Foundations of Machine Learning II
TP2: Compression and Prediction
Text Entropy

Gaétan Marceau Caron & Guillaume Charpiat[*]

In the following, we are interested in compressing and generating texts written in natural languages[1]. To do so, we aim at minimizing the encoding cost:

$$-\ln p(X_1 = x_1, \dots, X_n = x_n) \tag{1}$$

where X_i is the random variable associated to the i^{th} character of a sequence $(x_1, \dots, x_n)$, by considering various models of p. We use two simple models, an IID model and a Markov model.

IID Model
The first model assumes independent and identically distributed (IID) random variables, i.e.

$$-\ln p(X_1 = x_1, \dots, X_n = x_n) = -\sum_{i=1}^{n} \ln p(X_1 = x_i). \tag{2}$$

For this model, a given character does not depend on the previous characters, thus we say that its order of dependencies is one. We could also generalize to symbol with multiple characters, and so, we define the order to be the number of characters. For instance, for order 3, the symbols are *aaa*, *aab*, ... *zzz*.

Markov chain model
The second model, the Markov chain, assumes limited range of dependency. For instance, a Markov chain of order 1 satisfies:

$$p(X_i|X_{i-1}, \dots, X_1) = p(X_i|X_{i-1}) \quad \text{(order-1 Markov property)}. \tag{3}$$

Moreover, we consider here only *time invariant* Markov chain, i.e. satisfying $p(X_i|X_{i-1}) = p(X_2|X_1)$ for all i. The order of a Markov chain is defined as the number of conditional variables: a Markov chain of order k satisfies:

$$p(X_i|X_{i-1}, \dots, X_1) = p(X_i|X_{i-1}, \dots, X_{i-k}) \quad \text{(order-k Markov property)}. \tag{4}$$

[*]https://www.lri.fr/~gcharpia/machinelearningcourse/
[1]The chosen books are available at `https://www.lri.fr/~marceau/Courses/CentraleML2/texts.zip`, thanks to the Gutenberg project.`https://www.gutenberg.org/`

Questions

1. Interpret the time invariant assumption associated to our Markov chains for text compression.
2. How can we rewrite a Markov chain of higher order as a Markov chain of order 1?
3. Given a probability distribution over symbols, how to use it for generating sentences?
4. As for supervised learning, a model can overfit the training set. Propose a simple approach (cost function) for measuring the goodness of the model.

Implementation For both models, perform the following steps:

1. For different orders of dependencies, train the model on a novel and compute the associated entropy. What do you observe as the order increases? Explain your observations.
2. Use the other novels as test sets and compute the cross-entropy for each model trained previouly. How to handle symbols (or sequences of symbols) not seen in the training set?
3. For each order of dependencies, compare the cross-entropy with the entropy. Explain and interpret the differences.
4. Choose the order of dependencies with the lowest cross-entropy and generate some sentences.
5. Train one model per novel and use the KL divergence in order to cluster the novels.

Implementation hints

1. It is possible to implement efficiently the two models with dictionaries in Python. For the IID model, a key of the dictionary is simply a symbol and the value is the number of occurences of the symbol in the text. For a Markov chain, a key of the dictionnary is also a symbol, but the value is a vector that contains the number of occurences of each character of the alphabet. Notice that a symbol may consist of one or several characters. Note also that there is no need to explicitly consider all possible symbols; the ones that are observed in the training set are sufficient.
2. A low probability can be assigned to symbols not observed in the training set. How to choose this probability?