2003 정보 및 제어 학술회의 (2003.11.21-22)

문장음성인식을 위한 VCCV 기반의 효율적인 언어모델

Efficient Language Model based on VCCV unit for Sentence Speech Recognition

박 선 희, 노 용 완, 홍광석

성균관대학교 정보통신공학부(전화:(031)290-7196, 팩스:(031)290-7191, E-mail : seonhp@dreamwiz.com)

Abstract : In this paper, we implement a language model by a bigram and evaluate proper smoothing technique for unit of low perplexity. Word, morpheme, clause units are widely used as a language processing unit of the language model. We propose VCCV units which have more small vocabulary than morpheme and clauses units. We compare the VCCV units with the clause and the morpheme units using the perplexity. The most common metric for evaluating a language model is the probability that the model assigns the derivative measures of perplexity. Smoothing used to estimate probabilities when there are insufficient data to estimate probabilities accurately. In this paper, we constructed the N-grams of the VCCV units with low perplexity and tested the language model using Katz, Witten-Bell, absolute, modified Kneser-Ney smoothing and so on. In the experiment results, the modified Kneser-Ney smoothing is tested proper smoothing technique for VCCV units.

Keywords : Languge Model, Smoothing, VCCV, Perplexity

Ⅰ. 서론

언어 모델의 역할은 음성의 모호함 때문에 음향학적으로 구별하지 못하는 부분을 언어 정보를 이용하여 탐색 공간을 줄이는데 있다. 일반적으로 음성 인식의 성능은 이러한 모델들이 어떻게 결합되어지느냐에 따라 좌우된다. 음성인식에 사용되는 언어모델로는 문법기반의 언어모델과 통계적 언어모델이 있으나, 문법기반의 언어 모델은 단어수가 증가하면 네트워크의 상태수가 급격히 증가하여 탐색시간이 오래 걸리는 단점이 있어 일반적으로 검색 시간이 적게 걸리는 통계적 언어모델이 많이 사용되고 있다. 대표적 통계적 언어 모델로는 N-gram을 들 수 있으며, 충분한 학습 데이터가 존재할 경우에 매우 좋은 성능을 보여준다[1].

N-gram 언어 모델의 확률값 예측에 중요한 역할을 하는 것은 smoothing이다. smoothing을 통해 불확실한 데이타에 대해 좀 더 정확한 확률값 추정이 가능해진다. smoothing 방법으로는 back off 와 interpolation의 두가지 형태가 있고 각 방식에 적합한 discounting 방법이 있다[2].

언어모델의 언어처리 단위로 한국어에 주로 어절과 형태소를 사용한다. 어절은 한국어의 띄어쓰기 기본 단위로 발성의 지속시간이 길기 때문에 N-gram에서 어절 단위로 인식하는 경우 많은 양의 학습 데이터가 존재해야 좋은 성능을 나타낼 수 있다. 형태소를 사용하게 되면 단음소나 단음절을 가진 형태소가 많아서 인식 오류가 증가하게 되므로 인식 성능 개선을 위해 적절한 길이의 발성시간과 적절한 수의 사전 표제어를 가질 수 있게 하는 형태소 결합이 필요하게 된다[3].

본 논문에서는 어절이나 형태소단위보다 모델 구성에 있어서 학습 데이터를 많이 필요로 하지 않고 코퍼스의 크기가 줄어 복잡도가 적어 인식 성능에 좋은 영향을 미치는 VCCV 단위를 언어 처리 단위로 제안한다.

본 논문에서는 VCCV 단위가 한국어 언어모델의 언어처리 단위로 적합한지 알아보기 위해 기존의 언어처리 단위인 어절과 형태소 그리고 VCCV 단위로 언어 모델을 구성하여 어휘수와 복잡도를 비교하였다. 또한 VCCV 단위에 적합한 smoothing 기법이 무엇인지 평가하기 위하여 back off나 interpolation 형태를 가지는 Katz, absolute, modified Knesar-Ney 등의 smoothing을 사용한 VCCV 단위의 언어 모델을 구성하여 성능 평가를 수행하였다.

