Chapter 5. Language Modeling

[5.1.] Introduction:

A statistical language model is used in human language technology to calculate the probability of a word sequence in a language. It uses an alphabet or inventory of units and a sequence W, which can be challenging to define in languages other than English.

Language models are used in speech recognition, machine translation, information retrieval, authorship identification, and document classification. They are often combined with other models to hypothesize possible word sequences. Language models are also used in acoustic units or isolated text characters, such as phonemes or phonemes. This chapter focuses on language models over natural language words or word like units, discussing the fundamental n-gram modeling approach, advanced modeling techniques, and problems arising from language characteristics like morphologically rich or unstructured languages.

[5.2.] *n*-Gram Models:

The chain rule of probability can be used to decompose the probability of a word sequence of nontrivial length into component probabilities.

$$P(W) = P(w_1 \dots w_t) = P(w_1) \prod_{i=1}^{t} P(w_i | w_{i-1} w_{i-2} \dots w_2 w_1) \qquad (5.1)$$

Statistical language models use the n-gram approximation, which assumes all previous words except the n-1 words preceding the current word are irrelevant or equivalent for predicting the current word.

$$P(W) \approx \prod_{i=1}^{t} P(w_i | w_{i-1}, \dots w_{i-n+1}) \qquad (5.2)$$

An n-gram model, also known as an (n - 1)-th order Markov model, is a type of word that is independent of all other words except the n - 1 preceding words.

[5.3]Language Model Evaluation:

The performance of a language model is evaluated using two criteria: coverage rate and perplexity. Coverage rate measures the percentage of n-grams in a test set, while perplexity is an information theoretic measure of the number of unique word types not covered by the model. These criteria help determine a model's success in estimating word sequence probabilities.

$$PPL(p) = 2^{H(p)} = 2^{-\sum_x p(x) log_2 p(x)} \qquad (5.3)$$

To make a meaningful comparison between different language models, it's crucial to normalize their perplexities with respect to the same number of units.

$$PPL(p, q) = 2^{H(p,q)} = 2^{-\sum_{i=1}^{t} p(w_i) log_2 q(w_i)} \qquad (5.4)$$

or simply

$$2^{-\frac{1}{t}\sum_{i=1}^{t} log_2 q(w_i)} \qquad (5.5)$$

where $q(w_i)$ computes the probability of the $i'th$ word. If $q(w_i)$ is an n-gram probability, the equation becomes

$$2^{-\frac{1}{t}\sum_{i=1}^{t} log_2 p(w_i|w_{i-1},\ldots,w_{i-n+1})} \qquad (5.6)$$

Perplexity is the average number of equally likely successor words in a word string, ranging from zero to one. The goal in language model development is to minimize perplexity on a representative domain.

Language modeling aims to distinguish between "good" and "bad" word sequence hypotheses in frontend systems like machine translation or speech recognition, assigning maximally distinct scores to errorful, ungrammatical, or unacceptable word sequences, but not necessarily achieving minimum perplexity.

[5.4] Parameter Estimation:

5.4.1. Maximum-Likelihood Estimation and Smoothing

The standard procedure in training *n*-gram models is to estimate *n*-gram probabilities using the maximum-likelihood criterion in combination with parameter smoothing. The maximumlikelihood estimate is obtained by simply computing relative frequencies:

$$P(w_i|w_{i-1}, w_{i-2}) = \frac{c(w_i, w_{i-1}, w_{i-2})}{c(w_{i-1}, w_{i-2})} \qquad (5.7)$$

The method of assigning nonzero probabilities to word sequences in training data is called smoothing, which involves redistributing probability mass to flatten peaks and floor zero estimates. Backoff is a common smoothing technique, splitting n-grams into those with counts below a threshold and those with counts above. The backed-off probability (PBO) for wi is computed by dividing the n-grams by a factor.

The distribution is modeled using a discounting factor dc and a normalization factor α (wi–1, wi–2), ensuring the entire distribution sums to one.

$$P_{BO}(w_i|w_{i-1}, w_{i-2}) = \begin{cases} d_c P(w_i|w_{i-1}, w_{i-2}) \text{ if } c > \tau \\ \alpha(w_{i-1}, w_{i-2}) P_{BO}(w_i|w_{i-1}) \text{ otherwise} \end{cases} \qquad (5.8)$$

The precise smoothing technique, including Good-Turing, Witten-Bell, and Kneser-Ney, is determined by the computation of the discounting factor, as described in Chen and Goodman's study.

$$\alpha(w_{i-1}, w_{i-2}) = \frac{1 - \sum_{w_i : c(w_i, w_{i-1}, w_{i-2}) > \tau} d_c P(w_i|w_{i-1}, w_{i-2})}{\sum_{w_i : c(w_i, w_{i-1}, w_{i-2}) \leq \tau} P_{BO}(w_i|w_{i-1})} \qquad (5.9)$$

In modified Kneser-Ney smoothing, which is one of the most widely used techniques, different discounting factors D_1, D_2, D_{3+} are used for *n*-grams with exactly one, two, or three or more counts:

$$Y = \frac{n_1}{n_1 + 2 * n_2} \quad (5.11)$$

$$D_1 = 1 - 2Y\frac{n_2}{n_1} \quad (5.12)$$

$$D_2 = 2 - 3Y\frac{n_3}{n_2} \quad (5.13)$$

$$D_{3+} = 3 - 4Y\frac{n_4}{n_3} \quad (5.14)$$

where n_1, n_2, ... are the counts of n-grams with one, two, ..., counts.

Another common way of smoothing language model estimates is linear model interpolation [8]. In linear interpolation, M models are combined by

$$P(w_i|w_{i-1}, w_{i-2}) = \sum_{m=1}^{M} \lambda_m P(w_i|h_m) \quad (5.15)$$

Linear model interpolation is a method for smoothing language model estimates by combining multiple models with a specific weight, λ. The weights are estimated using the expectation-maximization (EM) procedure, minimizing perplexity on a different data set from the training and final evaluation sets, ensuring a consistent model weight.

5.4.2. Bayesian Parameter Estimation

Bayesian probability estimation is a method where a model's parameters are viewed as random variables with a prior statistical distribution, and the posterior distribution is expressed using

$$P(\theta|S) = \frac{P(S|\theta)P(\theta)}{P(S)} \quad (5.16)$$

Bayes's rule.

Language modeling involves a set of parameters, such as word probabilities (θ) for unigram models and n-gram models, with a training sample of words. A point estimate of θ is required, using either the maximum a posteriori (MAP) criterion or the Bayesian criterion, as per Equation 5.16
The Dirichlet distribution is the most commonly used prior distribution in language models, as it is the conjugate prior to the multinomial distribution, and its estimation functions are equivalent

to the maximum-likelihood estimate with Laplace smoothing.

$$\theta^B = E[\theta|S] = \int_\Theta \theta P(\theta|S)d\theta \quad (5.18)$$

$$= \frac{\int_\Theta \theta P(S|\theta)P(\theta)d\theta}{\int_\Theta P(S|\theta)P(\theta)d\theta} \quad (5.19)$$

Under the assumption that the prior distribution is a uniform distribution, the

The MAP estimate of P(θ|W, α) is equivalent to the maximum-likelihood estimate with add-m-smoothing, where pseudo counts of size ακ - 1 are added to each word count. Hyper parameters integrate different information sources into the language model estimation process, most successfully used for language model adaptation. Early Bayesian estimation models did not perform as well as standard n-gram models, but recent progress has developed alternative models that yield results comparable to Kneser-Ney smoothed n-gram models.

5.4.3. Large-Scale Language Models

The increasing availability of monolingual data has led to increased interest in scaling language models to large data sets, requiring modifications in training, storage, and integration into real-world systems, and potentially affecting parameter estimation.

