x \in \mathcal{X}} x P_X(x),$$

where x is each possible outcome of the experiment, and $P_X(x)$ the probability under the distribution of X of the outcome x. The expression given in Def. 8 generalizes this particular case.

Using the definition of the expectation of a random variable, we can approach to the concept of the *moments* of a random variable.

Definition 9. If $k \in \mathbb{N}$, then $E[X^k]$ is called the $k-th$ moment of X.

If we take $k = 1$, we have the definition of the *expectation*. It is sometimes written as $m_X = E[X]$, and called the *mean*. We use the *mean* in the definition of the variance:

Definition 10. Let X be a random variable. If $E\left[X^2\right] < \infty$, then the *variance* of X is defined to be

$$\mathrm{Var}(X) = E\left[(X - m_X)^2\right] = E\left[X^2\right] - m_X^2.$$

Thanks to the linearity of the *expectation* of a random variable, it is easy to see that, if $a, b \in \mathbb{R}$, then

$$\mathrm{Var}(aX + b) = E[(aX + b) - E[aX + b])^2] = a^2 E[(X - m_X)^2] = a^2\,\mathrm{Var}(X).$$

Usually, when it comes to applying these concepts to a real problem, we will be observing multiple features that a phenomenon in nature presents. We would like to have a collection of random variables each one representing one of this features. In order to set the notation for these kinds of situations, we will introduce *random vectors*.

Definition 11. A random vector is a row vector $\mathbf{X} = (X_1, \ldots, X_n)$ whose components are real-valued random variables on the same probability space $(\Omega, \mathscr{A}, P)$.

The probability distribution of a random variable can be extended in to the *joint probability distribution* of a random vector.

Definition 12. Let $\mathbf{X} = (X_1, \ldots, X_n)$ be a random vector. The *cumulative distribution function* (or simply, the *distribution function*) $F_{\mathbf{X}} : \mathbb{R}^n \to [0, 1]$ of $\mathbf{X}$ is defined as:

$$F_{\mathbf{X}}(x) = P(X_1 \leq x_1, \ldots, X_n \leq x_n).$$

We also name it *multivariate distribution*. Before, we presented the concept of independence between a pair of events. Using the cumulative distribution function, we can now define the independence between random variables.

Definition 13. A finite set of n random variables $\{X_1, \ldots, X_n\}$ is mutually independent if, and only if, for any sequence $\{x_1, \ldots, x_n\}$, the events $\{X_1 \leq x_1\}, \ldots, \{X_n \leq x_n\}$ are mutually independent. Equivalently, this finite set is mutually independent if, and only if,:

$$F_{X_1,\ldots,X_n}(x_1, \ldots, x_n) = F_{X_1}(x_1) \ldots F_{X_n}(x_n), \qquad \text{for all } x_1, \ldots, x_n.$$

We can also extend the notion of expectation to a random vector. Let $\mathbf{X} = (X_1, \ldots, X_n)$ be a random vector and assume that $E[X_i]$ exists for all $i \in \{1, \ldots, n\}$. The expectation of $\mathbf{X}$ is defined as the vector containing the expectations of each individual random vector, that is:

$$E[\mathbf{X}] = \begin{bmatrix} E[X_1] \\ \vdots \\ E[X_n] \end{bmatrix}.$$

To generalize the variance of a random variable, we have to build the following matrix.

Definition 14. Let $\mathbf{X} = (X_1, \ldots, X_n)$ be a random vector. Then, the *covariance matrix* of $\mathbf{X}$ is defined as:

$$\Sigma = \mathrm{Cov}(\mathbf{X}) = E[(\mathbf{X} - \mu_{\mathbf{X}})(\mathbf{X} - \mu_{\mathbf{X}})^T] = \begin{pmatrix} \sigma_{11} & \cdots & \sigma_{1n} \\ \vdots & \ddots & \vdots \\ \sigma_{n1} & \cdots & \sigma_{nn} \end{pmatrix},$$

where $\sigma_{ij} = \mathrm{Cov}(X_i, X_j) = E[(X_i - \mu_i)(X_j - \mu_j)] = \sigma_{ji}$.

It can also happen that, given a random vector, we would like to know the probability distribution of some of its components. That is called the *marginal distribution*.

Definition 15. Let $\mathbf{X} = (X_1, \ldots, X_n)$ be a random vector. The *marginal distribution* of a subset of $\mathbf{X}$ is the probability distribution of the variables contained in the subset.

In the simple case of having two random variables, e.g. $X = (X_1, X_2)$, then the marginal distribution of X_1 is:

$$P(x) = \int_{x_2} P(x_1, x_2) dx_2.$$

2 DISTRIBUTIONS

We have introduced the concepts of *random variable*, *random vector* and its *probability distribution*. Now, given two distributions, in the following chapters we will like to see how different they are from each other. In order to compare them, we enunciate the definition of the Kullback-Leibler divergence.

Definition 16. Let P and Q be probability distributions over the same probability space Ω. Then, the Kullback-Leibler divergence is defined as:

$$D_{KL}(P \mid\mid Q) = E_P \left[\log \frac{P(x)}{Q(x)} \right].$$

It is defined if, and only if, P is *absolutely continuous with respect to* Q, that is, if $P(A) = 0$ for any A subset of Ω where $Q(A) = 0$. There are some properties of this definition that must be stated.

Proposition 1. *If P, Q are two probability distributions over the same probability space, then $D_{KL}(P|Q) \geq 0$.*

Proof. Firstly, note that if $a \in \mathbb{R}^+$, then $\log\ a \leq a - 1$. Then:

$$\begin{aligned}
-D_{KL}(P \mid\mid Q) &= -E_P \left[\log \frac{P(x)}{Q(x)} \right] \\
&= E_P \left[\log \frac{Q(x)}{P(x)} \right] \\
&\leq E_P \left[\left(\frac{Q(x)}{P(x)} - 1 \right) \right] \\
&= \int P(x) \frac{Q(x)}{P(x)}\, dx - 1 \\
&= 0.
\end{aligned}$$

So we have obtained that $-D_{KL}(P \mid\mid Q) \leq 0$, which implies that $D_{KL}(P \mid\mid Q) \geq 0$. □

As a corollary of this proposition, we can affirm that $D_{KL}(P \mid\mid Q)$ equals zero if and only if $P = Q$ almost everywhere. We will also remark the discrete case, as it will be used later. Let P, Q be discrete probability distributions defined on the same probability space Ω. Then,

$$D_{KL}(P \mid\mid Q) = \sum_{x \in \Omega} P(x) \log \left(\frac{P(x)}{Q(x)} \right).$$

2.1 EXAMPLES OF DISTRIBUTIONS

Let us present some examples o common distributions. They will be used further in this document.

