말소리와 음성과학 제4권 제1호 (2012)
pp. 29~47

벅아이 코퍼스 오류 수정과 코퍼스 활용을 위한 프랏 스크립트 툴

Error Correction and Praat Script Tools for the Buckeye Corpus of Conversational Speech

윤 규 철[1)]
Yoon, Kyuchul

ABSTRACT

The purpose of this paper is to show how to convert the label files of the Buckeye Corpus of Spontaneous Speech [1] into Praat format and to introduce some of the Praat scripts that will enable linguists to study various aspects of spoken American English present in the corpus. During the conversion process, several types of errors were identified and corrected either manually or automatically by the use of scripts. The Praat script tools that have been developed can help extract from the corpus massive amounts of phonetic measures such as the VOT of plosives, the formants of vowels, word frequency information and speech rates that span several consecutive words. The script tools can extract additional information concerning the phonetic environment of the target words or allophones.

Keywords: Buckeye Corpus, Praat script, VOT, vowel formant, word frequency, speech rate, CMU dictionary

1. 서론

음성학의 연구 주제를 다루는 방법은 광범위하며 다양하지만 크게 세 가지로 볼 수 있을 것이다. 우선 가설을 바탕으로 이를 과학적으로 검증하는 방법이 있을 것인데 대부분의 연구가 이에 속한다고 할 수 있다. 다른 방법은 새로이 개발된 자료나 기술, 기법, 기자재를 이용하여 음성학적 현상의 실재를 있는 그대로 기술하고 밝히는 경우라고 볼 수 있을 것이다. 예를 들면, 스펙트럼이나 스펙트로그램 등의 기법을 활용하여 이전에 볼 수 없었던 방식으로 말소리를 표현하고 기술함으로써 말소리의 실체를 좀더 과학적으로 제시하는 것을 들 수 있을 것이다. 셋째는 관련 연구자들로 하여금 전술한 둘째의 방법으로 연구할 수 있도록 새로운 기법이나 연구 자료 혹은 소프트웨어를 개발하거나 활용할 수 있도록 하는 것으로 볼 수 있다. 예를 들면, 스펙트로그램 기법의 개발 및 응용, 또는 본 논문에서 소개할 자연발화 음성 코퍼스 자료인 벅아이 코퍼스(Buckeye Corpus of Conversational Speech)[1]나 음성 전문 소프트웨어인 프랏(Praat)[2] 등의 구축 및 개발이 이에 해당한다고 볼 수 있다.

본 논문은 오하이오 주립대학에서 이미 구축되어 공개된 벅아이 코퍼스를 바탕으로, 음성학 전공자들이 자연발화 음성의 다양한 주제를 연구하는데 도움이 되는 프랏 스크립트 툴을 개발하여 배포하는 것을 목표로 하고 있으며, 또한 개발 중에 발견된 코퍼스의 표기 상의 오류를 수정 보완하여 코퍼스의 활용도와 유용성을 증가시켰다.

벅아이 코퍼스는 미국 오하이오 주에 거주하는 영어 원어민들 대상으로 인터뷰 형식을 통하여 자연스런 대화를 녹음한 자연발화 음성 코퍼스이다. 녹음은 조용한 방에서 헤드셋 마이크를 착용한 채로 디지털 녹음기를 통해 실시되었다. 반복출현을 포함한 총 단어의 수는 약 30만개 정도이고, 반복을 제거한 단어 유형은 약 1만 3천여개 정도이다. 남녀노소별 각각 10명씩 총 40명의 미국인을 대상으로 구축되었으며 한 화자당 대략 1시간여 정도의 녹음이 저장되어 있고, 단어와 변이음별로 자동 레이블링 후 수동 검증이 되어 있으며, 연구목적으로 무료 배포되고 있는 매우 유용한 음성 코퍼스이다.

하지만, 자연 발화 영어에 대한 연구를 할 수 있는 귀중한 자료임에도 불구하고, 코퍼스가 연구용으로 무료 공개된 후 몇 년이 지났지만, 국내에서 특히 음성학을 전공하는 연구자들 사이에서 이 자료를 활용한 연구가 그리 활발히 진행되고

1) 영남대학교, kyoon@ynu.ac.kr.

접수일자: 2012년 2월 14일
수정일자: 2012년 3월 3일
게재결정: 2012년 3월 3일

있지 못한 것도 사실이다. 여러 이유 중 하나는 인문학 쪽에서 손쉽게 접근하기 힘든 방대한 자료라는 점이고, 또 하나는 음성 파일에 부속된 레이블 파일이 이미 사라진 Xwaves라는 소프트웨어로 작성되었고 현재 WaveSurfer[2]와 프랏에서 한 쌍씩 열람이 가능하긴 하지만, 음성 파일과 부속된 레이블 파일이 255개의 쌍으로 존재하는 방대한 자료에서 음성학 연구 주제에 맞는 부분만을 선별적으로 찾아내는 것은 매우 힘들기 때문일 것이다.