Distributed approaches to large-scale language modeling involve subdividing training data into partitions and storing probabilities in separate locations. Clients can request statistics from these partitions during runtime, producing probability estimates on demand. This approach scales to large data and vocabulary sizes, allowing dynamic data addition without recomputed static model parameters. However, it has a slow networked query speed.

In Brants et al. [13], a nonnormalized form of backoff is introduced that differs from standard backoff (Equation 5.8) in that it uses the raw relative frequency estimate instead of a discounted probability if the *n*-gram count exceeds the minimum threshold (in this case 0):

$$S(w_i|w_{i-1},w_{i-2}) = \begin{cases} P(w_i|w_{i-1},w_{i-2}) \text{ if } c > 0 \\ \alpha S(w_i|w_{i-1}) \text{ otherwise} \end{cases} \quad (5.23)$$

The α parameter is fixed for all contexts rather than being dependent on the

The α parameter is fixed across all contexts, resulting in a set of unnormalized scores (S) used similarly to standard probabilities. This approach simplifies computation in a distributed framework, eliminating the need for summing over all n-gram contexts. The model performs similarly on large data sets.

$$q(w_1 \ldots w_n) = 1 + [log_b c(w_1 \ldots w_n)] \tag{5.24}$$

At test time, the necessary statistics are queried from the filter, and the true frequency is approximated by the expected count given the quantized count:

$$E[c(w_1 \ldots w_n)|q(w_1 \ldots w_n) = j] = \frac{b^{j-1} + b^j - 1}{2} \tag{5.25}$$

Large-scale distributed language models can be used in second-pass rescoring stages after first-pass hypotheses are generated using smaller models. Another approach is to store large-scale models in a single machine's working memory using efficient, potentially lossy data structures like Bloom filters, representing corpus statistics quantized.

The Bloom filter language model, which uses fast data structure querying, can compute smoothed probabilities on the fly, despite potential overestimation. This method offers similar performance to exact parameter estimation and provides significant memory savings, despite potential inaccurate raw frequency counts.

Large-scale language modeling is shifting towards approximate techniques over exact parameter estimation, expected to continue with more refined approximation techniques as text data collections grow.

[5.5] Language Model Adaptation:

Language model adaptation is crucial when insufficient training data is available for a new domain or language, as it involves designing and tuning a model to perform well on the new test set.

The most common adaptation method is mixture language models, or model interpolation, which involves training an in-domain language model with a small amount of in-domain data and a larger background or generic model with a large amount of out-of-domain data. This approach can be generalized to more than two models and has several variations.

Topic-dependent language model adaptation involves clustering documents into various topics, building individual language models for each, and fine-tuning the final model by choosing and interpolating a smaller number of topic-specific language models. Dynamic self-adaptation of a language model is provided by trigger models, which suggest that certain word combinations are more likely to co-occur based on the underlying text topic. Latent semantic analysis (LSA) and probabilistic latent semantic analysis (PLSA) are also used for clustering words according to topic and using them as trigger pairs.

Language model adaptation is a complex process that involves incorporating trigger relationships within constraint-based modeling frameworks. The basic, non probabilistic LSA approach is extended by assuming a more sophisticated latent class model to decompose the word-document co-occurrence matrix. However, PLSA has a potential concern of over fitting to the training data. Latent Dirichlet allocation (LDA) is a more recent form of topic-based clustering that can be interpreted as a regularized version of PLSA.

$$P(w_i|h_i,\widetilde{h}_i) = \frac{P(w_i|h_i)\rho(w_i,\widetilde{h}_i)}{Z(h_i,\widetilde{h}_i)} \qquad (5.26)$$

Unsupervised adaptation is a variant of the standard adaptation framework, primarily relevant for speech recognition applications. It is possible to directly use the output of a speech recognizer instead of written text or perfectly transcribed speech as adaptation data. The Internet has become a common resource for additional language model data, and rapid adaptation methods based on this general procedure have been described.

$$P(w,d) = \sum_c P(c)P(w|c)P(d|c) = P(d)\sum_c P(c|d)P(w|c) \qquad (5.27)$$

An alternative probability estimation scheme for language model adaptation is maximum a posteriori (MAP) adaptation, which combines counts from generic out-of-domain (OD) and in-domain adaptation (ID) data. The ε parameter ranges between 0 and 1 and represents the weight assigned to the adaptation data. A comparison of MAP and mixture models has shown that mixture models are less robust than MAP adaptation toward variability in the adaptation data.

Language model adaptation has been performed in speech recognition, machine translation, and crosslingual data.

[5.6] Types of Language Models:

Despite being the most widely used statistical language model, alternative models have been developed for practical applications, often used in combination with n-gram models.
5.6.1. Class-Based Language Models
5.6.2. Discriminative Language Models
5.6.3. Syntax-Based Language Models
5.6.4. MaxEnt Language Models

5.6.1. Class-Based Language Models

Class-based language models [36] are a simple way of addressing data sparsity in language modeling. Words are first clustered into classes, either by automatic means [37] or based on linguistic criteria, for example, using part-of-speech (POS) classes. The statistical model makes the assumption that words are conditionally independent of other words given the current word class. If c_i is the class of word w_i, a class-based bigram model can be defined as follows:

Goodman [38] compared this decomposition to the following model:

where the current word is conditioned not only on the current word class but also on the preceding word classes. Experiments on the North American Business News Corpus (with the training size ranging between 100,000 and 284 million words), using 20,000 test sentences and a vocabulary of 58,000 words showed that the model in Equation 5.32 worked better unless the training data size was at the lower end. Class-based models have been successful in reducing perplexity as well as practical performance in a wide range of language processing systems; however, they typically need to be interpolated with a word-based language model.

$$P(w_i|w_{i-1}) = \sum_{c_i, c_{i-1}} P(w_i|c_i)p(c_i|c_{i-1}, w_{i-1})P(c_{i-1}|w_{i-1}) \qquad (5.29)$$

$$= \sum_{c_i, c_{i-1}} P(w_i|c_i)P(c_i|c_{i-1})P(c_{i-1}|w_{i-1}) \qquad (5.30)$$

under the assumption that c_i is independent of w_{i-1} given c_{i-1}. Usually, a class will contain more than one word, such that the model simplifies to

$$P(w_i|w_{i-1}) = P(w_i|c_i)P(c_i|c_{i-1}) \qquad (5.31)$$

Goodman [38] compared this decomposition to the following model:

$$P(w_i|w_{i-1}) \approx P(w_i|c_i, c_{i-1})P(c_i|c_{i-1}) \qquad (5.32)$$

where the current word is conditioned not only on the current word class but also on the preceding word classes. Experiments on the North American Business News Corpus (with the training size ranging between 100,000 and 284 million words), using 20,000 test sentences and a vocabulary of 58,000 words showed that the model in Equation 5.32 worked better unless the training data size was at the lower end. Class-based models have been successful in reducing perplexity as well as practical performance in a wide range of language processing systems; however, they typically need to be interpolated with a word-based language model.

Variable-length n-gram models are data-driven approaches to redefining vocabulary units, resulting in merged units composed of a variable number of basic units. These models aim to find the best segmentation of the word sequence into language modeling units and estimate language model probabilities. Deligne and Bimbot [39] model the segmentation as a hidden variable and use an ME procedure to find the best segmentation. A variable-length model of order 7 yielded slight improvements in perplexity compared to a standard word-based bigram model, but no application results were reported.