Bernoulli

Think for a moment that you want to model the possible outcomes of an experiment with two possibilites: sucess or failure. Imagine also that you already know that in your experiment there is a probability p of achieving success. That is the intuitive idea of a Bernoulli distribution. We can define it more formally as follows:

The *Bernoulli distribution* is a discrete probability distribution of a random variable that takes two values, $\{0,1\}$, with probabilities p and $q = 1 - p$, respectively. We will say that our distribution is a $Bern(p)$.

If k is a possible outcome, we can define the probability mass function f of a Bernoulli distribution as:

$$f(k,p) = \begin{cases} p, & \text{if } k = 1, \\ 1 - p, & \text{if } k = 0. \end{cases}$$

Using the expression of the mean for discrete random variables, we obtain that $E[X] = p$ and

$$\text{Var}[X] = E[X^2] - E[X]^2 = E[X] - E[X]^2 = p - p^2 = p(1-p) = pq.$$

As a note, this is just a particular case of the *Binomial distribution* with $n = 1$.

Gaussian Distribution

The Gaussian (or normal) distribution is used to represent real-valued random variables whose distributions are not known. Its importance relies in the fact that, using the *central limit theorem*, we can assume that the average of many samples of a random variable with finite mean and variance is a random variable whose distribution converges to a normal distribution as the number of samples increases.

Definition 17. We say that the real valued random variable X follows a *normal distribution* of parameters $\mu, \sigma \in \mathbb{R}$ if, and only if, its probability density function exists and it is determined by

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2},$$

where μ is the mean and σ is its standard deviation. We denote this normal distribution as $X \sim \mathcal{N}(\mu, \sigma)$.

The particular case where $\mu = 0$ and $\sigma = 1$ is widely used in statistics. In this case, the density function is simpler:

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}.$$

A remarkable property of these distributions is that, if $f : \mathbb{R} \to \mathbb{R}$ is a real-valued function defined as $f(x) = ax + b$, then $f(X) \sim \mathcal{N}(a\mu + b, |a|\sigma)$.

In the same way that we extended random variables to random vectors, we can extend the normal distribution to a multivariate random distribution.

Definition 18. We say that a random vector $\boldsymbol{X} = (X_1, \ldots, X_n)$ follows a multivariate normal distributions of parameters $\mu \in \mathbb{R}^n$, $\Sigma \in \mathcal{M}_N(\mathbb{R})$ if, and only if, its probabity density function is:

$$f(x) = \frac{1}{\sqrt{\det(2\pi\Sigma)}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}.$$

It is denoted $X \sim \mathcal{N}(\mu, \Sigma)$. In this case, μ is the mean vector of the distribution and Σ denotes the covariance matrix.

3 STATISTICAL INFERENCE

Statistical inference is the process of deducing properties of an underlying distribution by analyzing the data that it is available. With this purpose, techniques like deriving estimates and testing hypotheses are used.

Inferential statistics are usually contrasted with descriptive statistics, which are only concerned with properties of the observed data. The difference between these two is that in inferential statistics, we assume that the data comes from a larger population that we would like to know.

In *machine learning*, subject that concerns us the most, the term inference is sometimes used to mean *make a prediction by evaluating an already trained model*, and in this context, inferring properties of the model is refered as *training or learning*.

3.1 PARAMETRIC MODELING

In the following chapters, we will be trying to estimate density functions in a dataset. To do this we will be using *parametric models*. We say that a *parametric model*, $p_\theta(x)$, is a family of density functions that can be described using a finite numbers of parameters θ. We can get to the concept of *log-likelihood* now.

Definition 19. The *likelihood* $\mathcal{L}(\theta|x)$ of a parameter set θ is a function that measures how plausible is θ, given an observed point x in the dataset $\mathcal{D}$. It is defined as the value of the density function parametrized by θ at x. That is:

$$\mathcal{L}(\theta|x) = p_\theta(x).$$

In a finite dataset $\mathcal{D}$ consisting of independent observations, we can write:

$$\mathcal{L}(\theta|X) = \prod_{x \in D} p_\theta(x).$$

This can be computationally hard to work with, so the log-likelihood is often used instead.

Definition 20. Let $\mathcal{D}$ be a dataset of independent observations and θ a set of parameters. Then, we define the *log-likelihood* ℓ as the sum of the logarithms of the evaluations of p_θ in each x in the dataset. That is:

$$\ell(\theta|X) = \sum_{x \in \mathcal{D}} \log p_\theta(x).$$

Our goal would be to find the optimal value $\hat{\theta}$ that maximizes the likelihood of observing the dataset $\mathcal{D}$. We get to the following definition:

Definition 21. We say that $\hat{\theta} = \hat{\theta}(\mathcal{D})$ is a *maximum likelihood estimator*(MLE) for θ if

$$\hat{\theta} \in \arg\max_{\theta} \mathcal{L}(\theta|\mathcal{D})$$

for every observation $\mathcal{D}$.

3.2 MINIMAL SUFFICIENT STATISTICS

In parametric modeling, the goal was to determine the density function under a distribution. Another interesting task can be determining specific parameters or quantitys related to a distribution, given a sample $X = (x_1, \cdots, x_n)$.

Definition 22. A *statistic* is a measurable function of the data. That is, if $T : \Omega \to \mathbb{T}$ is measurable, then $T(X)$ is a statistic.

Remark 1. A statistic is also a random variable.

However, not all statistics will provide useful information for the statistical inference problem. We would like to find statistics that provide relevant information.

Definition 23. Let $X \sim P_\theta$. Then, the statistic $T(X) = T : (\Omega, \mathscr{A}) \to (\mathbb{T}, \mathcal{B})$, is sufficient for a family of parameters $\{P_\theta \ : \ \theta \in \Theta\}$ if the conditional distribution of X, given $T = t$, is indepentent of θ.

Example 2. The simplest example of a sufficient statistic is the mean μ of a gaussian distribution with known variance. Oppositely, the *median* of an arbitrary distribution is not sufficient for the mean since, even if the median of the sample is known, more information about the mean of the population can be obtained from the mean of the sample itself.

