Statistical Language Models

Introduction

A Statistical Language Model (LM) is a probability distribution, where words, characters, and documents are used as inputs. Probabilities of all outcomes sum to 1. The output is an unnormalized score for ranking. An Incremental LM is a model that provide probability of a given word basing on preceding words. In the following review, several LMs will be described and compared.

n-gram LMs

n-gram LMs base their prediction on the previous word(s).
Zero-gram LM is a special n-gram LM. In a zero-gram LM, all words are assumed to follow uniform distribution and assigned equal probabilities:

$$p(w_i) = \frac{1}{V}$$

, where V is the vocabulary. However, this is not ideal as we know words like "the" should have a lot higher probability then words like "science". Unigram model improves this by estimating term probabilities basing on word count, using maximum likelihood estimation. In this model,

$$p(w_i) = count(w_i)/count(w)$$

, where count(w) is the total count of all words in the document. But with this assumption, in a sentence "I love computer __", a unigram model will assign too much probability to word "the" and not enough probability to word "science".
To further improve this, a bigram model estimates probabilities basing on bigram and word counts:

$$p(w_i|w_{i-1}) = count(w_{i-1}, w_i)/count(w_{i-1})$$

Thus, in the sentence "I love computer __", previous word "computer" also plays a part when predicating the blank. Trigram and other n-gram LMs uses longer contiguous history. For example, trigram LM looks like this:

$$p(w_i|w_{i-2}, w_{i-1}) = count(w_{i-2}, w_{i-1}, w_i)/count(w_{i-2}, w_{i-1})$$

Skip LMs

Skip LMs are similar to n-gram LMs, in terms of allowing conditioning on previous words. It further allows intervening and skipping some history. For example,

$$p(w_i|w_{i-2}) = count(w_{i-2}, w_i)/count(w_{i-2})$$

, where prediction depends on the second preceding word, rather than the previous word. This captures basic word order variation with many possible combinations of histories to use. But it also unnecessarily fragments the training data instead of generalizing it.

Class LMs

Class LMs bring in another layer of information and groups similar words, for example "Monday" and "Tuesday". They are easy to use and useful for small to medium sized corpora. However, they are poor at handling fixed phrases and multi-word expressions.

Topic LMs

Both class LMs and topic LMs use a bottleneck variable to generalize the history, while topic LMs generalize the long-term lexical history while class LMs generalizes the shot-term grammatical history. In a topic LM, documents are clustered into a set of topics. This model is useful for domain adaptation and widely used in information retrieval. However, they are slow and don't scale up well. Thus, they are often combined with other LMs.

Neural Net LMs

Neural Net LMs was originally inspired by biological neurons. It has multiple hidden layers to allow multiple levels of generalization. Recurrent Neural Net LMs further allows previous hidden state feed into current hidden state and can capture longer dependencies. This is also known as "Elman Networks"

Conclusions

All of the LMs reviewed are incremental as they all condition on preceding words. n-gram LM has shortest distance as it only conditions on several previous words. While topic LM has contingency on the long-term lexical history. Topic and Neural Net LMs are slow to train due to the multiple layers they have in the model. The comparisons are summarized in the following table.

LMs	Incremental	Lexical	Distance	Speed
n-gram	Y	Y	Short	Fast
Skip	Y	Y	Medium	Fast
Class	Y	N	Medium	Fast
Topic	Y	N	Long	Slow
NN/RNN	Y	Y	Medium	Slow

Reference

https://jon.dehdari.org/tutorials/lm_overview.pdf