Ⅱ. 언어 모델

언어 모델은 문장 음성을 인식할 때 탐색해야 할 단어의 수를 줄임으로써 문장 구성을 위한 탐색 시간과 인식률을 높이는 역할을 한다. 음성 인식에 주로 사용되는 언어 모델로는 구 구조 문법에 기반한 모델과 통계적 언어 모델을 들 수 있다. 구 구조 문법의 경우 정규 문법이나 문맥 자유 문법을 사용하여 문장의 탐색 시간과 동시에 parsing을 수행하여 문법의 구조에 어긋난 탐색 공간을 제어하게 된다. 단어수가 늘어나면 네트워크의 상태수가 급격히 증가하게 되어 탐색 시간이 오래 걸리는 단점이 있다. 이해 비해 통계적 언어 모델은 보통 주어진 영역이 많은 텍스트 문장으로부터 쉽게 추출이 가능하고 문장의 발생 확률만을 계산하므로 학습 문장과 부분적으로 다른 문장도 인식 할수 있는 장점이 있다[1].

1. 통계적 언어모델

통계적 언어 모델은 음성인식에서 정확성 뿐만 아니라 탐색 공간을 급격히 줄이는 효과를 보여준다. 통계적 언어 모델의 목적은 주어진 인식 영역에 맞는 단어열 W의 확률을 예측하는 것이다[4].

단어열 W는 $w_1, w_2, \cdots, w_3$로 이루어진 단어열이라고 가정하면 단어열 W의 확률 P(W)는 식(1)과 같다.

$$p(w_1, w_2, \cdots, w_Q) = p(w_1)p(w_2|w_1)\cdots p(w_Q|w_1\cdots w_{Q-1}) \quad (1)$$

그러나 이 식은 주어진 언어에서 모든 가능한 단어에 대한 P(W)를 계산하기는 용이 하지 않다. 따라서, 마코브가정에 의해 P(W)를 근사하는 N-gram을 사용한다. N-gram 방식은 앞의 N-1개의 단어를 바탕으로 현재의 단어에 대한 확률을 계산하는 방법으로 식(2)와 같다.

$$P(w_j|w_1 w_2\cdots w_{j-1}) \approx P(w_j|w_{j-N+1}\cdots w_{j-1}) \quad (2)$$

언어모델 P(W)는 보통 정해진 영역의 많은 텍스트 코퍼스로부터 추출한다. 텍스트 코퍼스로부터 P(W)를 추정하는 방법은 다음과 같다. 실제적인 관점에서 P(W)는 식(3)과 같이 근사적으로 표시된다.

$$P_N(W) = \prod_{j=0}^{Q} P(w_j|w_{j-1}, w_{j-2}, \cdots, w_{j-N+1}) \quad (3)$$

이것을 N-gram laguage model이라고 한다[3].

2. Smoothing

언어모델에서 나올 수 있는 경우가 드문 어휘에 대한 확률의 올바른 추정을 위해서는 smoothing 기술이 필수적이다. Maximum likelihood에 의해서 언어 모델의 확률값을 구하게 되면 zero-count 단어들에 대해서는 zero의 확률값이 주어지게 되고 nonzero-count단어들에 대해서는 너무 높은 확률값이 주어진다. 이러한 값들에 대해서 높은 확률값은 discount해주고 zero 확률값에 대해서도 어느 정도의 확률을 부여해주는 smoothing을 해주어야 한다.

Smoothing하는 형식에 따라 back off 방법과 interpolation 방법이 있다[2].

Absolute smoothing은 back off 형태이고 absolute discounting을 사용한다. Bigram 형태는 식 (4)와 같다.

$$P_{abs}(w_j|w_{j-1}) = \begin{cases} \dfrac{C(w_j w_{j-1}) - D}{C(w_j)} & \text{if} \quad C(w_{j-1}w_j) > 0 \\ D\dfrac{N - n_0(w_{j-1})}{C(w_{j-1})} P(w_j) & otherwise \end{cases} \quad (4)$$

여기서 D는 $n_1/n_1 + 2n_2$이고 n_r은 r번 나타난 bigram의 수이다[3].