Although it will not be shown in this document, sufficient statistics are not unique. In fact, if T is sufficient, $\psi(T)$ is sufficient for any bijective mapping ψ. It would be interesting to find a sufficient statistic T that is *the smallest* of them.

Definition 24. A sufficient statistic T is minimal if, for every sufficient statistic U, there exists a mapping f such that $T(x) = f(U(x))$ for any $x \in \Omega$.

Part II

INFORMATION THEORY

4 MUTUAL INFORMATION

Obtaining good representations of data is one of the most important tasks in machine learning (ML). Recently, it has been discovered that maximizing *mutual information* between two elements in our data can give us good representations for our data. In this section, *information theory* notions will be presented, in order to use them in our ML models. This will provide a theoretical solid base for the notions explained later.

4.1 ENTROPY

The *mutual information* concept is based on the *Shannon entropy*, which we will introduce first, along with some basic properties of it. The Shannon entropy is a way of measuring the uncertainty in a random variable. Given an event $\mathcal{A} \in \mathscr{A}$, P a probability measure and $P[\mathcal{A}]$ the probability of $\mathcal{A}$, we can affirm that

$$\log \frac{1}{P[\mathcal{A}]}$$

describes *"how surprising is that $\mathcal{A}$ occurs"*. For instance, if $P[\mathcal{A}] = 1$, then the last expression is zero, which means that it is not a surprise that $\mathcal{A}$ occurred. With this motivation, we get to the following definition.

Definition 25. Let X be a discrete random variable with image $\mathcal{X}$. The *Shannon entropy*, or simply *entropy* $H(X)$ of X is defined as:

$$H(X) = E_X \left[\log \frac{1}{P_X(X)} \right] = \sum_{x \in \mathcal{X}} P_X(x) \log \frac{1}{P_X(x)}.$$

The *entropy* can trivially be expressed as:

$$H(X) = - \sum_{x \in \mathcal{X}} P_X(x) \log P_X(x).$$

The simple example, found in (link), even though it is very simple, it is very illustrative for our definition:

Example 3. Let $X \sim Bern(p)$. Then, the entropy of X is:

$$H(X) = -p \log p - (1-p) \log(1-p)) = H(p),$$

since H only depends on p. In Fig.?? we can see a representation of this function. We appreciate that in this case, H is concave and equals 0 if $p \in \{0, 1\}$, which are the values of p that give us no uncertainty. The maximum uncertainty is obtained when $p = \frac{1}{2}$, where we do not know what to expect as an outcome from our random variable X.

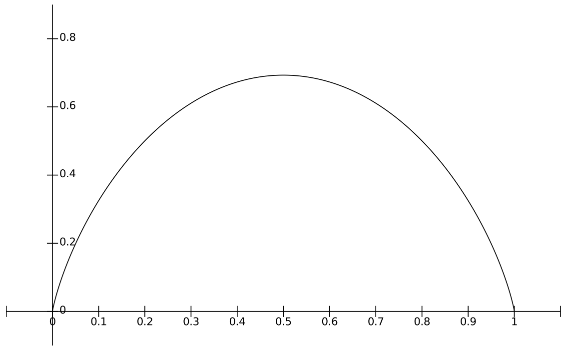

Figure 1: Representation of $H(p)$ in the example 3.

There are some properties of the *entropy* that must be remarked.

Proposition 2. *Let X be a random variable with image $\mathcal{X}$. If $|\mathcal{X}|$ is the cardinal of $\mathcal{X}$, then*

$$0 \leq H(X) \leq \log(|\mathcal{X}|).$$

Proof. Since $\log y$ is concave on $\mathbb{R}^+$, by Jensen's inequality (see Appendix A, Prop. 4), we obtain:

$$H(X) = -\sum_{x \in \mathcal{X}} P_X(x) \log P_X(x) \leq \log\left(\sum_{x \in \mathcal{X}} 1\right) = \log(|\mathcal{X}|).$$

For the lower bound we see that, since $P_X(x) \in [0,1]$ for all $x \in \mathcal{X}$ then $\log P_X(x) \leq 0 \quad \forall x \in \mathcal{X}$. Hence , $-P_X(x) \log P_X(x) \geq 0$ for all $x \in X$, so $H(X) \geq 0$. ☐

We can also see that the equality on the left holds if , and only if , exists x in X such that its probability is exactly one, that is $P_X(x) = 1$. The right equality holds if and only if , for all $x \in \mathcal{X}$, its probability is $P_X(x) = \frac{1}{|X|}$.

Conditional entropy

We have already said that entropy measures how surprising is that an event occurs. Usually, we will be looking at two random variables and it would be interesting to see how likely is that one of them, say $X(x)$, occurred, if we already know that $Y(y)$ occurred. This leads us to the definition of *conditional entropy*. Let us see a simpler case first:

Let A be an event, and X a random variable. The conditional probability $P_{X|A}$ defines the entropy of X conditioned to A:

$$H(X|A) = \sum_{x \in \mathcal{X}} P_{X|A}(x) \log \frac{1}{P_{X|A}(x)}.$$

If Y is another random variable and $\mathcal{Y}$ is its image, intuitively we can sum the conditional entropy of an event with all the events in $\mathcal{Y}$, and this way we obtain the conditional entropy of X given Y.

Definition 26 (Conditional Entropy)**.** Let X, Y be random variables with images $\mathcal{X}, \mathcal{Y}$. The *conditional entropy* $H(X|Y)$ is defined as:

$$\begin{aligned} H(X|Y) &:= \sum_{y\in\mathcal{Y}} P_{\mathcal{Y}}(y) H(X|Y=y) \\ &= \sum_{y\in\mathcal{Y}} P_{\mathcal{Y}}(y) \sum_{x\in\mathcal{X}} P_{X|Y}(x|y) \log \frac{1}{P_{X|Y}(x|y)} \\ &= \sum_{x\in X, y\in\mathcal{Y}} P_{XY}(x,y) \log \frac{P_Y(y)}{P_{XY}(x,y)}. \end{aligned}$$

The interpretation of the conditional entropy is simple: the uncertainty in X when Y is given. Since we know about an event that has occurred (Y), intuitively the conditional entropy , or the uncertainty of X occurring given that Y has occurred, will be lesser than the entropy of X, since we already have some information about what is happening. We can prove this:

Proposition 3. *Let X, Y be random variables with images $\mathcal{X}, \mathcal{Y}$. Then:*

$$0 \le H(X|Y) \le H(X).$$

Proof. The inequality on the left was proved on Proposition 2. The characterization of when $H(X|Y) = 0$ was also mentioned after it. Let us look at the inequality on the right. Note that restricting to the (x, y) where $P_{XY}(x, y) > 0$ and using the definition of the conditional probability we have:

$$\begin{aligned} H(X|Y) &= \sum_{y\in\mathcal{Y}} P_Y(y) \sum_{x\in\mathcal{X}} P_{X|Y}(x|y) \log \frac{1}{P_{X|Y}(x|y)} \\ &= \sum_{x\in\mathcal{X}, y\in\mathcal{Y}} P_Y(y) P_{X|Y}(x,y) \log \frac{P_Y(y)}{P_{XY}(x,y)} \\ &= \sum_{x\in\mathcal{X}, y\in\mathcal{Y}} P_{XY}(x,y) \log \frac{P_Y(y)}{P_{XY}(x,y)}, \end{aligned}$$

and

$$H(X) = \sum_x P_X(x) \log \frac{1}{P_X(x)} = \sum_{x,y} P_{XY}(x,y) \log \frac{1}{P_X(x)}.$$

Hence,

$$\begin{aligned} H(X|Y) - H(X) &= \sum_{x,y} P_{XY}(x,y) \left(\log \frac{P_Y(y)}{P_{XY}(x,y)} - \log \frac{1}{P_X(x)} \right) \\ &= \sum_{x,y} P_{XY} \log \frac{P_Y(y) P_X(x)}{P_{XY}(x,y)}. \end{aligned} \tag{1}$$

So, using Jensen's inequality, we obtain:

$$\begin{aligned} \sum_{x,y} P_{XY} \log \frac{P_Y(y)P_X(x)}{P_{XY}(x,y)} &\le \log \left(\sum_{x,y} \frac{\cancel{P_{XY}(x,y)}\ P_Y(y)P_X(x)}{\cancel{P_{XY}(x,y)}} \right) \\ &= \log \left(\left(\sum_x P_X(x) \right) \left(\sum_y P_Y(y) \right) \right) = \log 1 = 0, \end{aligned}$$

and this leads us to:

$$H(X|Y) - H(X) \leq 0 \quad \text{then} \quad H(X|Y) \leq H(X) \tag{2}$$

as we wanted. $\square$

It must be noted that the inequality the state of the proposition,

$$0 \leq H(X|Y) \leq H(X),$$

in the inequality of the left, equality holds if, and only if, $P_{XY}(x,y) = P_X(x)P_Y(y)$ for all (x,y) with $P_{XY}(x,y) > 0$, as it is said in Jensen's inequality. For the inequality on the right, equality holds if and only if $P_{XY}(x,y) = 0$, which implies $P_X(x)P_Y(y) = 0$ for any $x \in \mathcal{X}$, $y \in \mathcal{Y}$. It follows that $H(X|Y) = H(X)$ if and only if $P_{XY}(x,y) = P_X(x)P_Y(y)$ for all $(x,y) \in \mathcal{X} \times \mathcal{Y}$

4.2 MUTUAL INFORMATION

Using the entropy of a random variable we can directly state the definition of *mutual information* as follows:

Definition 27. Let X, Z be random variables. The *mutual information (MI)* between X and Z is expressed as the difference between the entropy of X and the conditional entropy of X and Z, that is:

$$I(X,Z) := H(X) - H(X|Z).$$

Since the entropy of the random variable $H(X)$ explains the uncertainty of X occurring, the intuitive idea of the *MI* is to determine the decrease of uncertainty of X occurring when we already know that Z has occurred. We also have to note that, using the definition of the *entropy* and the expression obtained in Eq. 1, we can rewrite the *MI* it follows:

$$\begin{aligned} I(X,Z) &= \sum_{x \in \mathcal{X}} P_X(x) \log \frac{1}{P(x)} - \sum_{x \in \mathcal{X}, z \in \mathcal{Z}} P_{XZ}(x,z) \log \frac{P_Z(x)}{P_{XZ}(x,z)} \\ &= \sum_{x,z} P_{XZ} \log \frac{P_Z(z)P_X(x)}{P_{XZ}(x,z)} = D_{KL}(P_{XZ} \mid\mid P_X P_Z) \end{aligned}$$

and we have obtained an expression of the mutual information using the *Kullback-Leibler* divergence. This provides with the following immediate consequences:

- (i) Mutual information is non-negative. That is : $I(X,Z) \geq 0$.
- (ii) If X, Z are random variables, then its mutual information equals zero if, and only if, they are independent. This easy to check since if $D_{KL}(P_{XZ} \mid\mid P_X P_Z) = 0$, then $P_{XZ} = P_X P_Z$ almost everywhere so X and Z are independent.p
- (iii) Since $P_{XZ} = P_{ZX}$ and $P_X P_Z = P_Z P_X$, mutual information is symmetric. That is: $I(X,Z) = I(Z,X)$.

Remark 2. We can set a connection between the mutual information and sufficient statistics. Let $T(X)$ be a statistic. We say that $T(X)$ is sufficient for θ if its mutual information with θ equals the mutual information between X and θ, that is:

$$I(\theta, X) = I(\theta, T(X)).$$

This means that sufficient statistics preserve mutual information and conversely.

Part III

REPRESENTATION LEARNING

5 CONTEXT

Before continuing to present the mathematical notions that are neccesary for the topics that are treated in this work, it is interesting to present what we are trying to achieve.

Machine learning is the field of computer science that studies algorithms that improve automatically through experience from examples. These algorithms allow computers to discover how to perform tasks without being explicitly programmed to do them. For the computers to learn, it is mandatory that a finite set of data (or dataset) $\mathcal{D}$ is available.

Whenever a computer is providen with data, the data can be *labeled* or *unlabeled*. Labeled data is the one that, each point $x_i \in \mathcal{D}$ is related to a tag $y_i \in Y$, where Y is a set of classes. Unlabeled data is the kind of data that does not have a label or class associated to it, so it is just $x_i \in \mathbb{R}^d$.