그림 1. SpeechSearcher 인터페이스
Figure 1. Interface of SpeechSearcher

벅아이 코퍼스를 배포하는 누리집에서 <그림 1>과 같은 SpeechSearcher라는 검색 프로그램을 2009년 8월부터 배포하기 시작하였다. 이 프로그램은 tcl 스크립트와 WaveSurfer를 이용하여 단어나 변이음 기호를 써넣으면 해당되는 음성과 레이블을 자동으로 찾아주는 기능을 갖고 있어 연구자들이 유용하게 이용할 수 있다. 하지만, 이러한 검색툴만으로는 아직까지 연구자들의 다양한 연구 주제에 맞는 검색을 하기에는 무리이다. 앞으로 더 다양한 종류의 검색 툴이 개발될 것임은 의심할 여지가 없지만, 현재 상태로는 코퍼스의 가치를 충분히 보상할만한 활용 툴이 부족한 것이 사실이다.

이상적인 경우는 인문학 분야의 음성학 연구자들이 필요에 따라서 자료를 스스로 찾아 쓰도록 툴을 만들어 쓸 수 있도록 하는 것인데, tcl 스크립트는 인문학 분야에 그리 널리 퍼져 있지 않는 스크립트 언어이므로 여의치 않다고 볼 수 있다. 대신 최근에 음성학자들을 중심으로 널리 활용되고 또한 스크립트 언어에 상대적으로 능통한 사람들이 점차 생겨나고 있는 프랏을 활용하는 것이 차선책으로 마땅하다고 생각된다.

따라서 본 논문에서는 벅아이 코퍼스의 255개 레이블 파일을 모두 프랏의 TextGrid(이하 텍스트그리드) 포맷으로 전환하는 방법을 소개하고, 전환 절차 중에 발견하게 된 몇 종류의 오류와 이를 수정하는 방법 및 수정한 결과를 보고하고자 한다. 또한 본격적으로 프랏 포맷의 레이블 파일인 텍스트그리드를 활용하여 음성학 연구를 수행하는데 도움을 줄 수 있는 몇 가지 프랏 스크립트 툴을 개발하여 소개하고자 한다.

우선 코퍼스 레이블 파일을 프랏 포맷으로 바꾸는데 이용할 수 있는 스크립트 세트를 소개하고, 코퍼스에서 파열음을 지닌 단어들을 찾아내어 VOT를 측정하고 동시에 해당 음소 주변의 다양한 정보를 동시에 텍스트 파일로 저장하는 스크립트, 영어 모음별로 포먼트를 측정하면서 동시에 해당 모음 주변의 다양한 음성학적 정보를 텍스트 파일로 저장하는 스크립트, 벅아이 코퍼스를 구성하는 모든 단어들의 출현 빈도수를 자동 측정하는 스크립트, 단어의 음절수를 자동으로 셀 수 있는 스크립트, 또 이를 바탕으로 초당 음절수로 정의되는 발화 속도를 계산하는데 필요한 스크립트 등을 소개한다.

본 논문은 두 부분으로 구성되어 있다. 우선 벅아이 코퍼스를 구성하는 파일 중에서 레이블 파일들을 프랏 형식으로 변환하고 이 과정에서 발견된 코퍼스 자체의 몇몇 오류를 수정하여 처리하는 부분이고, 다음으로 수정한 코퍼스를 활용할 수 있는 스크립트 툴의 소개이다.

2. 연구방법

2.1 벅아이 코퍼스의 변환 및 수정

2.1.1 벅아이 코퍼스 레이블 포맷의 전환

벅아이 코퍼스는 구축 당시에 Xwaves라는 프로그램을 이용하여 레이블링이 되었는데, 이 프로그램을 개발한 회사는 다른 회사에 합병되었고 해당 프로그램은 더 이상 판매되거나 사후 관리가 되고 있지 않은 상태이다. 계속적인 사후 지원이 되지 않기 때문에, 설혹 프로그램을 구한다 할지라도 일반 연구자들이 자신의 컴퓨터에 설치하여 사용할 수는 없다. 윈도우 운영체제 용 버전은 없으며 유닉스 계열의 시스템에만 설치하여 사용할 수 있는데, 그나마도 최근의 시스템에 설치되어 있는 그래픽 카드 등을 지원하지 못하여 있어도 활용할 수 없는 상황이다. 다행히도 프랏에서는 Xwaves에서 생성된 레이블 파일을 읽어들여 프랏의 레이블 포맷으로 살펴볼 수 있고 또 저장할 수도 있는 명령어인 Read IntervalTier from Xwaves를 내장하고 있어 일괄적으로 변환하는데 이용할 수 있다.

벅아이 코퍼스 배포 당시에 받은 세 종류의 Xwaves 레이블 파일, 즉 단어별 레이블, 변이음별 레이블, 기타정보 레이블을 <부록 1>의 프랏 스크립트를 이용하여 일괄적으로 읽어 들여 프랏 형식의 레이블 파일로 바꾼 다음, <부록 2>의 프랏 스크립트를 이용하여 이 세 종류의 레이블 파일을 하나로 통합하여 레이블 부분의 순서가 위에서부터 단어층, 변이음층, 로그층의 형식이 되도록 하였다. 또한 변환 후에 매 단어나 변이음 끝에 세미콜론이 붙어 있어 이를 간단한 스크립트로 또한 일괄적으로 제거하였다. 단어와 변이음 층에는 화자들이 말한 단어와 그 단어에 사용된 변이음이 기록되어 있으며 로그층에는 음성의 종류 등의 정보가 기록되어 있다.