The text suggests a simpler method for identifying phrasal units in language models by starting with the initial whitespace-based segmentation suggested by the standard orthography and merging selected units. This approach reduces the complexity of the development corpus by selecting candidates that reduce perplexity. A greedy iterative algorithm is used to select phrasal units, with each iteration reducing the perplexity of the development corpus. In a study by Zitouni, Smaili, and Haton, word class information is used to identify candidate phrasal units, reducing perplexity by about 10% compared to word-based selection. This model also achieved an 18% reduction in worderror rate on a medium-scale French automatic speech recognition task.

5.6.2. Discriminative Language Models

Standard n-gram models are generative models that assign probability to word sequences, but in practical applications like machine translation or speech recognition, they often need to separate good sentence hypotheses from bad ones. To achieve this, language models should be trained discriminatively, ensuring that word strings of varying quality receive maximally distinct probability estimates. Recent attempts at discriminative language modeling include Roark et al., Collins, Saraçlar, and Roark, Shafran and Hall, and Arisoy et al. The language model is applied to competing sentence hypotheses generated for an input x, and arbitrary feature functions φ(x, y) are defined jointly over the input and each output y ∈ Y. These functions are then used in a global linear model to select the best hypothesis.

$$F(x) = \underset{y \in \mathrm{GEN}(x)}{\operatorname{argmax}} \; \phi(x,y)\alpha \qquad (5.33)$$

The parameter vector α, which represents the weight vector of feature functions, can be trained using the perceptron algorithm or a conditional log-linear model. The perceptron algorithm iterates through all training samples and selects the best-scoring hypothesis for each sample. If it differs from the true reference hypothesis, it updates the current weights by adding the counts of features in the correct hypothesis and subtracting their counts in the chosen hypothesis. This training procedure directly minimizes the desired objective function, such as word-error rate in a speech recognition system. The weights with which the counts of different n-grams contribute to the final model are trained to optimize system performance rather than minimize the perplexity criterion. Recent studies have reported a 1.8% absolute improvement in word-error rate on a large-vocabulary speech recognition task for a one-pass system and a 0.9% reduction for a multipass recognizer. A discriminative language model has also been applied to statistical machine translation, resulting in an improvement of 1 to 2 BLEU points over a state-of-the-art baseline system.

5.6.3. Syntax-Based Language Models

Syntax-based language models have been developed to address the drawbacks of n-gram language models, which cannot consider relevant words outside the limited window of the directly preceding n-1 words. Natural language exhibits long-distance dependencies, where the choice of the current word is dependent on words relatively far removed in terms of sentence position. To address this problem, several approaches have been developed, such as Chelba and Jelinek's structured language model, which computes the joint probability of a word sequence and its parse and decomposes it into a product of component probabilities involving various elements of the word sequence proper, head words from the parse structure, and POS tags in the parse structure. Results show that a structured language model interpolated with a trigram model achieved a relative reduction in perplexity of 8% on the Wall Street Journal Continuous Speech Recognition (CSR) and Switchboard corpora, and a 6% relative reduction on the Wall Street Journal corpus and a 0.5% absolute reduction on Switchboard. Another syntax-based model is Wang and Harper's "almost-parsing" language model, also called the SuperARV model, which is based on a constraint-dependency grammar. In this model, sentences are annotated with SuperARVs, rich tags combining lexical features and syntactic information pertaining to a word.

Syntax-based language models have been developed to address the drawbacks of n-gram language models, which cannot consider relevant words outside the limited window of the directly preceding n-1 words. Natural language exhibits long-distance dependencies, where the choice of the current word is dependent on words relatively far removed in terms of sentence position.

To address this problem, several approaches have been developed, such as Chelba and Jelinek's structured language model, which computes the joint probability of a word sequence and its parse and decomposes it into a product of component probabilities involving various elements of the word sequence proper, head words from the parse structure, and POS tags in the parse structure.

Results show that a structured language model interpolated with a trigram model achieved a relative reduction in perplexity of 8% on the Wall Street Journal Continuous Speech Recognition (CSR) and Switchboard corpora, and a 6% relative reduction on the Wall Street Journal corpus and a 0.5% absolute reduction on Switchboard. Another syntax-based model is Wang and Harper's "almost-parsing" language model, which is based on a constraint-dependency grammar. In this model, sentences are annotated with SuperARVs, rich tags combining lexical features and syntactic information pertaining to a word.

$$P(w_1, \ldots, w_N, t_1, \ldots, t_N) = \prod_{i=1}^{N} P(w_i t_i | w_1 \ldots, w_{i-1}, t_1 \ldots t_{i-1}) \tag{5.34}$$

$$= \prod_{i=1}^{N} P(t_i | w_1 \ldots, w_{i-1}) P(w_i | w_1 \ldots w_{i-1}, t_1 \ldots t_{i-1}) \tag{5.35}$$

$$\approx \prod_{i=1}^{N} P(t_i | w_{i-2}, w_{i-1}, t_{i-1}, t_{i-1}) P(w_i | w_{i-2}, w_{i-1}, t_{i-2}, t_{i-1}) \tag{5.36}$$

The model is smoothed by recursive linear interpolation of higher-order and lower-order models. The SuperARV model was evaluated on the Wall Street Journal Penn Treebank and CSR tasks and was compared to other parser-based language models, including the structured language model described previously, as well as to standard trigram and POSbased models. It was found that the SuperARV achieved the lowest perplexity scores out of all. When used for lattice rescoring on the CSR task, the SuperARV model yielded relative word-error rate reductions between 3.1% and 13.5% and again outperformed all other models.

5.6.4. MaxEnt Language Models

One shortcoming of maximum-likelihood-based probability estimation for language models is that the constraints imposed by estimates solely derived from the training data are too strong. MaxEnt models represent an alternative where these constraints are relaxed. Rather than setting the probability of a given *n*-gram to its relative frequency in the training data (modulo smoothing), the requirement of MaxEnt modeling is that the model agree, on average, with the observed counts of events in the training data. The MaxEnt model is formulated as follows:

$$P(y|x) = \frac{1}{Z(x)} \exp\left(\sum_k \lambda_k f_k(x,y)\right) \qquad (5.37)$$

where $f(x, y)$ is a feature function defined both on the input and the predicted variables, λ is a feature function specific weight, and $Z(x)$ is a normalization factor, computed as

$$Z(x) = \sum_{y \in Y} \exp\left(\sum_k \lambda_k f_k(x,y)\right) \qquad (5.38)$$

Once appropriate feature functions have been defined, the expected value of f_κ is

$$E(f_k) = \sum_{x \in X, y \in Y} \widetilde{p}(x) p(y|x) f_k(x,y) \qquad (5.39)$$

where $\widetilde{p}(x)$ is the empirical distribution of x in the training data. The empirical expectation of f_k (derived from the training data) is

$$\widetilde{E}(f_k) = \sum_{x \in X, y \in Y} \widetilde{p}(x,y) f_k(x,y) \qquad (5.40)$$

The model is then trained such that expected values match empirical expected values

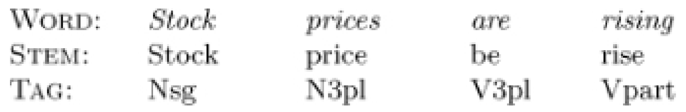
$$E(f_k) = \widetilde{E}(f_k) \qquad \forall k \qquad (5.41)$$

The MaxEnt framework was first applied to language modeling by Rosenfeld [52]. In the context of language modeling, y represents the predicted word and x the history or, more generally, the conditioning variables used for prediction. Note that in this case, a much larger context than the $n - 1$ directly preceding words may be included: feature functions may be defined over the entire sentence [53] or over an even larger domain. Most commonly, however, feature functions are simply defined over n-grams; for example, for a given word w_i and history h_i, a bigram feature function would be defined as follows:

Wu and Khudanpur's study on a MaxEnt model for speech recognition on the Switchboard task showed a 7% perplexity reduction and a 0.7% word-error rate reduction. The model also showed a 7% perplexity reduction and a 0.8% reduction in word-error rate when syntactic constraints were integrated. The combination of both types of constraints improved the overall performance.