Depending on how the data (*or signal*) is given to the computer, the machine learning approaches can be divided into three broad categories:

1. *Supervised learning*. In this category the goal is to use the labeled data in order to find a function that maps the dataset to the set of classes. That is a function $g : \mathcal{D} \to Y$. An example of supervised learning is image classification: giving a label to an image.
2. *Unsupervised learning*. In ths case, the data is unlabeled, so the approach is completely different. Usually, the goal here is to discover hidden patterns in data or to learn features from it. An example of this kind of learning is K means, which consistis in clustering the data in k groups. It is also known as *self-supervised* learning.
3. *Reinforcement learning*. This is the area concerned with how intelligent agents take decisions in a specific environment in order to obtain the best reward in their objective. This kind of learning is used

In this work, we will focus on unsupervised learning. Particularly, in representation learning.

There are many different tasks that can be performed with the data, such as linear regression, logistic regression, or classification. In any of these tasks, computers might need to do intermediate steps before giving a label to the input example. Sometimes, they must create a *representation* that contains the data's key qualities. Here is where *representation learning* is born.

Intuitively, if x is a datapoint in a dataset $\mathcal{D} \subset \mathbb{R}^d$, a *representation* of x is a vector $r \in \mathbb{R}^n$ (usually, $n \leq d$), that shares information with the datapoint

x. That is, if a machine learning model tries to make a posterior task, such as classification, the input x must be transformed to a, usually lower dimensional, vector r in order to perform the final label.

Features are parts or patterns of an datapoint $x \in \mathcal{D}$ that help to identify it. In fact, this attribute is shared by all independent units that represent the same object. For instance, if we consider an image of any square, we should be able to identify 4 corners and 4 edges. These could be features of a square. When we mention feature detection, we are adressing the methods for detecting these features of a datapoint.

Representation learning is a set of techniques that allow a system to discover the representations needed for feature detection or classification. In contrast to manual feature engineering (which involves manually exploring the data and finding relationships in it), feature learning allows a machine to learn the features and to use them to perform a task.

Feature learning can be supervised or unsupervised. In supervised feature learning, representations are learned using labeled data. Examples of this kind of feature learning are supervised neural networks and multilayer perceptron. In unsupervised learning, the features are learned using unlabeled data. There are many examples of this, such as independent component analysis (ICP) and autoencoders. In this work, we will be working with unsupervised feature learning.

The performance of machine learning methods is heavily dependent on the choice of data features (Bengio *et al.*, 2014). This is why most of the current effort in machine learning focuses on designing preprocessing and data transformation that lead to good quality representations. A representation will be of good quality when its features produce good results when we evaluate the *accuracy* of our model.

The main goal in representation learning is to obtain features of the data that are generally good for either of the supervised tasks. That is, we would like to obtain a representation that is either good for image classification (giving an image a label of what we can see in it) or image captioning (producing a text that describes the image).

Data's features that are invariant through time are very useful for machine learning models. In (Wiskott & Sejnowski, 2002), *slow features* are presented. Slow features are defined as features of a signal (which can be the input of a model) that vary slowly during time. That means, if $\boldsymbol{X}$ is a *time series*[1], we will try to find any number of features in $\boldsymbol{X}$ that vary the most slowly. These kind of features are the most interesting ones when creating representations, since they give an abstract view of the original data.

1 A time series is an ordered sequence of values of a random variable at, usually, equally spaced time intervals.

Let us give an example: In computer vision, the value of the pixels in an image can vary fast. For instance, if we have a zebra on a video and the zebra is moving from one side of the image to the other, due to the black stripes of this animal, the pixels will fast change from black to white and viceversa, so value of pixels is probably not a good feature to choose as an slow feature. However, there will always be a zebra on the image, so the feature that indicates that there is a zebra on the image will stay positive throughout all the video, so we can say that this is a slow feature.

We will be studying different models that try to learn representations from raw data without labels, as we have mentioned. We usually need a function that measures what is the penalty that the model gets for a choice of a parameter. This is called a *loss function*, that we will want to optimize.

For instance, in a regression problem, a good example of loss function is *mean squared error*, which is expressed as follows:

$$\mathrm{MSE} = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n}.$$

In a classification problem, each datapoint x_i has a correct classification y_i. In this case, the score of the correct category y_i should be greater than the sum of the scores of all incorrect categories y_j with $j \neq i$, so we could use a function like *support vector machine (SVM) loss*:

$$\mathrm{SVMLoss} = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

In the context of machine learning, a *model* is the result of running a maching learning algorithm in data. This model will represent what the computer has learned from the data using this algorithm. As an easy example, when we use the linear regression algorithm, we obtain a model that is a vector of coefficients with specific values.

In the following chapters, we will explain different kinds of neural networks and which kind of loss functions they use.

6 GENERATIVE MODELS

From now on, let $\mathcal{D}$ be any kind of observed data. This will always be a finite subset of samples taken from a probability distribution p_{data}. There are models that, given $\mathcal{D}$, try to approximate the probability distribution that lies underneath it. These are called *generative models (G.M.)*.

Generative models can give parametric and non parametric approximations to the distribution p_{data}. In our case, we will focus on parametric approximations where the model searches for the parameters that minimize a chosen metric (which can be a distance or other kind of metric such as K-L divergence) between the model distribution and the data distribution.

We can express our problem more formally as follows. Let θ be a generative model within a model family $\mathcal{M}$. The goal of generative models is to optimize:

$$\min_{\theta \in \mathcal{M}} d(p_{\text{data}}, p_\theta),$$

where d stands for the distance between the distributions. We can use, for instance, K-L divergence.

Generative models have many useful applications. We can however remark the tasks that we would like our generative model to be able to do. Those are:

- Estimate the density function: given a datapoint, $x \in D$, estimate the probability of that point $p_\theta(x)$.
- Generate new samples from the model distribution $x \sim p_\theta(x)$.
- Learn useful features of the datapoints.

If we have a look again at the example of the zebras, if we make our generative model learn about images of zebras, we will expect our $p_\theta(x)$ to be high for zebra's images. We will also expect the model to generate new images of this animal and to learn different features of the animal, such as their big size in comparison with cats.

6.1 AUTOREGRESSIVE MODELS

In time-series theory, autoregressive models use observations from previous time steps to predict values at the current time. Fixing an order of the variables $x_1, \dots, x_n$, the distribution for the i-th random variable depends on all the preceding values in the particular chosen order. We will make use of the name of these models to define the machine learning approach.