Katz smoothing은 back off 형태이고 Good-Turing discount을 사용한다. Good-Turing 방식은 bigram이 r번 발생하는 상태는 추정하는 것으로 r^*로 나타내고 r^*는 식 (5)와 같다.

$$r^* = (r+1)\frac{n_{r+1}}{n_r} \quad (5)$$

Katz smoothing의 bigram 형태는 다음 식 (6)과 같고 d_r값은 식 (7)과 같다.

$$P_{katz}(w_j|w_{j-1}) = \begin{cases} C(w_{j-1}w_j)/C(w_{j-1}) & \text{if} \quad r > k \\ d_r C(w_{j-1}w_j)/C(w_{j-1}) & \text{if} \quad k \geq r > 0 \\ \alpha(w_{j-1})P(w_j) & \text{if} \quad r = 0 \end{cases} \quad (6)$$

$$d_r = \begin{cases} 1 & r > k \\ \dfrac{\dfrac{r^*}{r} - \dfrac{(k+1)n_{k+1}}{n_1}}{1 - \dfrac{(k+1)n_{k+1}}{n_1}} & r < k \end{cases} \quad (7)$$

$\alpha(w_{j-1})$은 확률값의 합이 1이 되어야 하기 때문에 식 (8)로 계산된다[3].

$$\alpha(w_{j-1}) = \frac{1 - \Sigma_{w_j: r>0} P_{katz}(w_j|w_{j-1})}{1 - \Sigma_{w_j: r>0} P(w_j)} \quad (8)$$

Kneser-Ney smoothing은 back off 형태이고 absolute discount을 사용한다. bigram 형태는 식 (9)와 같고 $P_{KN}(w_j)$와 $\alpha(w_{j-1})$은 식 (10)과 같다[2].

$$P_{KN}(w_j|w_{j-1}) = \begin{cases} \dfrac{C(w_j w_{j-1}) - D}{C(w_j)} & \text{if} \quad C(w_{j-1}w_j) > 0 \\ \alpha(w_{j-1})P_{KN}(w_j) & otherwise \end{cases}$$

$$P_{KN}(w_j) = \frac{C(\cdot\, w_j)}{\Sigma_{w_j} C(\cdot\, w_j)} \quad (9)$$

$$\alpha(w_{j-1}) = \frac{1 - \Sigma_{w_j: C(w_{j-1}w_j)>0} \dfrac{C(w_{j-1}w_j) - D}{C(w_{j-1})}}{1 - \Sigma_{w_j: C(w_{j-1}w_j)>0} P_{KN}(w_i)} \quad (10)$$

(식 (10)에서 $C(\cdot\, w_j)$는 w_j에 선행하는 unique한 단어의 수이다.)

Witten-Bell smoothing은 interpolated 형태이고 linear discount을 사용한다. 이 기법은 bigram의 count

수에 의존하지 않고 t에 따라서 discounting된다. 이 smoothing 기법의 bigram 형태는 식 (11)과 같다.

$$P_{WB}(w_j|w_{j-1}) = \frac{C(w_{j-1})}{C(w_{j-1})+t}\frac{C(w_{j-1}w_j)}{C(w_{j-1})} + \frac{t}{C(w_{j-1})+t}P(w_j) \quad (11)$$

여기서 t는 특정 문맥 다음에 올수 있는 다른 event의 수이다[3].

Modified Kneser-Ney smoothing 기법은 Knerser-Ney의 변형된 형태로 back off 대신에 interpolated 형태를 사용하고 하나의 absolute discount가 아닌 3개의 absolute discount을 사용한다. 식은 다음 식 (12),(13)과 같다[2].

$$P_{M-KN}(w_j|w_{j-1}) = \frac{C(w_{j-1}w_j) - D(C(w_{j-1}w_j))}{C(w_{j-1})} + \gamma(w_{j-1})P(w_j)$$

$$D(c) = \begin{cases} 0 & \text{if } c=0 \\ D_1 & \text{if } c=1 \\ D_2 & \text{if } c=2 \\ D_{3+} & \text{if } c\geq 3 \end{cases} \quad (12)$$

$$\gamma(w_{j-1}) = \frac{D_1 n_1(w_{j-1}\cdot) + D_2 n_2(w_{j-1}\cdot) + D_3 n_3(w_{j-1}\cdot)}{C(w_{j-1})}$$

$$D_1 = 1-2Y\frac{n_2}{n_1},\quad D_2 = 2-3Y\frac{n_3}{n_2},$$

$$D_{3+} = 3-4Y\frac{n_4}{n_3}\quad (Y=\frac{n_1}{n_1+2n_2}) \quad (13)$$

Jelinek-Mercer smoothing 기법은 interpolated 형태이고 linear discount를 사용한다. 그 식은 다음 식 (14)와 같다[2].