5.6.5. Factored Language Models

The factored language model (FLM) approach [62, 63] builds on the observations that the prediction of words is dependent on the surface form of the preceding words and that better generalizations can be made when additional information, such as the word's POS or morphological class, is taken into account. In particular, word n-gram counts might be insufficient to robustly estimate the probability of word w_i given word w_{i-1}, but if we know that word w_{i-1} is of a particular class, say a determiner, we might be able to obtain a good probability estimate for $P(w_i|\text{determiner})$. This is reminiscent of the class-based models described previously; however, in an FLM, many such class-based estimates are combined and structured hierarchically through the use of a **generalized backoff** strategy. FLMs assume a factored word representation, where words are considered feature vectors rather than individual surface forms; that is, $W \equiv f_{1:K}$. An example is shown



WORD:	*Stock*	*prices*	*are*	*rising*
STEM:	Stock	price	be	rise
TAG:	Nsg	N3pl	V3pl	Vpart

The word surface form itself can be one of the features. A statistical model over this representation can be defined as follows (using a trigram approximation):

$$p\left(f_1^{1:K}, f_2^{1:K}, \ldots, f_t^{1:K}\right) \approx \prod_{i=3}^{t} p\left(f_i^{1:K} | f_{i-1}^{1:K}, \ldots, f_{i-2}^{1:K}\right) \qquad (5.45)$$

Words in a language model (FLM) depend on both temporally ordered and parallel feature variables. Standard backoff involves moving from higher-order to lower-order distributions. However, in FLM, backoff is not immediately clear due to temporally ordered and parallel conditioning variables. A decision must be made on which subset of features to back off to in which order. There are several ways to choose backoff paths: based on linguistic knowledge, statistical criteria at runtime, or multiple paths and their probability estimates. The last option, parallel backoff, is implemented using a generalized backoff function.

$$P_{GBO}(f|f_1, f_2) = \begin{cases} d_c P_{ML}(f|f_1, f_2) \text{ if } c > \tau \\ \alpha(f_1, f_2) g(f, f_1, f_2) \text{ otherwise} \end{cases} \qquad (5.46)$$

where, similar to Equation 5.8, c is the count of (f, f_1, f_2), $P_{ML}(f|f_1, f_2)$ is the maximum likelihood estimate, τ is the count threshold, and $\alpha(f_1, f_2)$ is the normalization factor that ensures that the resulting scores form a probability distribution. The function $g(f, f_1, f_2)$ determines the backoff strategy. In a typical backoff procedure, $g(f, f_1, f_2)$ equals $P_{BO}(f|f_1)$. In generalized parallel backoff, however, g can be any nonnegative function of f, f_1, f_2 and can be instantiated to, for example, the mean, weighted mean, product, or maximum functions. For example, the mean function would take the average of the individual estimates:

$$g_{\text{mean}}(f, f_1, f2) = 0.5P_{BO}(f|f_1) + 0.5P_{BO}(f|f_2) \tag{5.47}$$

In addition to different choices for g, different discounting parameters can be chosen at different levels in the backoff graph.

It is not a priori obvious which backoff strategy works best; the optimal strategy is highly dependent on the particular language modeling task. Because the space of possible factored language model structures and backoff parameters is very large, it is advisable to use an automatic, data-driven procedure to find the best settings. An automatic FLM optimization procedure based on genetic algorithms was proposed by Duh and Kirchhoff [64].

FLMs have been implemented as an add-on to the widely used SRILM (Stanford Research Institute Language Modeling) toolkit [65] and have been used successfully for morpheme-based language modeling [62], multispeaker language modeling [66], dialog act tagging [67], and speech recognition [68, 63], especially in sparse data scenarios such as highly inflecting languages (see also §5.7.2).

5.6.6. Other Tree-Based Language Models

The hierarchical class-based backoff model, proposed by Zitouni, is a language modeling approach that uses tree structures to backoff words along a hierarchy of word classes. This model is different from Tree-Based Language Models (FLMs) as it is fixed and predefined in advance, allowing for the combination of probability estimates from different paths in the backoff graph and dynamic choice of a path at runtime. Zitouni found that this model yields gains primarily when the test set contains a large number of unseen events. On a speech recognition language modeling task with a 5,000-word vocabulary, unseen word perplexity was reduced by 10%, while on a task with a 20,000-word vocabulary, it was reduced by 26% and the word-error rate decreased by 12%. Wang and Vergyri proposed extensions to this hierarchical class n-gram language model, integrating POS information into the word clustering process. On an Egyptian Colloquial Arabic speech recognition task, the model achieved 8% relative improvement in perplexity and a 3% relative improvement over the model described by Zitouni.

Random forest language models (RFLMs) are a method proposed by Xu and Jelinek to model word histories in training data as a collection of randomly grown decision trees. These trees are created by dividing the set of histories into two subsets based on the word identity at a particular position in the history. The split that maximizes the log-likelihood of the training data is chosen. Randomness is introduced by assigning the set of histories from a parent node to the two subnodes and selecting the set of splits to undergo the log-likelihood test. After the growing process, each leaf node in a decision tree forms an equivalence class, defined based on the set membership of words in the history. The decision tree-growing procedure is repeated multiple times, and the RFLM probabilities are computed as averages of the individual decision tree probabilities.

$$P_{RF}(w_i|w_{i-n+1},\dots w_{i-1}) = \frac{1}{M}\sum_{j=1}^{M} P_{DT_j}(w_i|\phi_{DT_j}(w_{i-n+1},\dots,w_{i-1})) \qquad (5.48)$$

Here, φ_{DTj} is the *j'th* is a function mapping the history $w_{i-1+n+1},\dots, w_{i-1}$ to a leaf node in the *j*'th decision tree. The number of decision trees *M* usually ranges in the dozens or hundreds. Perplexity and word-error rate tests on the Penn Treebank portion of the Wall Street Journal corpus showed that the perplexity of a random forest trigram was reduced by 10.6% relative to a trigram with interpolated Kneser-Ney smoothing. Interpolation of the RFLM with a Kneser-Ney model did not improve perplexity any further. *N* -best list rescoring with RFLMs yielded a relative word-error rate improvement of 11% on the Wall Street Journal DARPA'93 HUB1 benchmark task. Since their inception, RFLMs have been applied to structured language modeling [71] and prosodic modeling [72]. In the context of multilingual language modeling, they have also been applied to morphologically rich languages (see §5.7.1).