A very first definition of *autoregressive models (AR)* would be the following one: *autoregressive models are feed-forward models that predict future values using past values*. Let us go deeper into this concept and explain how it behaves.

Again, let $\mathcal{D}$ be a set of $n-$dimensional datapoints x. We can assume that $x \in \{0,1\}^n$ for simplicity, without losing generality. If we choose any $x \in \mathcal{D}$, using the chain rule of probability, we obtain

$$p(x) = \prod_{i=1}^{n} p(x_i|x_1, \ldots, x_{i-1}) = \prod_{i=1}^{n} p(x_i|\boldsymbol{x}_{<i}),$$

where $\boldsymbol{x}_{<i} \in \mathbb{R}^{i-1}$ is a vector whose componentes are the previous x_j for $j = 1, \ldots, i-1$, that is: $\boldsymbol{x}_{<i} = [x_1, \ldots, x_{i-1}]$.
It is known that given a set of discrete and mutually dependent random variables, they can be displayed in a table of conditional probabilities. If K_i is the number of states that each random variable can take then $\prod K_i$ is the number of cells that the table will have. If we represent $p(x_i|\boldsymbol{x}_{<i})$ for every i in tabular form, we can represent any possible distribution over n random variables.

This, however, will cause an exponential growth on the complexity of the representation, due to the need of specifying 2^{n-1} possibilities for each case. In terms of neural networks, since each column must sum 1 because we are working with probabilities, we have $2^{n-1} - 1$ parameters for this conditional, and the tabular representation becomes impractical for our network to learn when n increases.

In autoregressive generative models, the conditionals are specified as we have mentioned before: parameterized functions with a fixed numbers of parameters. More precisely, we assume the conditional distributions to be Bernoulli random variables and learn a function f_i that maps these random variables to the mean of the distribution. Mathematically, we have to find

$$p_{\theta_i}(x_i|\boldsymbol{x}_{<i}) = \text{Bern}(f_i(x_1, \ldots, x_{i-1})),$$

where θ_i is the set of parameters that specify the mean function $f_i : \{0,1\}^{i-1} \to [0,1]$.

Then, the number of parameters is reduced to $\sum_{i=1}^{n} |\theta_i|$ so we can not represent all possible distributions as we could when using the tabular form of the conditional probabilities. We are now setting the limit of its expressiveness because we are setting the conditional distributions $p_{\theta_i}(x_i|\boldsymbol{x}_{<i})$ to be *Bernoulli* random variables with the mean specified by a restricted class of parametrized functions.

Let us see a very simple case first in order to understand it better and later we will generalize it. Let σ be a *sigmoid*[1] non linear function and $\theta_i = \left\{\alpha_0^{(i)}, \alpha_1^{(i)}, \ldots, \alpha_{i-1}^{(i)}\right\}$ the parameters of the mean function. Then, we can define our function f_i as :

$$f_i(x_1, \cdot, x_{i-1}) = \sigma(\alpha_0^{(i)} + \alpha_1^{(i)} x_1 + \cdots + \alpha_{i-1}^{(i)} x_{i-1}).$$

In this case, the number of parameters would be $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$, so using *Big O* notation, we would be in the case of $O(n^2)$. We will state now a more general and useful case, giving a more interesting parametrization for the mean function: *multi layer perceptrons*[2](MLP).

For this example we will consider the most simple MLP: the one with one hidden layer. Let $h_i = \sigma(\boldsymbol{A}_i \boldsymbol{x}_{<i} + c_i)$ be the hidden layer activation function. Remember that $h_i \in \mathbb{R}^d$. Let $\theta_i = \{\boldsymbol{A}_i \in \mathbb{R}^{d\times(i-1)},\ c_i \in \mathbb{R}^d,\ \alpha^{(i)} \in \mathbb{R}^d,\ b_i \in \mathbb{R}\}$ the set of parameters for the mean function f_i, that we define as:

$$f_i(\boldsymbol{x}_{<i}) = \sigma(\alpha^{(i)} h_i + b_i).$$

In this case, the number of parameters will be $O(n^2 d)$.

1 A sigmoid function is a bounded,differentiable, real function which derivative is non-negative at each point and it has exactly one inflection point.

2 Multi layer perceptrons are feed-forward neural networks with at least 3 layers: input, hidden and output layers; each one using an activation function.

7 THE INFONCE LOSS

We are now ready to connnect the concepts of mutual information and generative models that we have presented. In unsupervised learning, it is a common strategy to predict future information and to try to find out if our predictions are correct. In *natural language processing*, for instance, representations are learned using neighbouring words (Mikolov *et al.*, 2013), and in images, some studies have been able to predict color from grey-scale (Doersch *et al.*, 2016).

When we talk about high-dimensional data, it is not useful to make use of an unimodal loss function to evaluate our model. If we did it like this, we would be assuming that there is only one peak in the distribution function and that it is actually similar to a Gaussian. This is not always true, so we can not assume it for our models. Generative models can be used for this purpose: they will model the relationships in the data x. However,they ignore the context c in which the data x is involved. As an easy example of this, an image contains thousands of bits of information, while the label that classifies the image contains much less information , say, 10 bits for 1024 categories. Because of this, modeling $p(x|c)$ might not be the best way to proceed if we want to obtain the real distribution that generates our data.

During the last few years, the representation learning problem has been approached using different machine learning frameworks. The most competitive ones have been self-supervised contrastive representation learning Oord *et al.* (2019); Tian *et al.* (2020); Hjelm *et al.* (2019); Gutmann & Hyvarinen (n.d.); Chen *et al.* (2020); He *et al.* (2020) using *contrastive losses*, and they have empirically outperformed other approaches.

In contrastive learning, different "views" of the same input are created. These are also called *positive samples*. Then, they are compared with *negative samples*, which are views created from an input that does not share information with the input of the positive sample. The idea is to try and maximize the mutual information between positive samples and push appart the views taken from negative samples.

There are many ways of creating samples, both positive and negative, of an input. For instance:

- Randomly cropping different parts of an image. These would be examples of positive examples.
- Rotating or flipping images or crops of them would also be examples of positive samples.
- Taking different time-steps of a video would create positive samples.
- Selecting different parts of the same text would also be a positive example.

- Negative samples are created by applying one of the previous techniques to images that have nothing in common with the positive input.