2.1.2 층간 경계 불일치 오류 유형

벅아이 코퍼스의 레이블 정보는 자동 음성 인식을 이용하여 생성된 후 수작업으로 확인 및 정정을 거쳤기 때문에 상당히 정확한 고급 정보라고 볼 수 있다. 하지만, 이 정보를 본 연구에서 이용하려면 주의해야 할 특성이 존재한다. 그것은 바로 각 층의 구간별 경계가 일치하지 않는 경우가 존재한다는 것이다.

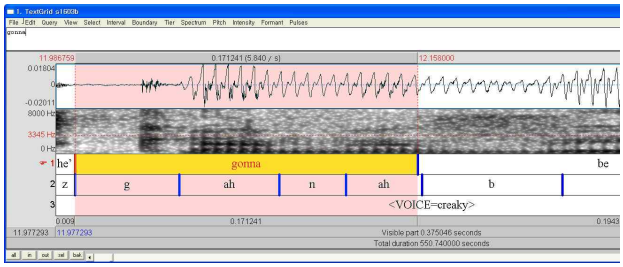

그림 2. 층간 경계 불일치 오류의 예
Figure 2. Sample of unsynchronized inter-tier boundaries

<그림 2>를 보면 맨 위의 단어 층과 두 번째의 변이음 층 사이에 경계가 일치하지 않는 것을 볼 수 있다. 즉, 단어 “gonna”의 끝 경계가 변이음 층의 “ah”의 끝 경계와 일치하지 않는 것을 볼 수 있다. 사람의 눈으로 볼 때에는 그다지 큰 문제가 되지 않을 수도 있다. 경계 사이에 약간의 차이가 있더라도 이를 고려하여 단어와 변이음의 소속 관계를 충분히 알아낼 수 있기 때문이다.

하지만, 40여명이 1시간여 동안 말한 총 40여시간의 방대한 음성 파일을 대상으로, 예를 들어 특정 모음의 포먼트를 자동으로 측정하고 각 모음이 어떤 단어에 소속되어 있고 그 단어의 어두에 있는지 아니면 어중인지, 또 이 단어가 발화 문장의 처음에 속하는지 아닌지 등의 정보를 스크립트를 이용하여 자동으로 추출해 내고자 할 때에는 어려움에 부딪히게 된다. 인간의 추론 과정은 아무리 단순한 과정처럼 보여도 그 밑에는 꽤 복잡한 논리적인 과정이 밑바탕이 되어 있지만 컴퓨터로 같은 과정을 재현하는 것은 그만큼 어렵기 때문이다.

예를 들어 <그림 2>에서 “gonna”의 마지막 변이음인 “ah”가 해당 단어의 어두에 위치하는지 아니면 어중인지 아니면 어말인지를 컴퓨터로 하여금 알아내게 하는 일은 간단치 않다. 우선 해당 단어나 변이음 구간이 몇 번째 구간인지 구간 번호 정보를 이용하려 해도 각 층에 존재하는 구간의 개수가 달라 이용할 수 없다. 유일하게 이용할 수 있는 정보는 변이음의 시작과 끝 경계의 시점 정보이다.

이 정보를 이용하여 해당 변이음이 어떤 단어의 어말에 있다는 것을 알아내려면 변이음의 오른쪽 경계, 즉 변이음의 끝 시점이 단어의 오른쪽 경계와 일치한다는 것을 보이면 된다. 그런데 <그림 2>의 경우에서 보듯 코퍼스의 레이블 구축 당시의 오류로 인해 일치해야할 이 경계가 일치하지 않고 있고 이로 인해 자동으로 정보를 추출해 내기가 어렵다. 이러한 오류가 코퍼스 내에 상당히 많이 존재하기 때문에 이를 일괄적으로 수정하는 작업이 반드시 필요한 것이다.

이러한 경계 일치 오류를 수정하려면 전술한대로 단어와 그 단어 안의 변이음 사이의 시점 정보에 대한 관계를 활용하면 될 것이다. 즉, 어떤 변이음이 소속 단어의 어두에 있다면 그 변이음의 시작 부분의 시점이 소속 단어의 시작 시점과 일치할 것이고, 변이음이 어말에 있다면 변이음의 끝 시점과 단어의 끝 시점이 일치할 것이며, 변이음이 어중에 있으면 단어의 시작과 끝 시점 사이에 변이음의 시간대가 존재할 것이다.

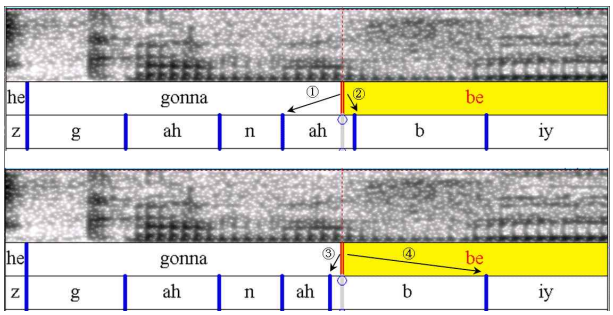

그림 3. 경계 불일치 오류 수정 방법
Figure 3. A way to fix the inter-tier sync error

<그림 3>에서는 단어 “gonna”의 변이음 “ah”가 단어의 어말 위치에 있을 때, 경계가 일치하지 않는 두 가지 경우를 보여주고 있다. <그림 3>의 윗 부분은 <그림 2>의 경우처럼 “ah” 변이음의 끝 경계가 단어 “gonna”의 끝 경계를 넘어서 오른쪽으로 잘못 표시된 오류를 보여주고 있고, 밑 부분은 설명하기 위한 목적으로 인위적으로 만들어낸 오류인데, 윗 부분과는 반대로 단어의 끝 경계에 미치지 못하고 왼쪽으로 잘못 표시된 것을 보여주고 있다.