5.6.7. Bayesian Topic-Based Language Models

A significant recent trend in statistical language modeling is Bayesian modeling of latent topic structure in documents. One of the first models of this type was the latent Dirichlet allocation model proposed by Blei, Ng, and Jordan [25]. The LDA model assumes that a document is composed of *K* topics, denoted as $z_1,\dots, z_K$. Each topic generates words according to a topic-specific distribution over individual words (i.e., a topic is modeled as a bag of words; *n*-grams are not taken into account). The probability vector over words is denoted by φ_κ for each topic $k = 1,\dots,K$. Each topic has a prior probability, denoted by θ_k. The topic priors $\theta_1,\theta_2,\dots,\theta_K$ are themselves distributed according to the Dirichlet distribution with hyperparameters $\alpha_1,\dots,\alpha_K$ (see §5.4.2 for an explanation of the Dirichlet distribution):

$$P(\theta_1,\dots,\theta_k) = \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \prod_{k=1}^{K} \theta_k^{\alpha_k - 1} \qquad (5.49)$$

he generative model underlying this approach is that a set of priors $\theta_1,\dots,\theta_K$ is sampled from the Dirichlet distribution. A given topic z_κ is chosen with probability θ_k, then a word *w* is generated from the topic with probability $\varphi_\kappa(w)$. The probability of an entire document consisting of a sequence *W* of *t* words is modeled as

$$p(W|\alpha,\phi) = \int p(\theta|\alpha) \left(\prod_{i=1}^{t} \sum_{z_i} p(z_i|\theta) p(w_i|z_i,\phi) \right) d\theta \qquad (5.50)$$

LDA is a machine learning model that focuses on the posterior distribution of latent variables, which is challenging to exact inference. It requires sampling techniques like Markov chain Monte Carlo or variational inference. LDA is a unigram model, so it needs to be combined with an n-gram model for practical purposes. Combining LDA with a trigram and a probabilistic context-free grammar can reduce perplexity by 9% to 23% on the Wall Street Journal corpus. LDA combined with a hidden Markov model can reduce perplexity by 16.1% and word-error rate by 2.4% over an already adapted trigram model.

LDA has been extended to utilize Dirichlet processes, a prior for nonparametric models that can handle an infinite number of topics. Hierarchical Dirichlet processes (HDP) can be used to structure latent topic variables hierarchically, allowing each topic to have multiple subtopics and individual topics to be shared between different data clusters. HDP-adapted language models have shown slight word-error rate reductions (0.3% absolute) compared to standard adaptation methods. A Bayesian language model based on a Pitman-Yor process achieves perplexities comparable to those of a Kneser-Ney smoothed trigram model without requiring interpolation with the baseline model.

5.6.8. Neural Network Language Models

With the exception of LSA-based language models, all of the language modeling approaches described previously estimate probabilities for events in a discrete space. Neural network language models (NNLMs) [79] take a different approach: discrete word sequences are first mapped into a continuous representation, and *n*-gram probabilities are then estimated within this continuous space. The assumption is that words having similar distributional properties will receive similar continuous representations, which will in turn result in smoother probability estimates.

The neural network is typically a multilayer perceptron with an input layer, a projection layer, a hidden layer, and an output layer of nodes. A graphical representation of the architecture of an NNLM is shown in Figure 5–1. Adjacent layers are fully interconnected by weights. For a vocabulary of V words, the input consists of a concatenation of $n - 1$ V-dimensional binary feature vectors representing the history of $n - 1$ words (e.g., the first two words in a trigram). The projection layer i of a given fixed dimensionality d encodes the shared continuous representation of the words, which is learned during training. The hidden layer h also has a fixed number of J nodes, each of which computes a thresholded, nonlinear combination of incoming activations, for example, using the tangent function:

$$h_j = \tanh\left(\sum_{k=1}^{d} w_{jk}^{h} i_k + b_j^{h}\right) \qquad \forall j, i = 1, \ldots, J \qquad (5.51)$$

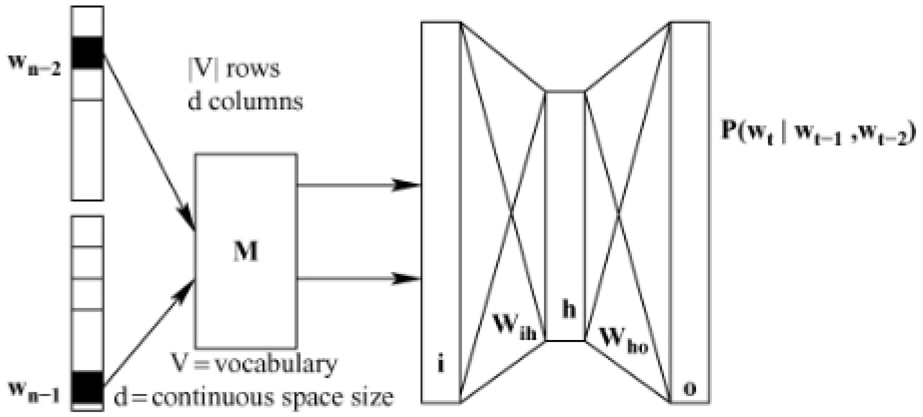


Figure 5–1. Neural network language model

where w^h denotes the weights connecting the projection, and the hidden layers and b^h are the biases on the hidden layer nodes. Finally, the output layer o computes a posterior probability distribution over V by

$$o_j = \frac{a_j}{\sum_{k=1}^{V} a_k} \qquad \forall j, j = 1, \ldots, V \qquad (5.52)$$

and

$$a_j = \sum_{k=1}^{J} w_{jk}^{o} h_k + b_j^{o} \qquad (5.53)$$

During training, binary target labels are associated with the output layer, 1 for the predicted word and 0 for all other words. The network is trained (e.g., using back-propagation) to maximize the log-likelihood of the training data, possibly enriched with a regularization term R to constrain the parameter values θ:

$$L = \frac{1}{T}\sum_{i=1}^{T} log(P(w_i|h_i);\theta) - R(\theta) \qquad (5.54)$$

Regularization terms in neural network learning (NNLMs) are used to limit the complexity of the network and reduce the chance of overfitting. A word in a history is projected into a continuous representation shared among all words, which is then used to estimate the probability of the predicted word given its history. NNLMs have been successfully used for speech recognition, machine translation, and machine translation. They typically use a truncated vocabulary consisting of the m most frequent words, but have yielded significant improvements when combined with standard n-gram models. For example, Schwenk reported an 8% relative reduction in perplexity and a 0.5% absolute improvement in word-error rate on a French broadcast news recognition task, while Emami and Mangu obtained a 0.8% absolute improvement in word-error rate on an Arabic speech recognition task. Emami and Jelinek used neural network-based probability estimation in combination with a structured language model, and Alexandrescu and Kirchhoff combined NNLMs with a factored word representation for Arabic.

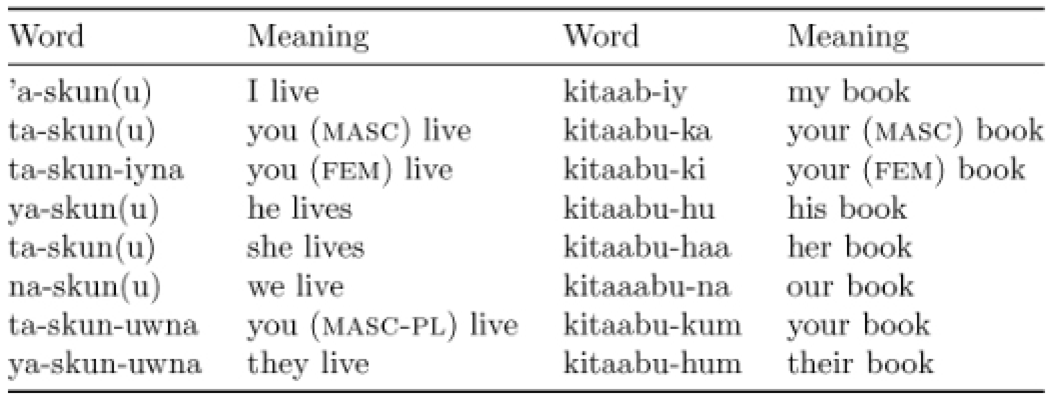
[5.7.1]Language Modeling for Morphologically Rich Languages:

A morphologically rich language is characterized by a large number of different unique word forms (types) in relation to the number of word tokens in a text that is due to productive morphological (word-formation) processes in the language. A morpheme is the smallest meaning-bearing unit in a language. Morphemes can be either free (i.e., they can occur on their own), or they are bound (i.e., they must be combined with some other morpheme). Morphological processes include compounding (forming a new word out of two independently existing free morphemes), derivation (combination of a free morpheme and a bound morpheme to form a new word), and inflection (combination of a free and a bound morpheme to signal a particular grammatical feature).