In fact, if v_1, v_2 are two views of an input, we can think of the positive pairs as points coming from a joint distribution over the views $p(v_1, v_2)$, and negative samples coming from the product of the marginals $p(v_1)p(v_2)$, (Tian *et al.*, 2020).

It is important to find a way to determine how much shared information between the views is needed, in order to make the representations obtained good enough for any downstream task. Here is where the *InfoMin principle* is born. A good set of views are those that share the minimal information neccesary to perform well at the downstream task. We will follow the argument presented in Oord *et al.* (2019) to find a lower bound on mutual information.

Our goal here will be to seek for a way of extracting shared information between the context and the data. Due to the differences in data dimensionality, if we want to predict the future x using the context c, firstly we must encode our entry data x into a representation which size is comparable to context size. Firstly, an *encoder* is used. An encoder is a model that, given an input x, provides a feature map or vector that holds the information that the input x had. In fact, and here is where we link the mutual information with the current topic, we want our encoder to maximize

$$I(x,c) = \sum_{x,c} p(x,c) \log \frac{p(x|c)}{p(x)} \tag{3}$$

that is, the mutual information between the input x and the context c. Maximizing the mutual information between x and c, we extract the latent variables that the inputs have in common.

So, we will use an encoder g_{enc} that transforms the input sequence of observations x_t to a sequence of latent representations

$$z_t = g_{enc}(x_t).$$

After we have obtained z_t, we use it as input of an autoregressive model to produce a context latent representation:

$$c_t = g_{ar}(z_{\leq t}).$$

In this case, c_t will summarize the information of z_i for $i \leq t$. Following the argument that we gave before, predicting the future x_{t+k} using only a generative model (say $p_k(x_{t+k}|c)$) might not be correct, since we would be ignoring the context.

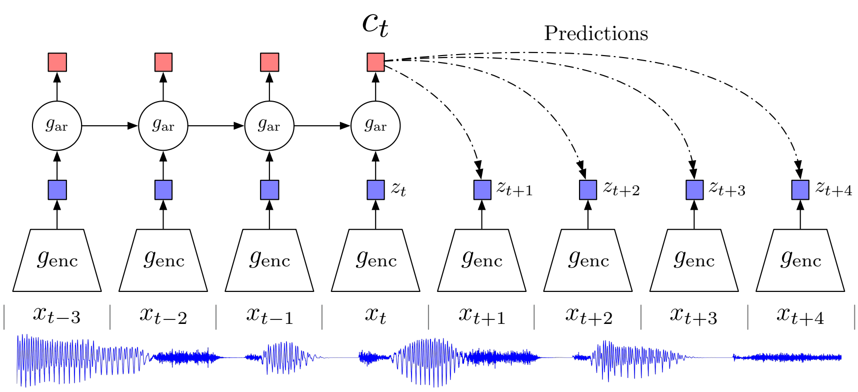

Figure 2: Image from (Oord *et al.*, 2019). Overview of Contrastive Predictive Coding framework using audio signal as input.

Let us see how we train the encoder g_{enc} and the autoregressive model g_{ar}.

Let $X = \{x_1, \dots, x_N\}$ be a set of N random samples. X will contain positive sample taken from the distribution $p(x_{t+k}|c_t)$ and $N-1$ negative samples from the distribution propused $p(x_{t+k})$. With this set, we would like to optimize the following loss function:

$$\mathcal{L}_N = -\underset{X}{E}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_k)}\right]. \tag{4}$$

Let us have a look at the *categorical cross-entropy* loss function:

$$\mathcal{L}(y, s) = -\sum_{i}^{C} y_i \log(s_i)$$

where C is the number of possible classes in a classification problem, y_i are the groundtruth of each class and s_i is the score of each class.

We can say that $\mathcal{L}_N$ is no more than the categorical cross-entropy of classifying the positive sample correctly, with the argument of the logarithm being the prediction of the model. If we note with $[d = i]$ as an indicator of the sample x_i being the positive sample in X, the optimal probability for this loss can be written as $p(d = i|X, c_t)$.

Now, the probability that x_i was drawn from the conditional distribution $p(x_{t+k}|c_t)$ that has the context in account, rather than the proposal distribution $p(x_{t+k})$ that does not have c_t in account, leads us to the following expression:

$$p(d = i|X, c_t) = \frac{\frac{p(x_i|c_t)}{p(x_i)}}{\sum_{j=1}^{N} \frac{p(x_j|c_t)}{p(x_j)}}.$$

This is the optimal case for (4). Lets provide with further explanation on this formula. Firstly, it is necessary to see that:

$$\begin{aligned} p(d=i|X) &= \frac{p(d=i,X)}{p(X)} \\ &= \frac{X|d=i}{p(X)} \\ &= \frac{p(X|d=i)}{\sum_{j=1}^{N} p(X|d=j)p(d=j)} \\ &= \frac{D(i)p(x_i)\prod_{j\neq i} q(x_j)}{\sum_{j=1}^{N} D(j)p(x_j)\prod_{k\neq j} q(x_k)} && (*) \\ &= \frac{p(x_i)\prod_{j\neq i} q(x_j)}{\sum_{j=1}^{N} p(x_j)\prod_{k\neq j} q(x_k)}, && \textit{assume } D \textit{ is uniform} \end{aligned}$$

Where, in $(*)$, we have used that since $p(X|d=i)$ reffers to the probability of the sample X given that $d=i$ which means that x_i has been extracted from $p(x_{t+k}|c_t)$ and x_j for $j \neq i$ have been extracted from $p(x_{t+k})$ (noted q in the formula to remark the difference). That means $p(X|d=i) = p(x_i)\prod_{j\neq i} q(x_j)$. Also, we can assume that D is uniform because each x_i with $i = 1, \ldots, N$ has the same probability to have been chosen to be the positive sample in X.