2.1.3 층간 경계 불일치 오류 수정

프랏 스크립트를 이용하여 자동으로 이러한 오류를 수정하는 작업은, 우선 오류가 생긴 단어 경계를 중심으로 시작한다. <그림 3>의 윗 그림을 보자. 오류가 난 단어 경계의 시점을 알아내어 그 시점에 걸쳐 있는 변이음이 무엇인지를 알아낸다. 그 다음 그 변이음의 양쪽 경계, 즉 시작과 끝의 시간을 알아낸 다음, 이 두 시점들과 단어 경계 시점과의 차이를 계산한다. 단어 경계와 변이음 시작 시점과의 차이는 ①이 되고, 단어 경계와 변이음 끝 시점과의 차이는 ②가 될 것이다. 이 둘 중에서 값이 작은 ② 쪽의 변이음 경계가 바로 수정되어야 할 오류 경계인 것이다. 따라서 이 경계를 이동하는 작업이 수행되어야 한다.

<그림 3>의 아랫 부분도 유사한 방법으로 진행하면 된다. 오류가 난 단어 경계의 시간을 통해 그 부분에 걸쳐있는 변이음을 알아낸다. 이번에는 다음 단어인 “be”의 변이음 “b”가 된다. 그 변이음의 시작과 끝 경계 시점이 오류가 난 단어 경계와 얼마만큼의 시간 차이가 나는지 ③과 ④를 알아내어 이들 중에서 작은 값에 해당하는 ③ 쪽의 변이음 경계를 수정 대상으로 잡고 이동시키면 된다.

이러한 방법을 이용하여 경계 불일치 오류를 자동으로 찾아내고 자동으로 수정하기 위하여 <부록 3>에 있는 프랏 스

크립트를 작성하였다. 이 스크립트는 우선 단어층을 대상으로 모든 단어 경계를 하나 하나 조사하여 이 경계 중에서 변이음층의 경계와 정확히 일치하지 않는 것이 검출되면 그 경계를 대상으로 <그림 2>에서 상술한 방식으로 변이음 경계를 이동하여 수정하게 되며 또 수정 중에 텍스트 파일로 어느 단어 경계가 오른쪽(+) 혹은 왼쪽(-)으로 얼마나 잘못 설정되어 있었는지를 출력하여 저장한다. 아래에는 화자 s01의 레이블 파일에서 오류가 난 단어와 그 구간 번호 그리고 좌우 에러 시간의 일부를 보여주고 있다.

file	interval#	word	syncError(+/-,sec)
s0101a	52	position	+0.00010900000000191312
s0101a	82	and	+6.399999999473494e-05
s0101a	86	from	+0.0006779999999935171
s0101a	95	working	+0.0006779999999935171
s0101a	124	um	+0.0002420000000000755

2.1.4 그 밖의 오류 유형 및 수정

이 밖에도 레이블과 사운드 파일에서 몇 가지 유형의 오류가 발견되어 수정되었다. 예를 들면, Xwaves로 만든 레이블 파일을 프랏 포맷으로 변경 시 오류가 생기는 경우가 있는데 이는 대부분 원래 파일의 헤더 정보 중의 하나인 nfields 값이 맞지 않는 경우였다. 단어 레이블 정보를 담고 있는 .words 확장자를 지니고 있는 파일의 경우 이 값이 3이 되어야 하는데, 가끔 1로 잘못 표기되어 있었고 이를 수정하면 프랏 포맷으로 변환되는데 문제가 없었다.

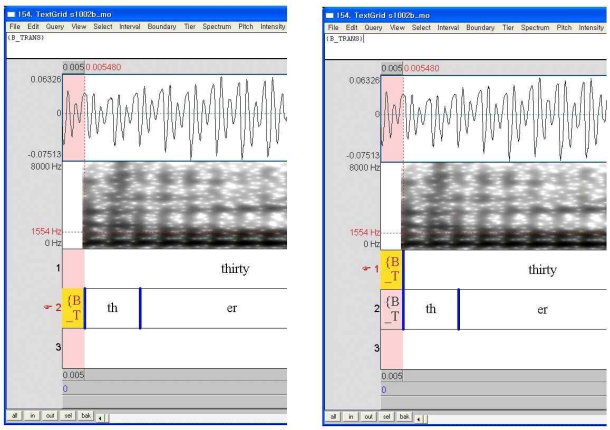

그림 4. 레이블 시작 부분에서의 오류
Figure 4. An error at the beginning of a label file

벅아이 코퍼스의 모든 레이블 파일은 처음 구간과 마지막 구간이 시작과 끝을 알리는 {B_TRANS}와 {E_TRANS}로 표시되는데, <그림 4>의 왼쪽과 같이 가끔 이러한 표기가 누락되어 있는 경우가 있었다. 이런 경우는 그리 많지 않아 오른쪽과 같이 필요한 경계와 정보를 수작업으로 추가해 주었다.