$$P_{JM}(w_{j-1}|w_j) = \lambda\frac{C(w_{j-1}w_j)}{C(w_{j-1})} + (1-\lambda)P(w_j)$$

$$\lambda = n_1/N \quad (14)$$

(λ는 Good-Turing discount에서 나타나지 않은 이벤트와 같은 확률값이다.)

3. 언어모델의 평가

언어모델을 평가하는 일반적인 방법은 perplexity를 측정하는 것이다. 낮은 perplexity는 일반적으로 낮은 word error rate를 갖기 때문에 인식 성능에 중요한 영향을 미치게 된다. smooth된 n-gram모델이 확률값 $P(w_j|w_{j-1})$를 가지면 식 (15)를 사용하여 문장에 대한 확률값을 계산할 수 있다. 문장 $(t_1,\ldots,t_{l_T})$을 가지는 테스트 집합 T에 대한 확률값 P(T)는 다음 식 (15)처럼 테스트 집합에 있는 모든 문장의 확률값의 곱으로 나타낼 수 있다.

$$P(T) = \prod_{i=1}^{t_r} p(t_i) \quad (15)$$

이것에 대한 cross entropy $H_p(t)$는 식 (16)과 같다.

$$H_{p(T)} = -\frac{1}{W_T}\log_2 P(T) \quad (16)$$

W_T는 w로 측정한 테스트 T의 길이이다.

Perplexity $PP_p(T)$는 $2^{H_p(T)}$이다. 낮은 perplexity를 가진 언어 모델이 좋은 성능을 가진다[3].

III. VCCV 단위

인식 단위의 경계 추출은 무성음과 유성음 사이에서 뚜렷하게 나타나는 경우가 있지만 유성음, 무성음 혹은 유성음과 무성음들 사이에서 상호간의 조음현상 때문에 정확한 경계를 찾기가 어려울 뿐만 아니라, 이러한 경계 영역의 천이 구간에서 음성 데이터들은 별다른 의미를 갖지 못한다. 따라서 이러한 음소의 경계를 정확히 찾는 것보다 특성의 변화가 적은 안정된 대표구간인 모음 영역을 찾는 것이 더 효율적이라 할 수 있다. 이러한 단위를 음향 모델 뿐만 아니라 언어 모델의 언어 처리 단위로도 적용하기 위해서 텍스트를 VCCV로 분할하여 사용한다.

언어학적 특성인 모음 정보를 이용하여 VCCV 단위로 분할하는 예를 <표 1>에서 들었다.

<표 1> VCCV 분할의 예

강에 살고 읻슴니까
- 가/VC 앙에/VCV 에/V
- 사/VC 알고/VCCV 오/V
- 이/V 읻스/VCCV 음니/VCCV 이까/VCCV 아/V

코퍼스의 음절 수와 모음 정보를 이용하여 코퍼스를 VCCV 단위로 분할하는데 '강에'는 두 음절로 이루어져 있고, 그중에서 모음은 '아'와 '에'의 두 개로 이루어져 있다. 이것을 VCCV로 나누면 '아'와 '에'를 경계로 '가', '앙에', '에'의 세 부분으로 나누어진다.

본 논문에서는 코퍼스가 분할되어 나타날 수 있는 단위인 V(vowel), VC(vowel consonant) ,VCV(vowel consonant vowel), VV(vowel vowel), VCCV(vowel consonant consonant vowel), CV(consonant vowel) 등

을 한꺼번에 묶어 VCCV 단위라 한다.