Germanic languages, for example, are notorious for their high degree of compounding, especially for nominals. Turkish, an agglutinative language, combines several morphemes into a single word; thus, the same material that would be expressed as a syntactic phrase in English can be found as a single whitespace-delimited unit in a Turkish sentence, such as: *görülmemeliydik* = ‘we should not have been seen’.

As a result, Turkish has a huge number of possible words. Many languages have rich inflectional paradigms. In languages like Finnish and Arabic, a root (base form) may have thousands of different morphological realizations. Table 5–1 shows two Modern Standard Arabic (MSA) inflectional paradigms, one for present tense verbal inflections for the root *skn* (basic meaning: ‘live’), one for pronominal possessive inflections for the root *ktb* (basic meaning ‘book’).

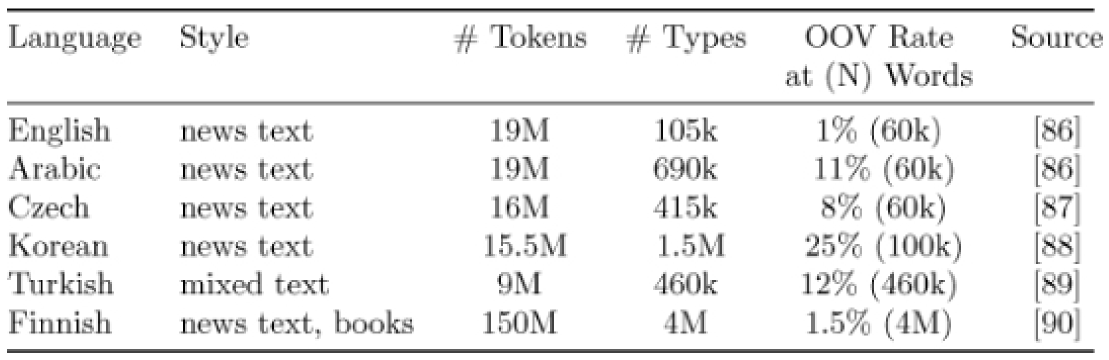
Table 5–1. MSA inflectional paradigms for present tense verb forms and possessive pronouns (affixes are separated from the word stem by hyphens)



Word	Meaning	Word	Meaning
'a-skun(u)	I live	kitaab-iy	my book
ta-skun(u)	you (MASC) live	kitaabu-ka	your (MASC) book
ta-skun-iyna	you (FEM) live	kitaabu-ki	your (FEM) book
ya-skun(u)	he lives	kitaabu-hu	his book
ta-skun(u)	she lives	kitaabu-haa	her book
na-skun(u)	we live	kitaaabu-na	our book
ta-skun-uwna	you (MASC-PL) live	kitaabu-kum	your book
ya-skun-uwna	they live	kitaabu-hum	their book

Morphological complexity in language modeling leads to issues such as a high ratio of types to tokens, resulting in a lack of training data and unreliable probability estimates. This also leads to a high Out of Vocabulary (OOV) rate. While this can be mitigated by collecting more training data, vocabulary growth does not slow down as sharply as for morphologically poor languages. Table 5-2 provides examples of the relationship between word types and tokens for different languages and typical OOV rates on held-out test sets.

Table 5–2. Number of word tokens, word types, and OOV rates for different languages in non-decomposed form



Language	Style	# Tokens	# Types	OOV Rate at (N) Words	Source
English	news text	19M	105k	1% (60k)	[86]
Arabic	news text	19M	690k	11% (60k)	[86]
Czech	news text	16M	415k	8% (60k)	[87]
Korean	news text	15.5M	1.5M	25% (100k)	[88]
Turkish	mixed text	9M	460k	12% (460k)	[89]
Finnish	news text, books	150M	4M	1.5% (4M)	[90]

When processing morphologically rich languages, it is crucial to decide whether to use full word forms or smaller subword units as basic language modeling units. The choice depends on available computing resources, such as speech recognition application efficiency, memory, and desired speed, as well as the amount of training data. Decomposing words into smaller units reduces vocabulary size, reducing the number of distinct n-grams, improving speed and memory consumption. Subword units occur in multiple words, allowing for more robust probability estimation and sharing training data across words. Modeling based on subword units may also allow the language model to assign probabilities to words not seen in the training data. However, linear decomposition of a word reduces predictive power and ensures that language modeling vocabulary is not too short and acoustically confusable when identical to speech recognizer vocabulary.

Arabic, a morphologically rich language, is an example of situations where decomposition may or may not be necessary. Studies have shown that integrating morphological information into language models is helpful for modeling dialectal Arabic. However, Arabic dialects have sparse training data due to being essentially spoken languages and needing manual transcription. Large amounts of data are available for MSA, but significant improvements through morphological decomposition have not been observed for this variety when large training sets are used. The vocabulary size needed for large-scale applications like speech recognition of MSA is around 600,000 to 800,000 words, which is within the range that current decoders can accommodate. However, for languages with high type to token ratios (e.g., Finnish or Turkish), some form of decomposition is required, as the vocabulary size may exceed current decoder capabilities and the corresponding language model would not be able to be estimated reliably.

5.7.2. Selection of Subword Units:

Subword unit identification can be done using data-driven, unsupervised methods, linguistic information, or a combination of both. Linguistically based methods use handcrafted morphological analysis tools like the Buckwalter Morphological Analyzer for Arabic. Statistical disambiguation is performed in cases where multiple analyses are provided. Data-driven approaches may incorporate varying levels of information about the language and optimization criteria. Some aim to discover units corresponding to linguistically defined morphemes, while others select an inventory of units best suited to the task or application.

Automatic algorithms for identifying linguistic morphemes have been around since Zellig Harris's 1955 approach. A modified version of this approach can be used to decompose German compounds, reducing the Out of Vocabulary (OOV) rate by 23% to 50% in a German corpus of 300 million words and a fixed vocabulary of 65,000 to 100,000. However, simple frequency-based approaches are prone to overfitting due to the unbalanced fit to the training data. To address this, models that include explicit penalty terms for the size of the morpheme inventory are used. The Morfessor package is an example of such a model, which derives a morpheme inventory from a corpus by maximizing its posterior probability.

$$M = \operatorname*{argmax}_{M} P(M|C) = P(C|M)P(M) \tag{5.55}$$

Morfessor is a method used for automatic segmentation of words, which involves a greedy algorithm that splits each word into two subparts to reduce code length. This approach has been applied to Finnish, Turkish, and English speech recognition, achieving good results on the first three languages. Morfessor models have been used in Finnish, Estonian, Turkish, and Arabic speech recognition. Another approach is to derive a unit inventory that directly optimizes a criterion related to language model performance, such as perplexity or OOV rate. This approach is preferred in cases where languages are not strictly agglutinative but contain some fusion, such as nontransparent changes in word form produced by the combination of two or more morphemes. For example, a particle-based model was developed for Russian language modeling.

$$P(w_i|h) = \frac{1}{Z(h)} P\left(u^{w_i}_{L(w_i)} | u^{w_i}_{L(w_i)-1}\right) P\left(u^{w_i}_{L(w_i)-1} | u^{w_i}_{L(w_i)-2}\right), \ldots, P(u^{w_i}_1) \left| u^{w_{i-1}}_{L(w_{i-1})}\right) \quad (5.56)$$

The particle language model is a method that decomposes words into particles, which are then computed based on their history. Two data-driven methods for deriving particles were compared: a greedy search over possible units of a fixed length and a particle-growing technique. Another approach was proposed by Kiecza, Schultz, and Waibel, which combined elementary syllable units into eojols, larger than syllables but smaller than Korean words, similar to Turkish words. Both approaches improved perplexity/OOV rate but only limited gains in speech recognition word-error rate. Recently, the choice of subword units has been optimized for final system performance, usually by trying different segmentation schemes and evaluating each's effect on performance metric. Arisoy, Sak, and Saraçlar's study compared words, statistically derived units, linguistically defined morphemes, and stems plus endings for Turkish language modeling for speech recognition, finding stems plus endings to yield the best word-error rate out of all four choices.