In fact, if we denote $f(x_{t+k}, c_t)$ as the density ratio that preserves the mutual information between x_{t+k} and c_t in the mutual information definition (3), we have just proved that if k is a step in the experiments, then

$$f_k(x_{t+k}, c_t) \propto \frac{p(t_{x+k}|c_t)}{p(x_{t+k})}, \tag{5}$$

where $\propto$ means that the member on the left is proportional to the member on the right. We can see that the optimal value $f_k(x_{t+k}, c_t)$ does not depend on $N-1$, the number of negative samples in X. Using this density ratio, we are relieved from modeling the high dimensional distribution x_{t_k}. In (Oord *et al.*, 2019), paper that we have followed and tried to explain in this document, for instance, the following log-bilinear model expression is used:

$$f_k(x_{t+k}, c_t) = exp(z_{t+k}^T W_k c_t).$$

In the proposed model, we can either use the representation given by the encoder (z_t) or the representation given by the autoregressive model (c_t) for downstream tasks. Clearly, the representation that aggregates information from past inputs will be more useful if more information about the context is needed. Furthermore, any type of models for the encoder and the autoregressive models can be used in this kind of framework.

Now, it will be shown how optimizing the loss presented in 4, we are maximizing mutual information between c_t and z_{t+k}. Using that the optimal value for $f(x_{t+k}, c_t)$ was proven to be $\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}$, if we have that:

$$L_N^{\text{opt}} = -E_X \log \left[\frac{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}}{\sum_{x_j \in X} \frac{p(x_j|c_t)}{p(x_j)}} \right].$$

We can split the denominator in the positive sample x_{t+k} and the negative samples, and use that $-log(a) = log(a^{-1})$ to obtain:

$$L_N^{\text{opt}} = E_X \left[\frac{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})} + \sum_{x_j \in X_{\text{neg}}} \frac{p(x_j|c_t)}{p(x_j)}}{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}} \right] = E_X \log \left[1 + \frac{\sum_{x_j \in X_{\text{neg}}} \frac{p(x_j|c_t)}{p(x_j)}}{\frac{p(x_{t+k})}{p(x_{t+k}|c_t)}} \right],$$

and, since the negatives are uniformly distributed, this we obtain:

$$E_X \log \left[1 + \frac{\sum_{x_j \in X_{\text{neg}}} \frac{p(x_j|c_t)}{p(x_j)}}{\frac{p(x_{t+k})}{p(x_{t+k}|c_t)}} \right] \approx E_X \log \left[1 + \frac{(N-1) E_{x_j} \left[\frac{p(x_j|c_t)}{p(x_j)} \right]}{\frac{p(x_{t+k})}{p(x_{t+k}|c_t)}} \right].$$

If we observe that $E_{x_j} \left[\frac{p(x_j|c_t)}{p(x_j)} \right] = \sum_{x_j} p(x_j) \frac{p(x_j|c_t)}{p(x_j)} = 1$, then we can derive the last expression as follows:

$$\begin{aligned} E_X \log \left[1 + \frac{(N-1) E_{x_j} \left[\frac{p(x_j|c_t)}{p(x_j)} \right]}{\frac{p(x_{t+k})}{p(x_{t+k}|c_t)}} \right] &= E_X \log \left[1 + \frac{N-1}{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}} \right] \\ &= E_X \log \left[1 + (N-1) \frac{p(x_{t+k})}{p(x_{t+k}|c_t)} \right] \end{aligned}$$

Part IV

APPENDIX A

This appendix will be used to set forth some theoretical results that might not always be relevant but are needed to understand some details during this thesis. Not all of them will be proven.

Proposition 4 (Jensen's Inequality). *Let $f : \mathcal{D} \to \mathbb{R}$ be a concave function and $n \in \mathbb{N}$. For any $p_1, \ldots, p_n \in \mathbb{R}_0^+$ with $\sum p_i = 1$ and any $x_1, \ldots, x_n \in \mathcal{D}$, it holds that:*

$$\sum_{i=1}^{n} p_i f(x_i) \leq f\left(\sum_{i=1}^{n} p_i x_i\right).$$

Furthermore, if f is strictly *concave and $p_i \geq 0$ for all $i = 1, \ldots, n$, then the equality holds if, and only if, $x_1 = \cdots = x_n$.*

BIBLIOGRAPHY

Bengio, Yoshua, Courville, Aaron, & Vincent, Pascal. 2014. Representation Learning: A Review and New Perspectives. *arXiv:1206.5538 [cs]*, Apr. arXiv: 1206.5538.

Chen, Ting, Kornblith, Simon, Norouzi, Mohammad, & Hinton, Geoffrey. 2020. A Simple Framework for Contrastive Learning of Visual Representations. *arXiv:2002.05709 [cs, stat]*, June. arXiv: 2002.05709.

Doersch, Carl, Gupta, Abhinav, & Efros, Alexei A. 2016. Unsupervised Visual Representation Learning by Context Prediction. *arXiv:1505.05192 [cs]*, Jan. arXiv: 1505.05192.

Gutmann, Michael U, & Hyvarinen, Aapo. Noise-Contrastive Estimation of Unnormalized Statistical Models, with Applications to Natural Image Statistics. 55.

He, Kaiming, Fan, Haoqi, Wu, Yuxin, Xie, Saining, & Girshick, Ross. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. *arXiv:1911.05722 [cs]*, Mar. arXiv: 1911.05722.

Hjelm, R. Devon, Fedorov, Alex, Lavoie-Marchildon, Samuel, Grewal, Karan, Bachman, Phil, Trischler, Adam, & Bengio, Yoshua. 2019. Learning deep representations by mutual information estimation and maximization. *arXiv:1808.06670 [cs, stat]*, Feb. arXiv: 1808.06670.

Löwe, Sindy, O'Connor, Peter, & Veeling, Bastiaan. 2019. Putting an End to End-to-End: Gradient-Isolated Learning of Representations. *Pages 3039–3051 of: Advances in Neural Information Processing Systems*.

Mikolov, Tomas, Chen, Kai, Corrado, Greg, & Dean, Jeffrey. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv:1301.3781 [cs]*, Sept. arXiv: 1301.3781.

Oord, Aaron van den, Li, Yazhe, & Vinyals, Oriol. 2019. Representation Learning with Contrastive Predictive Coding. *arXiv:1807.03748 [cs, stat]*, Jan. arXiv: 1807.03748.

Tian, Yonglong, Sun, Chen, Poole, Ben, Krishnan, Dilip, Schmid, Cordelia, & Isola, Phillip. 2020. What Makes for Good Views for Contrastive Learning? *arXiv:2005.10243 [cs]*, Dec. arXiv: 2005.10243.

Wiskott, Laurenz, & Sejnowski, Terrence J. 2002. Slow Feature Analysis: Unsupervised Learning of Invariances. *Neural Computation*, **14**(4), 715–770.