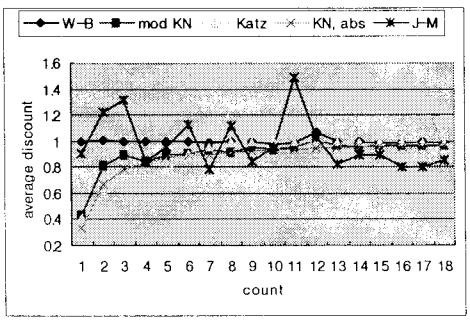
Ⅳ. 실험 및 결과

본 논문에서 사용한 DB는 스무고개 게임에 적용하기 위해서 스무고개에서 나올 수 있는 질문 중에 동물 영역의 데이터를 가지고 구성하였다. 전체 문장은 1746 문장이고 그 중에서 smoothing 할때 훈련에 사용된 문장은 1624문장이고 테스트에 사용한 문장은 122문장이다. 1746문장에 대한 어절, VCCV, 형태소의 개수와 복잡도 <표 2>와 같다.

<표 2> 어절, VCCV, 형태소 단위의 개수와 복잡도

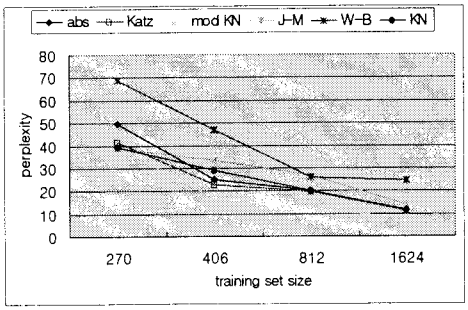
단 위		개수	복잡도
어 절		1955	3.98
VCCV	CV	154	2.35
	VCCV	1404	
	VC	40	
형태소		1543	2.57

<표 2>의 결과에서 VCCV 단위가 복잡도가 가장 적게 나오는 것을 볼 수 있다. VCCV 언어모델을 각 smoothing하고 discount 정도를 보기 위해 count 1-17까지의 평균 discount된 확률값을 구한 결과는 (그림 1)과 같다. 그림에서 good-turing discount나 abs- olute discount를 사용한 Katz, abs, KN방식은 bigram의 count수에 비례하여 count수가 적을수록 더 많이 discount되는 것을 볼수 있다.



(그림 1) bigram count에 따른 평균 discount된 확률값의 비

Training set 크기에 따른 smooth된 언어 모델의 성능을 평가하기 위해 trainging set 크기 270문장, 406문장, 812문장, 1624문장에 대한 test set의 perplexity를 계산하였다. (그림 2)에서 볼수 있듯이 training set 크기가 클수록 복잡도가 적게 나타났다.



(그림 2) training set size에 따른 perplexity

1624문장으로 training한 data에 대한 test data의 perplexity를 계산한 결과 modified Kneser-Ney smoothing이 가장 낮게 나왔다. 그러므로 본 논문에서 사용한 corpus에 대해서는 modified Kneser-Ney smoothing을 사용한 언어 모델 방법이 효율적인 것을 볼수 있다.

Ⅴ. 결론

본 논문에서는 VCCV 단위로 언어 모델을 구성하여 기존의 언어모델 언어처리 단위와 비교하였다. bigram 언어모델을 적용하여 코퍼스를 형태소, 어절, VCCV로 나누어서 실험한 결과 VCCV가 어휘수도 적게 나오면서 복잡도가 적게 나온 것을 볼 수 있었다. 복잡도가 적게 VCCV 언어 모델의 올바른 확률값 추정을 위해서 smoothing 기법을 적용하여 언어 모델의 성능을 평가하였다. 그 결과 언어 모델의 성능이 가장 좋게 나온 smoothing 방법이 Modified Kneser-Ney로 평가되었다. 향후 연구 과제로는 이러한 smoothing 기법을 가진 VCCV 언어 모델을 문장 음성인식에 적용하여 인식 성능을 평가하고 다른 단위와 비교 할 것이다.

< acknowledgement >
본 연구는 한국과학재단 목적기초연구 (R05-2002-000-01007-0) 지원으로 수행되었음.

참고문헌

[1] 이건상, 양성일, 권성헌 공저, "음성인식", 한양대학교 출판부, 2001.
[2] Stanley F.Chen and Joshua Goodman, "An Emperical Study of Smoothing Techniques for language modeling", Technical Report TR-10-98, Computer Science Group, Harvard Uni., 1998.
[3] Huang X., Acero A., Hon H.-W., "Spoken language processing", Prentice Hall PTR, October 2001.
[4] 오영환, "음성언어정보처리", 홍릉과학출판사, 1997.