Speech Recognition and Understanding

Notes on Speech and Audio Processing

Chia-Ping Chen

Department of Computer Science and Engineering
National Sun Yat-Sen University
Kaohsiung, Taiwan ROC

Introduction

- We have described ASR as a pattern recognition problem requiring signal processing, probability estimation and temporal integration.
- To have the probability of a hypothesized sequence of words in a speech, we need to study the so-called language models.
- We will show to these aspects are integrated in the decoding process for recognition.
- In addition, we discuss a speech-understanding system based on speech recognition and further language processing.

Word Pronunciations

- An ASR system needs to know the pronunciation(s) of each word. Specifically, these pronunciations are expressed as the modeling units such as phones.
- A simple way to this is use a lexicon which lists of the pronunciation for every word in the vocabulary.
- For a refined system, one may want to learn the pronunciation from data. The results are often stored in the form of pronunciation models (Figure 28.2) or phonological rules (Table 28.1).

Pronunciation Models

- A pronunciation model can be obtained by
  - Run forced-alignments on training data for the known models.
  - Count the occurrences of each pronunciation and normalize to get pronunciation probabilities.
  - Repeat until convergence is achieved.
- This procedure can be generalized to learn the variations in pronunciations for larger linguistic units.

Language Models

- The Bayes rule requires the model probability in addition to data likelihood. In ASR, that is the probability of a sentence.
- How do we assign such probability? Among the challenges, note
  - There are infinitely many sentences.
  - The probabilities of these sentences sum to 1.

Entropy and Perplexity

- A stationary stochastic process has a measure for its degree of uncertainty called the entropy rate. Let the true probability be p, then

$$H = \lim_{n\to\infty} \frac{-1}{n} \log p(w_1, \ldots, w_n).$$

- If the probability is estimated to be q, then

$$\begin{aligned}\lim_{n\to\infty} \frac{-1}{n} \log q(w_1, \ldots, w_n) &= \lim_{n\to\infty} \frac{-1}{n} \log p(w_1, \ldots, w_n) + \lim_{n\to\infty} \frac{1}{n} \log \frac{p(w_1, \ldots, w_n)}{q(w_1, \ldots, w_n)} \\ &\geq \lim_{n\to\infty} \frac{-1}{n} \log p(w_1, \ldots, w_n) = H.\end{aligned}$$

The left-hand side is called the cross entropy. It is an upper bound for H.

Cross Entropy and Perplexity

- The perplexity is the exponential of the entropy. It is the average number of candidates for next word,

$$p(w_1, \ldots, w_n) \sim 2^{-nH} = (\frac{1}{2^H})^n.$$

- If we use the cross entropy from estimated q, then

$$q(w_1, \ldots, w_n) = (\frac{1}{\mathrm{PPL}})^n, \text{ or } \mathrm{PPL} = q(w_1, \ldots, w_n)^{\frac{-1}{n}}.$$

It is an upper bound for the true perplexity.

n-Gram Models

- n-gram LM is an $(n-1)$th order Markov model. I.e.

$$p(w_i|w_1^{i-1}) = p(w_i|w_{i-n+1}^{i-1}).$$

- Without any approximation, the probability of a sentence can be written by

$$p(w_1^N) = p(w_1| < s >) \prod_{i=2}^{N} p(w_i|w_1^{i-1})\, p(< /s > |w_1^N).$$

With n-gram, this becomes

$$p(w_1^N) = p(w_1| < s >) \prod_{i=2}^{N} p(w_i|w_{i-n+1}^{i-1})\, p(< /s > |w_{N-n+2}^N).$$

Issues on Language Models

- The long-range word dependency is not directly modeled in n-gram unless n is large.
- Syntactic and semantic rules are not explicitly implemented.
- The data sparsity problem: in n-gram, there are V^n parameters to be learned from data. This is a huge number when $n \geq 4$. Normally we use $n = 3$. Even in this case there are many unseen trigrams and we need smoothing schemes such as discounting, backoff and deleted interpolation.

Smoothing

- The maximum likelihood estimate of n-gram is

$$p(w_i|w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^{i})}{c(w_{i-n+1}^{i-1})}$$

- Some n-grams may not appear in the corpus used for counting, resulting in 0 probability.
- There are so many n-grams (say $n = 3$) that a corpus can not cover all of them. Therefore we need to deal with the 0-occurrence problem.

Add-One Smoothing

- The count of each n-gram is increased by 1. The total count is increased by V, so

$$p_i^* = \frac{c_i + 1}{N + V}.$$

- Equivalent to modify the counts of the ith n-gram by

$$c_i^* = (c_i + 1)\frac{N}{N + V} \quad (\text{and } p_i^* = \frac{c_i^*}{N})$$

- Every occurrence of the ith n-gram is discounted to

$$d_i = \frac{c_i^*}{c_i}.$$

Backoff

- If we have no occurrence for an n-gram, we use the count of the occurrences of $(n-1)$-gram. For trigram,

$$\hat{p}(w_i|w_{i-1}w_{i-2}) = \begin{cases} \tilde{p}(w_i|w_{i-1}w_{i-2}), & c(w_{i-2}w_{i-1}w_i) > \\ \alpha(w_{i-1}w_{i-2})\hat{p}(w_i|w_{i-1}), & \text{otherwise} \end{cases}$$

- The α's are used to make the total probability 1,

$$\alpha(w_{i-1}w_{i-2}) = \frac{1 - \sum_{w \in A} \tilde{p}(w|w_{i-1}w_{i-2})}{1 - \sum_{w \in A} \hat{p}(w|w_{i-1})},$$

where $A = \{w|c(w_{i-2}w_{i-1}w) > 0\}$.

Deleted Interpolation

- The basic idea is

$$\hat{p}(w_i|w_{i-1}w_{i-2}) = \lambda_1 p(w_i|w_{i-1}w_{i-2}) + \lambda_2 p(w_i|w_{i-1}) + \lambda_3 p(w_i),$$

with $\sum_i \lambda_i = 1$.

- The λ values can be made dependent on the context w_{i-1}, w_{i-2}.

Decoding

- With HMM and bigram ($n = 2$) language model, a time-synchronous Viterbi search algorithm can be used.
- For higher order language models or more refined acoustical models, one can
  - use depth-first search methods.
  - use multiple-pass decoding. The first pass generates good candidates with fast and simple model. Later passes re-score these hypotheses with more refined models.

BERP

- A speaker-independent spontaneous speech understanding system
  - A user call the system to inquire about restaurants around Berkeley.
  - The system knows something about the world with a knowledge base.
  - The query is mapped against the knowledge base.
- The user's speech is decoded by a Viterbi decoder, and the recognition result is processed by a parser using stochastic context-free grammar (SCFG) so the fields in a database query can be filled in.

Accepting Real Inputs

- Some recognition results have very low scores, for which the recognizer is likely to be wrong.
- In speech understanding we also want to reject irrelevant words.
- A confidence measure (and threshold) for the recognition-understanding output is useful.
- For spontaneous speech, we need to handle disfluency, non-speech sounds, and speech fragments.

Evaluations

- Word error rates is the standard metric for evaluation. It is defined by

$$\mathrm{WER} = \frac{I + S + D}{N} * 100\%,$$

  - I is the number of insertions,
  - D is the number of deletions,
  - S is the number of substitutions,
  - N is the number of words in the reference.
- These numbers are based on the optimal alignment between the output and the reference transcription.