shift probability mass from the observed tuples to unobserved ones that are similar. See Chen and Goodman (1999) for a review and empirical comparisons. One basic technique consists of adding non-zero probability mass to all of the possible next symbol values. This method can be justified as Bayesian inference with a uniform or Dirichlet prior over the count parameters. Another very popular idea is to form a mixture model containing higher-order and lower-order n-gram models, with the higher-order models providing more capacity and the lower-order models being more likely to avoid counts of zero. **Back-off methods** look-up the lower-order n-grams if the frequency of the context $x_{t-1}, \ldots, x_{t-n+1}$ is too small to use the higher-order model. More formally, they estimate the distribution over x_t by using contexts $x_{t-n+k}, \ldots, x_{t-1}$, for increasing k, until a sufficiently reliable estimate is found.

Classical n-gram models are particularly vulnerable to the curse of dimensionality. There are $|\mathbb{V}|^n$ possible n-grams and $|\mathbb{V}|$ is often very large. Even with a massive training set and modest n, most n-grams will not occur in the training set. One way to view a classical n-gram model is that it is performing nearest-neighbor lookup. In other words, it can be viewed as a local non-parametric predictor, similar to k-nearest neighbors. The statistical problems facing these extremely local predictors are described in section 5.11.2. The problem for a language model is even more severe than usual, because any two different words have the same distance from each other in one-hot vector space. It is thus difficult to leverage much information from any "neighbors"—only training examples that repeat literally the same context are useful for local generalization. To overcome these problems, a language model must be able to share knowledge between one word and other semantically similar words.

To improve the statistical efficiency of n-gram models, **class-based language models** (Brown *et al.*, 1992; Ney and Kneser, 1993; Niesler *et al.*, 1998) introduce the notion of word categories and then share statistical strength between words that are in the same category. The idea is to use a clustering algorithm to partition the set of words into clusters or classes, based on their co-occurrence frequencies with other words. The model can then use word class IDs rather than individual word IDs to represent the context on the right side of the conditioning bar. Composite models combining word-based and class-based models via mixing or back-off are also possible. Although word classes provide a way to generalize between sequences in which some word is replaced by another of the same class, much information is lost in this representation.