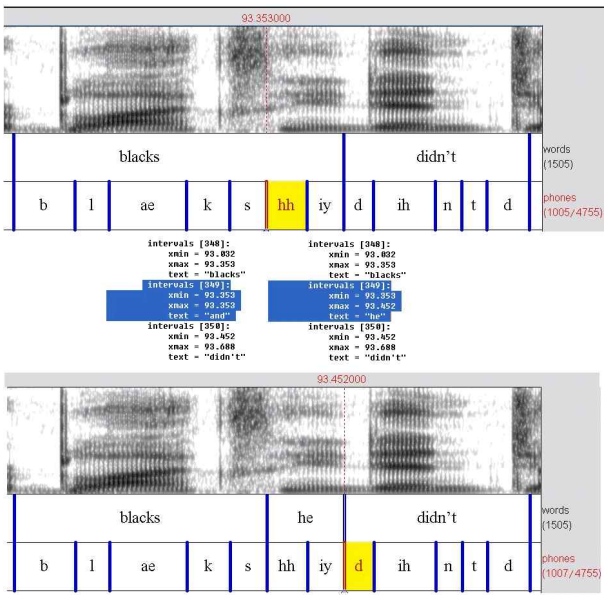

그림 5. 단어층에서 경계 표시가 되지 않은 오류
Figure 5. An error of a missing boundary at the word tier

또 다른 오류의 예는 프랏 스크립트를 실행하다가 단어층의 구간 번호 형식이 맞지 않는다는 오류 메시지를 통해 발견되었는데, 단어 경계가 있어야 하는 곳임에도 불구하고 단어 경계가 표시되어 있지 않은 오류였다. <그림 5> 윗부분에서와 같이 “blacks he didn't”로 단어 경계가 표시되어야 하나 “he” 부분의 경우 변이음은 표시되어 있으나 단어 시작 경계가 누락되어 있다. 이 경우는 텍스트그리드를 조작해서는 변화가 생기지 않아, 그림의 중간 부분에서처럼 레이블 파일인 텍스트 파일을 직접 수작업으로 열어 시작 경계 시간을 직접 수정해주어야만 아래 그림처럼 경계가 살아난다.

이 밖에도 음성 녹음이 되어 있지 않은 부분에 레이블 정보가 존재하는 경우도 몇몇 발견되어 필요 없는 레이블 정보는 잘라내기도 하였다.

또한 벅아이 코퍼스의 오류는 아니지만, 음성 파일과 레이블 파일에 성별과 연령대 정보를 표시하도록 이름을 바꾸었다. 코퍼스 매뉴얼에 성별과 연령대 정보가 있지만, 파일 이름에 직접 표시될 경우 후반에 소개될 스크립트를 짜는 경우에 여러 면에서 편리할 수 있기 때문에 성별은 m(ale)과 f(emale)로, 연령대는 y(oung)과 o(ld)로 표시하였다. 예를 들면 s0101a.wav 파일은 s0101a_fy.wav로 바꾸었다.

2.2 수정된 벅아이 코퍼스 활용을 위한 스크립트

이번에는 앞에서 수정한 벅아이 코퍼스를 이용하여 음성학 연구에 활용할 수 있는 몇몇 스크립트를 소개하고자 한다. 자음과 모음 연구에 대표적인 방법 중의 하나인 VOT 정보와 모

음 포먼트 정보를 벅아이 코퍼스에서 추출해 내는 방법을 소개하고자 한다. 특히 VOT와 포먼트 뿐 아니라 이와 관련된 부가 정보를 함께 뽑아낸다는 점이 특징이다. 예전의 연구에서는 녹음실에서 주어진 문장이나 단어를 읽고 말하는 경우가 많았는데 이러한 경우 여러 요인들을 통제하기가 비교적 쉬웠지만, 벅아이 코퍼스는 그러한 요인을 통제할 수 없으므로 대신에 그런 요인들을 더불어 추출해 내야만 의미 있는 분석을 수행할 수 있을 것이다.

부가 정보에 포함되는 사항들은 자음 혹은 모음의 단어 내 위치(어두, 어중, 어말), 해당 음절의 강세 여부, 단어의 내용어 혹은 기능어 여부, 직전과 직후의 변이음(들) 혹은 단어(들), 단어의 음절수, 단어의 발화시간, 음절수와 발화시간을 이용한 발화속도 계산, 휴지기 존재로 파악하는 발화문 내의 위치(문두, 문중, 문말), 단어의 코퍼스 전체 내 출현 빈도수 등이다. 이들 정보가 아래에 소개하는 모든 스크립트에 다 포함되어 있는 것은 아니지만, 필요에 따라 일부 혹은 전부를 포함하도록 스크립트를 수정할 수도 있을 것이며 또 이를 권장하는 바이다.

2.2.1 파열음의 VOT를 측정하고 주변 정보 추출하기

벅아이 코퍼스에 존재하는 파열음으로부터 VOT를 재고 주변 정보를 추출하기 위한 스크립트는 <부록 4>에 있다. 이 스크립트를 실행하면 처음에 <그림 6>과 같은 창이 나타난다. 그러면 여기에 찾고자 하는 파열음의 기호 등을 넣고 확인을 눌러 실행시키면 된다.

