# The Roots of Statistical Language Modeling

## Introductory Notes

A goal of statistical language modeling is to learn the joint probability function of sequences of words in a language. This is intrinsically difficult because of *the curse of dimensionality*: a word sequence on which the model will be tested is likely to be different from all the word seen during training. Traditional but very successful approaches based on n-grams obtain generalization by concatenating very short overlapping sequences seen in the training set. Bengio et al ([[1]](https://github.com/dimitarpg13/large_language_models/blob/main/articles/A_Neural_Probabilistic_Language_Model_bengio03a.pdf)) proposed to fight the curse of dimensionality by ***learning a distributed representation for words*** which allows each training sentence to inform the model about an exponential number of semantically neighboring sentences. The model learns simultaneously (1) a distributed representation of each work along with (2) the probability function for word sequences, expressed in terms of these representations.

Generalization is obtained because a sequence of words that has never been seen before gets high probability if it is made of works that are similar (in the sense of having a nearby representation) to words forming an already seen sentence. Training such large models within reasonable time is itself a significant challenge. Report is presented on experiments using neural networks for the probability function, showing on two text corpora that the proposed approach significantly improves on state-of-art n-gram models, and that the proposed approach allows to take advantage of longer contexts.

It was mentioned in the previous paragraph that language modeling is difficult due to the *curse of dimensionality*.

It is obvious in case one wants to model the joint distribution between many discrete random variables and has been discussed widely in the literature. For instance, words in a sentence or discrete attributes in a data-mining task are represented by such joint distribution and fall into this category of many random variables.

*Example*: if we want to model the joint distribution of 10 consecutive words in NLP with a vocabulary of size, there are potentially free parameters. When modeling continuous variables, we obtain generalization more easily (e.g. with smooth classes of functions like multi-layer networks or Gaussian mixture models) because the function to be learned can be expected to have some local smoothness properties. For discrete spaces, the generalization structure is not as obvious: any change of these discrete variables may have a drastic impact on the value of the function to be estimated, and when the number of values that each discrete variable can take is large, most observed objects are almost maximally far from each other in hamming distance.

A useful way to visualize how different learning algorithms generalize, inspired from the view of non-parametric density estimation, is to think of how probability mass that is initially concentrated on the training points (e.g. training sentences) is distributed in a larger volume, usually in some form of neighborhood around the training points.

## Literature

[1] [A Neural Probabilistic Language Model, Yoshua Bengio et al, Université de Montréal, Québec, 2003](https://github.com/dimitarpg13/large_language_models/blob/main/articles/A_Neural_Probabilistic_Language_Model_bengio03a.pdf)