Natural Language Processing

Michael Rose

In which we see how to make use of the copious knowledge that is expressed in natural language.

22.1 | Language Models

Languages can be vague. Thats why it is more fruitful to define a natural language model as a probability distribution over sentences rather than a definitive set. That is, rather than asking if a string of words is or is not a member of the set defining the language, we instead ask for $P(S = words)$ - what is the probability a random sentence would be words.

22.1.1 | N-gram Character Models

A model of the probability distribution of n-letter sentences is called an n-gram model. An n-gram model is defined as a **Markov chain** of order $n - 1$. For example, for a trigram model (a Markov chain of order 2), we have

$$P(c_i|c_{1:i-1}) = P(c_i|c_{i-2:i-1})$$

We can define the probability of a sequence of characters $P(c_{1:N})$ under the trigram model by first factoring with the chain rule and then using the Markov assumption:

$$P(c_{1:N}) = \prod_{i=1}^{N} P(c_i|C_{1:i-1}) = \prod_{i=1}^{N} P(c_i|C_{i-2:i-1})$$

For a trigram character model in a language with 100 characters, $P(C_i|C_{i-2:i-1})$ has a million entries, and can be accurately estimated by counting character sequences in a body of text of 10 million characters or more. We call a body of text a **corpus**, from the Latin word for body.

One task for which these are suited is **language identification**. We can build a trigram character model of each candidate language, $P(c_i|c_{i-2:i-1}, l)$, where the variable l ranges over languages. For each l the model is built by counting trigrams in a corpus of that language (about 100,000 characters of each language are needed). That will give us a model of $P(Text|Language)$, but we want to select the most probable language given the text, so we use Bayes' rule followed by the Markov assumption to get the most probable language:

$$\begin{aligned} l^* &= \arg\max_l P(l|C_{1:N}) \\ &= \arg\max_l P(l)P(C_{1:N}|l) \\ &= \arg\max_l P(l) \prod_{i=1}^{N} P(c_i|C_{i-2:i-1}, l) \end{aligned}$$

The exact number for our priors is not critical because the trigram model usually selects one language that is several orders of magnitude more probable than any other.

Other tasks for character models include spelling correction, genre classification, and named-entity recognition.

22.1.2 | Smoothing n-gram Models

A major complication with n-gram models that the training corpus only provides an estimate of the true probability distribution. If our corpus gives us a 0% chance of a pattern occuring, this could cause problems down the road. We want our models to generalize well to texts that they haven't seen yet. Thus, we adjust our language model so that sequences that have a count of 0 in the training corpus will be assigned a small nonzero probability (and the other counts will be adjusted slightly downward so that the probability still sums to 1). The process of adjusting the probability of low-frequency counts is called **smoothing**.

The simplest type of smoothing is Laplace Smoothing which says that, in the lack of further information, if a random Boolean variable X has been false in all n observations, then the estimate for $P(X = true)$ should be $\frac{1}{(n+2)}$. He assumes that with two more trials, one might be true and one false. Laplace smoothing performs relatively poorly.

A better performing smoothing algorithm is a **backoff model**, in which we start by estimating n-gram counts, but for any particular sequence that has a low or zero count, we back off to (n-1)-grams. **Linear Interpolation Smoothing** is a backoff model that combines trigram, bigram and unigram models by linear interpolation. If defines the probability estimate as

$$\hat{P}(c_i|c_{i-2:i-1}) = \lambda_3 P(c_i|c_{i-2:i-1}) + \lambda_2 P(c_i|c_{i-1}) + \lambda_1 P(c_i)$$

where $\lambda_3 + \lambda_2 + \lambda_1 = 1$. The parameters λ_i can be fixed, or they can be trained with an expectation-maximization algorithm.

22.1.3 | Model Evaluation

A way of describing the probability of a sequence is with a measure called **perplexity**, defined as

$$Perplexity(c_{1:N}) = P(c_{1:N})^{-\frac{1}{N}}$$

Perplexity can be thought of as the reciprocal of probability, normalized by sequence length. It can also be thought of as the weighted average branching factor of a model.

22.2 | Text Classification