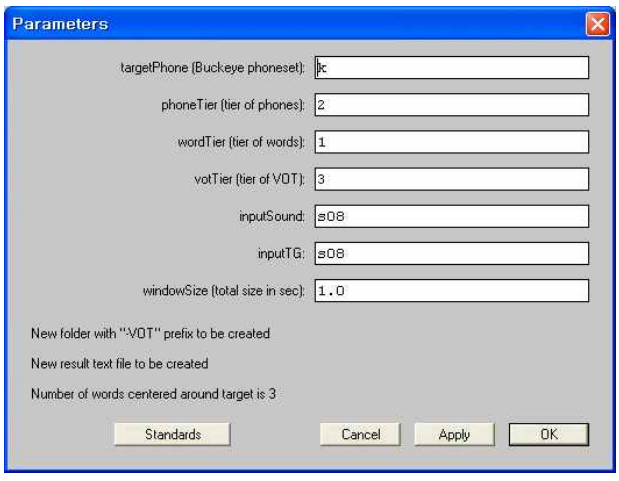

그림 6. VOT를 측정하는 프랏 스크립트
Figure 6. Praat script for measuring VOT

실행이 되면 스크립트는 변이음 층에서 해당 파열음이 발견되는 순서대로 사용자에게 <그림 7>과 같은 화면을 보여준다. 그러면 사용자는 수동으로 VOT 구간을 선택한 다음 계속 버튼을 누른다. 그 후 스크립트는 레이블 파일에 <그림 7>에서 보듯이 해당 VOT 구간의 경계점과 변이음 기호를 삽입하고 다음 파열음으로 계속해서 진행한다.

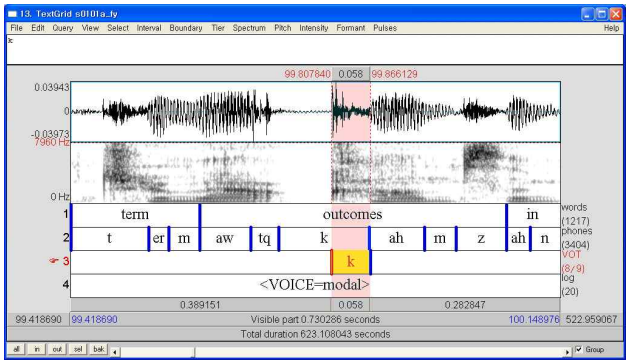

그림 7. 표시한 VOT를 레이블 파일에 기록한다
Figure 7. Inserting boundaries for the marked VOT

동시에 일반 텍스트 파일로도 관련 정보가 출력된다. 자동으로 생성된 결과 텍스트 파일에는 파일이름, 변이음 구간 번호, 변이음이 속한 단어의 구간 번호, msec로 표시된 VOT, 단어 내 위치, 이전 변이음의 [s] 여부, 직전과 직후의 단어와 각각의 경과 시간을 msec로 표시해 준다. 이러한 정보를 후에 소개할 발화속도 계산에 활용하면 속도에 따른 VOT의 변화 양상을 살펴보는 것도 가능하다.

2.2.2 모음 포먼트 측정과 주변 정보 추출하기

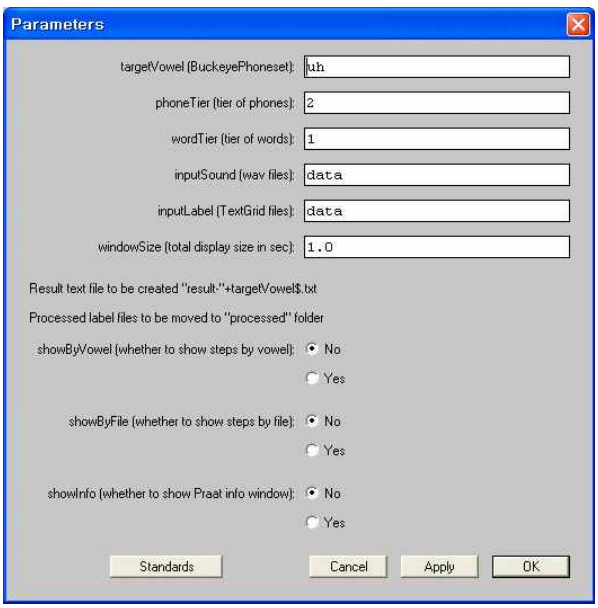

그림 8. 모음의 포먼트를 측정하는 프랏 스크립트
Figure 8. Praat script for measuring formants

벅아이 코퍼스에 존재하는 모음의 포먼트를 자동으로 재고 주변 정보를 추출하기 위한 스크립트는 <부록 5>에 있다. 이

스크립트를 실행하면 <그림 8>과 같은 설정 창이 나타난다. 여기에 포먼트를 측정하고자 하는 모음 기호와 기타 정보를 넣으면 된다. 특히 진척 상황을 실시간으로 모음별로 혹은 파일별로 확인하거나 별도의 정보창으로 확인하고자 할 때에는 showByVowel, showByFile, showInfo의 Yes/No 부분을 선택하면 된다. 또한 성별에 따라 포먼트 세팅 중 Maximum formant 값이 남성이면 5000, 여성이면 5500을 취하도록 하였다. 이는 파일 이름에 성별 정보를 표시하도록 파일 이름을 미리 변경해 놓았기 때문에 가능하다.