5.7.3. Modeling with Morphological Categories

The majority of language modeling using subword units focuses on linear decomposition of words, mainly for agglutinative languages. These subword units are commonly used in standard n-gram models. However, the n-gram context needs to be increased to model inter-word and sub-word dependencies, which increases the amount of training data needed.

Alternative approaches to language modeling have been developed, incorporating statistics over subword components or morphological classes. Arisoy et al. proposed a discriminative language model for Turkish that uses feature functions defined over morphemes, such as root n-gram counts or inflectional group counts. This model assigns probabilities to sequences of entire words, taking into account the constraints provided by morpheme-based feature functions. This approach showed slightly better results (0.3% absolute word-error rate reduction) than a discriminative language model using only word-based features.

Kirchhoff et al. and Vergyri et al. developed a factored language model for Arabic using morphological classes and words as conditioning variables. The model predicts probabilities for entire word forms, but uses morphological constituents. FLMs showed slight reductions in word error rate on a dialectal Arabic speech recognition task with limited training data. FLMs have been successfully used for speech recognition in other highly inflected languages.

Morphological features were also used as additional input features (beyond words) to a neural language model (see §5.6.9) for both Arabic and Turkish [85], thus creating a factored neural language model. Perplexity was shown to improve substantially over a wordbased neural language model (up to 10% for Arabic and 40% for Turkish), but no application results were reported.

Sarikaya and Deng [105] proposed a joint morphological-lexical language model for Arabic. Here, a sentence is annotated with a rich parse tree representing morphological, syntactic, semantic, and other attribute information. The language model predicts the probability of the word string and the associated tree jointly, using MaxEnt probability estimation (see §5.6.5). The model was evaluated on an English-to-Arabic translation task and improved the BLEU score by 0.3 points (absolute) over a word trigram model and 0.6 points over a morpheme trigram model.

RFLMs (see §5.6.7) were applied to morphological language modeling by Oparin et al. [106]. The difference from standard word-based RFLMs is that the decision trees used in the random forest model can ask questions not only about the set membership of words but also about morphological features (inflections or morphological tags), stems, lemmas, and parts of speech. Morphological RFLMs were evaluated on a Czech spoken lecture recognition task with a 240,000-word vocabulary. Whereas word-based RFLMs did not show any substantial gain over Kneser-Ney-style trigram language models, morphological RFLMs yielded a relative gain in perplexity of up to 10.4% and a relative improvement in word accuracy of up to 3.4%. Unlike previous studies' results, it was also found that interpolation of RFLMs and standard *n*-gram models yielded an improvement (of up to 15.6% in perplexity). In addition to producing different word history clusters (induced by morphological features rather than words), a morphological RFLM offers more possibilities for randomization because the set of potential splits at each decision tree node is greatly enlarged, which may contribute to the superior performance of morphological RFLMs in this case.

5.7.4. Languages without Word Segmentation

Although agglutination produces a large number of long and complex word forms in many languages, other languages do not possess any explicit segmentation of character strings into words at all. In languages such as Chinese and Japanese, sentences are written as sequences of characters delimited by punctuation signs but without any intervening whitespaces to indicate word boundaries. Readers with knowledge of the language can immediately decompose character sequences into the word segmentation that corresponds to the most plausible interpretation of the sentence. Although statistical language models for such languages can be constructed over characters, it is preferable to first segment sentences into words automatically and then train the language model over the resulting words. Similar to the situation in which words are decomposed into subword units (§5.7.1), a language model using characters as the basic modeling units may fail to express important interword relationships. Moreover, the word segmentation may determine how characters are pronounced, which is important if the same modeling units are intended to be used in the language model and the acoustic model of a speech recognition system. Finally, it has been shown experimentally that a language model built on automatically segmented text outperforms a character-based language model in terms of perplexity for both Japanese [107] and Chinese [108].

Automatic segmentation algorithms typically use a combination of dictionary information, statistical search, and additional features such as nonnative letters, character co-occurrence counts, and character positions. Generating the most likely segmentation follows a statistical decoding framework that uses, for example, Viterbi search. Other modeling approaches that have been explored recently include conditional random fields [109, 110, 111], MaxEnt modeling [112, 113], and discriminative modeling using perceptrons [114]. Much of the work on Chinese word segmentation has been performed in the context of the SIGHAN Chinese word segmentation competitions that have been taking place since 2003, organized by the Association for Computational Linguistics. This competition serves as a benchmark comparing different word segmentation systems. Automatic word segmentations are compared against a linguistically segmented gold standard and are evaluated in terms of precision (P), that is, the percentage of correct cases out of all hypothesized boundaries, recall R (the percentage of identified boundaries out of all possible boundaries), and their combination in the form of F-measure, $F = 2PR/(P + R)$. These measures can be calculated separately on the OOV words versus in-vocabulary words. The best-performing systems in the most recent evaluation achieved an F-measure of 0.96; however, performance on OOV words was much lower with F-measures around 0.76 [115].

Rather than trying to match linguistically defined words, it is also possible to optimize segmentation directly for language model performance. Sproat et al. [108] showed that the dictionary used for Chinese word segmentation significantly influences the perplexity of a bigram language model trained on the segmented text and that it is possible to iteratively optimize the dictionary by merging frequent word co-occurrences, such that the perplexity is reduced at each iteration. Note that such approaches are similar to the data-driven algorithms for deriving morpheme-like subword units described earlier in Section 5.7.1. Another example of a data-driven approach, in this case for Japanese, uses chunks of characters for language modeling [107]. Chunks are derived from the training data by selecting the highest-frequency *n*-grams and additional patterns with close similarity to them. Thus, the basic modeling units are neither characters nor words but intermediate units.

5.7.5. Spoken versus Written Languages

Statistical language modeling crucially relies on large amounts of written text data, and a significant trend within language modeling research is the development of methods for scaling current language modeling techniques to ever-larger databases. However, many of the world’s 6,900 languages are spoken languages, that is, languages without a writing system. They are either indigenous languages without a literary tradition, or they are linguistic varieties such as regional dialects that are used in everyday spoken communication but are rarely put into writing.

This is the case, for example, with the many dialects of Arabic, which are used in everyday conversation but are almost never found in written form. Other languages may be spoken as well as written but may not have a standardized orthography.

Both cases represent difficulties for language modeling. In the first case, the only way of obtaining language model training data is to manually transcribe the language or dialect. This is a costly and time-consuming process because it involves (i) the development of a writing standard, (ii) training native speakers to use the writing system consistently and accurately, and (iii) the actual transcription effort. In the second case, those text resources that can be obtained for the language in question (e.g., from the web) will need to be normalized, which can also be a laborious process. As a consequence, very little work has been carried out on language model development for such underresourced languages. Most studies have concentrated on how to rapidly collect corpora for underresourced languages using web resources. Some of the challenges inherent in this process are described by Le et al. [116] and Ghani, Jones, and Mladenic [117]. Possible methods for rapid language model bootstrapping for spoken languages and languages characterized by a lack of standardization might include grammar-based or class-based approaches in combination with a limited amount of transcribed material. For a constrained application, such as the development of a dialog system, the structure of possible utterances could be predefined by a task grammar or a class-based language model, whereas more fine-grained word sequence probabilities, or probabilities of words given their classes, are modeled by a language model trained from a small amount of data. An interesting research direction is the possible use of data from language that is closely related to the language in question or from a language that is unrelated but resource rich. The following section describes some of these approaches.