모음 포먼트 측정이 시작되면 프랏은 <그림 9>과 같은 화면을 순차적으로 보여주면서 자동으로 성별에 따라 포먼트를 추출하여 이를 텍스트 파일에 기록하게 된다. 결과 텍스트 파일에 실시간으로 기록되는 정보는 파일이름, 성별, 연령대, 소속 단어와 구간 번호, 직전과 직후의 두 단어 총 네 단어, 각 단어의 소요 시간, 단어 내 혹은 문장 내 위치, 측정 모음, 직전과 직후의 변이음, 두 포먼트 값 등이다.

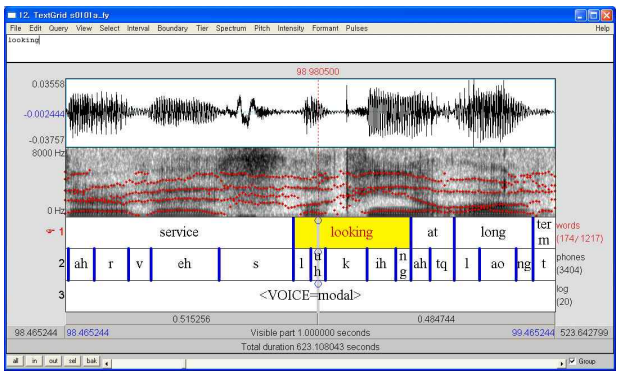

그림 9. 모음 포먼트를 측정하는 화면
Figure 9. Measuring formants

2.2.3 벅아이 코퍼스 단어 출현 빈도수 측정하기

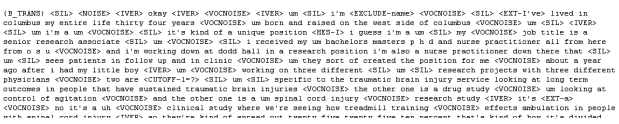

그림 10. 레이블 파일로부터 추출된 단어들
Figure 10. Words extracted from the label files

앞에서 수정된 벅아이 코퍼스 255개의 레이블 파일들을 대상으로 모든 단어를 추출하여 이들의 출현 빈도수를 측정하기 위하여 우선 <부록 6>의 스크립트를 이용하였다. 이 스크립트는 단어층에 모든 구간에 존재하는 단어를 추출하여 <그림 10>에서와 같이 일반 텍스트 파일로 출력하여 주는 기능을 지니고 있다. 약 10분 분량인 각 레이블 파일에 존재하는 모든 단어들을 한 문장으로 취급하여 하나의 문단으로 출력하도록 되어 있다. 따라서 출력된 하나의 텍스트 파일에 255개의 문장 혹은 문단이 존재하게 된다.

이 파일을 다시 입력으로 하여 <부록 7>의 스크립트를 이용하여 단어 목록과 빈도수 등 두 컬럼으로 구성된 결과 텍스트 파일을 출력하게 된다. 이 스크립트는 입력된 문장을 빈칸을 기준으로 하여 단어로 나누는 단계(<부록 7>의 TOKENIZE 부분), 각 단어에 대하여 알파벳을 제외한 기호를 제거하고 소문자로 바꾸는 전처리 단계(<부록 7>의 PREPROCESS 부분), 중복되어 출현하는 각 단어의 빈도수를 세는 단계(<부록 7>의 COUNT TOKEN 부분), 단어 목록과 단어별 빈도수 등 두 개의 컬럼을 지닌 텍스트 파일로 출력하는 단계(<부록 7>의 PRINT FREQUENCY 부분)로 구성되어 있다.

2.2.4 발화속도 계산하기

발화속도는 1초에 발화한 음절수로 정의하고 이를 계산하기 위한 측정구간 혹은 윈도우의 설정은 세 가지로 살펴보았다. 우선 해당 단어만을 고려하는 경우와, 해당 단어를 기준으로 좌우로 한 단어씩 추가하여 세 단어를 고려하는 경우, 그리고 해당 단어를 기준으로 좌우에 두 단어씩 추가하여 다섯 단어를 고려하는 경우로 나누어 고려하였다. 해당 단어만을 고려하는 경우 주변 단어들과의 연속적이고 유기적인 관계를 놓칠 수 있어 그 범위를 점차 확대하여 고려하고자 하였다.

우선 초당 발화된 음절수로 정의된 발화속도를 계산하기 위해서는 각 단어의 음절수가 몇인지를 알아야한다. 그러나 엄청난 수의 단어에 대하여 그 음절수를 일일이 수작업으로 입력하기는 불가능하다. 그래서 음절수와 관련된 정보는 코퍼스에서 직접 얻지 못하고 Carnegie Mellon University에서 배포하는 CMU Pronouncing Dictionary[4]를 활용하여 간접적으로 계산되었다.

전산언어학 등에서 널리 활용되는 이 영어 사전은 125,000개의 영어 단어와 이들의 발음이 강세 정보와 함께 텍스트 파일로 담겨 있다. 단어의 각 모음마다 0(강세 없음), 1(제1강세), 2(제2강세)가 표시되어 있다. 예를 들어 단어 “happy”는 이 사전 상에 “HH AE1 P IY0”로 표기되어 있는데 모음 기호 바로 옆에 숫자가 붙어 있는 것을 알 수 있다. 모든 모음 옆에 숫자가 있으므로 각 단어마다 이 숫자가 몇 개 있는지를 세면 음절수를 간접적으로 계산할 수 있다. 우선 <부록 8>의 프랏 스크립트를 이용하여 CMU 사전의 전 단어를 대상으로 강세 종류별 수를 계산하여 각 단어별로 총 음절수를 계산하여 데이터베이스로 이용하기 위한 텍스트 파일로 출력을 하였다. 이 텍스트 파일을 데이터베이스로 활용하여 <부록 9>의 프랏 스크립트를 이용하면 각 단어별로 음절수 정보 자동으로 구할 수 있다.