[5.8] Multilingual and Crosslingual Language Modeling:

5.8.1. Multilingual Language Modeling

Up to this point we have discussed problems that arise when tailoring statistical language models to a particular language or language type, such as agglutinative languages or languages without word segmentation. The tacit assumption has been that the resulting language model is used in an application where only the language of interest is encountered. However, in many situations, a system can be presented with multiple languages sequentially (e.g., different users speaking different languages, without advance indication of which language will be encountered next), or simultaneously, as happens in the case of **code switching**. Here, speakers may use several languages or dialects side by side, often within the same utterance. The phenomenon of code switching exists in a variety of bilingual and multilingual communities or where diglossia exists, such as where a formal standard language is used in addition to colloquial or dialectal varieties. The use of "Spanglish" (mixed Spanish/English) in the United States is one example of code switching, as demonstrated by the following example from Franco and Solorio [118]:

I need to tell her que no voy a poder ir.
'I need to tell her that I won't be able to make it.'

To handle multilingual input where language can be switched dynamically between utterances, separate language models can be constructed from monolingual corpora, and a system using these models (e.g., a speech-based information kiosk or telephone-based dialog system) can access them dynamically based on the output from a first-step language identification module or based on which language model (possibly combined with an acoustic model in the case of speech recognition) yields the highest scores after the first processing steps.

Fügen et al. showed how several such monolingual models can be combined into a single multilingual language model by means of a context-free grammar whose nonterminal states can encode language information and whose terminal states correspond to monolingual *n*-gram models. Explicit grammar rules can be leveraged to expand current states with *n*-grams from the matching language only, thus preventing language switching at inappropriate times. Alternative ways of constructing a single multilingual language model are to combine monolingual corpora and train a single model on the pooled data or to interpolate several monolingual language models. The first technique has been shown to degrade performance, especially when the sizes of the corpora are unbalanced [120, 121]. The second technique may fare slightly better yet was shown to be inferior to the grammar-based combination method described earlier [119].

Building a language model for the second case (intrasentential language switching) is more difficult because little or no relevant training material is available. Weng et al. [122] built a four-lingual language model by introducing a common backoff node in the form of a pause unit; that is, language switches are allowed with some probability after a pause has occurred.

5.8.2. Crosslingual Language Modeling

Another question that might be asked is whether data in one language can be leveraged to improve a language model for a different language, provided that the style or domain are closely related. If insufficient data is available for the language of interest, inaccurate probability estimation might be improved if sufficient information can be extracted from a large amount of foreign-language text.

The most straightforward way of applying this idea is to automatically translate the foreign-language text into the desired language and to use the translated text (though errorful) as additional language training data. This approach was chosen, for example, by Khudanpur and Kim [123] and Jensson et al. [124].

In the former study, training data for a Mandarin news text language model intended for speech recognition was enriched with training data obtained by automatically translating English text from the same domain. A unigram extracted from the translated text was interpolated with a trigram baseline language model trained on the available Mandarin data. The selection of English text for translation, and the determination of the interpolation parameter λ were specific to each news story, thus providing at the same time an implicit form of topic adaptation. The resulting model improved character perplexity by about 10% relative and the word-error rate of the speech recognizer by 0.5% absolute (for various different systems ranging around 26% baseline character-error rate). The authors also noted that the English text was more recent than the Chinese text, which may have contributed to the improved performance. This is an important consideration when investigating potential out-of-language data resources.

Jensson et al. [124] developed a language model for weather reports in Icelandic using a small amount of in-language data and a limited amount of parallel English–Icelandic data for training a machine translation system. A language model trained on a larger set of automatically translated data was then interpolated with the baseline language model, with positive effects on perplexity (9.20% improvement) and word-error rate (1.9–9.5% relative improvement) of an Icelandic speech recognition system.

The use of machine translation technology for processing out-of-language data is likely to fail when the desired language does not have sufficient data to train a machine translation system in the first place, although the abovementioned experiments on Icelandic show that a limited amount of parallel data may still be sufficient if the domain is very constrained. An alternative to using a fully fledged machine translation system is to rely on a high-quality word-based translation dictionary only. Kim and Khudanpur [125] showed that a goodquality translation dictionary can be extracted by simply computing mutual information statistics on word pairs from document-aligned parallel corpora, eliminating the need for sentence-aligned parallel text. In their experiments on Mandarin broadcast news recognition, they found that a unigram model built from the resulting dictionary-based translations and interpolated with a baseline language model achieved a performance similar to that of a crosslingual language model. Another possibility for constructing a translation dictionary from document-aligned data is to use crosslingual latent semantic analysis [126]. Under this approach, words in both languages are projected into a common semantic space, and a measure of similarity between words from different languages in this space is used to construct word translation probabilities.

A drawback of the previous approaches is that the quality of the resulting model is heavily dependent on the translation accuracy. Tam et al. [127] recently presented an alternative model in which bilingual latent semantic analysis (bLSA) is used for adaptation before translation. The approach uses one LSA model each for the source and the target languages; it thus requires a parallel training corpus. The LSA models incorporate Dirichlet-style prior distributions over topics (see §5.6.8). Topic mixture weights are determined from the source LSA model and are projected onto the target LSA model, where they can be used to compute target-language marginal distributions. Assuming that the source language is Chinese (Ch) and the target language is English (En), the marginal word probability distribution in English is

$$P_{\text{En}}(w) = \sum_k \phi_k^{\text{En}}(w)\theta_k^{\text{Ch}} \qquad (5.57)$$

where θ_k is the prior for the k^{th} topic and $\varphi_k(w)$ is the probability assigned to word w by the k^{th} latent topic. As we can see, topic priors are determined by the source language, whereas the topic-dependent word probability distribution is determined by the target language. The target-language marginals are then incorporated into the target-language model as follows:

$$P_{\text{target}}(w|h) \propto \left(\frac{P_{\text{bLSA}}(w)}{P_{\text{base}}(w)}\right)^{\beta} P_{\text{base}}(w|h) \qquad (5.58)$$

where P_{bLSA} is the adapted probability and P_{base} is the baseline probability. This approach enforces a one-to-one topic correspondence across languages. Evaluation on a Chinese– English statistical machine translation task in the news domain showed that the bLSAadapted language model reduced perplexity by 9% to 13% and improved the BLEU score by up to 0.3 points.

SUMMARY:

Statistical language modeling has evolved significantly in recent years, with new models like neural network and discriminative language models being used alongside traditional n-gram models. Language modeling techniques, such as language model adaptation, are applicable to various languages and are language independent. However, there are differences in cases with rich morphology, particularly in highly agglutinative languages. Word decomposition or integration of statistics over subword components into discriminative, factored, or neural language models can help.

A recent trend is the use of large-scale, distributed language models, which use approximate scores or counts instead of traditional probability estimation schemes. This trend is particularly significant for languages with large vocabularies and facilitates the use of large language models in practical systems.

Limited research has been conducted on languages that are resource-poor, meaning they do not have a large amount of text data. Speech recognition technology combined with bootstrapping techniques could automatically transcribe more data incrementally. However, the development of accurate speech recognition systems requires sufficient text and acoustic data to train initial models.

Future research should focus on crosslingual adaptation techniques for language models, which would significantly contribute to applying human language technology to resource-poor languages.