단어별 음절수 정보와 단어별 소요 시간이 구해지면 이를 바탕으로 발화속도를 구할 수 있다. <부록 10>의 스크립트를 이용하여 관찰대상인 단어가 존재하는 부분의 정보를 자동으

로 추출할 수 있다. <그림 11>에서와 같이 발화속도를 측정하고자 하는 단어가 첫 번째 컬럼(컬럼 이름은 TOKEN)으로 들어 있는 텍스트 파일과 음절수 정보가 들어 있는 데이터베이스 파일을 입력으로 지정하고 실행하면, 파일이름, 성별, 연령대, 단어 구간 번호, 다섯 단어 목록, 각 단어 구간별 시간, 단어별 음절수, 발화속도 등의 정보를 하나의 텍스트 파일에 출력하게 되며, 동시에 다섯 단어를 포함하는 사운드와 레이블 파일을 잘라내어 후속 처리 및 측정을 위하여 별도의 폴더에 보관하게 된다.

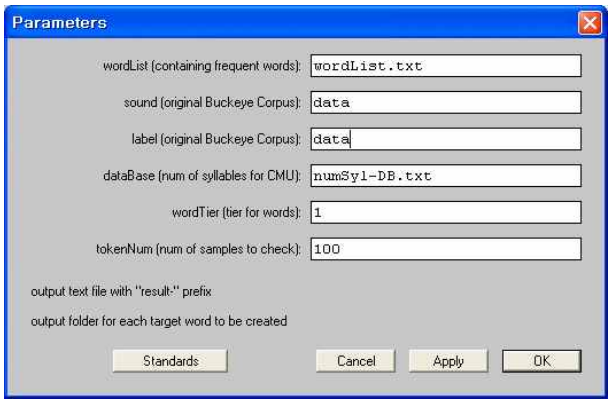

그림 11. 발화속도를 추출하는 스크립트
Figure 11. Praat script for calculating speech rates

3. 연구결과

3.1 벅아이 코퍼스의 오류 수정 결과

단어층이 변이음층과 경계가 일치하지 않는 부분을 자동으로 검출하여 오류를 수정한 결과를 살펴보자. 벅아이 코퍼스를 구성하는 전체 레이블 파일의 개수는 255개이다. 경계 불일치 오류는 총 126,660개가 발견되었다. 이 오류는 변이음 경계가 있어야 할 곳보다 왼쪽 혹은 오른쪽에 잘못 표시된 경우인데 오류 전체의 분포를 히스토그램을 통해서 보면 <그림 12>과 같다.

있어야 할 곳인 0초를 기준으로 좌우로 비교적 균일하게 퍼져 있다. 방향을 고려하면 평균적으로 0.24932 msec만큼 왼쪽으로 잘못 표시되어 있었고, 이를 프랏 스크립트가 자동으로 경계를 수정하여 레이블 파일을 출력하였다. 양 방향을 고려하지 않고 순수하게 잘못 표시된 양을 따지기 위해 음수를 양수로 바꾸어 절대값에 대한 평균을 계산해 보면 1.36114 msec이다. 이는 어느 쪽으로든 평균 1.36114 msec만큼 경계가 어긋나 있다는 것을 의미한다. 또한 수평축을 살펴보면 다수의 경계 불일치 오류가 양 방향으로 5 msec 안에 있는 것을 볼 수 있다.

255개의 레이블 파일을 하나 하나 살펴보았을 때 개별 파일 하나 당 오류의 개수를 히스토그램으로 나타내면 <그림 13>와 같다. 레이블 파일 하나 당 평균 499개의 오류가 존재했으며 히스토그램에서 보듯이 빈도수로 보면 많은 레이블 파일의 경우 이보다 적은 수의 오류가 존재하고 있었다.

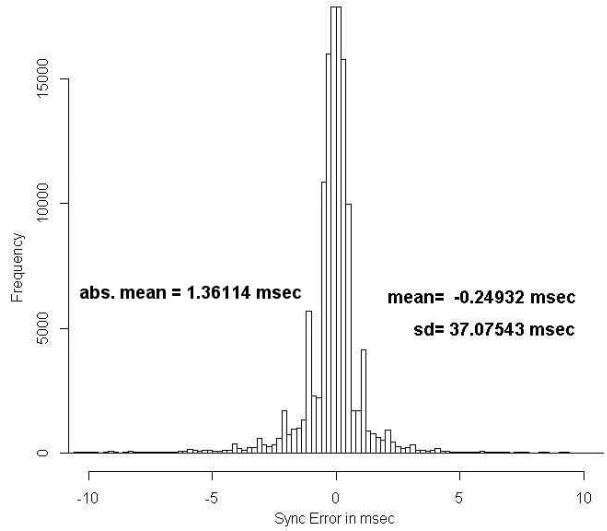

그림 12. 경계 불일치 오류의 시간 분포 히스토그램
Figure 12. Histogram of the boundary sync errors

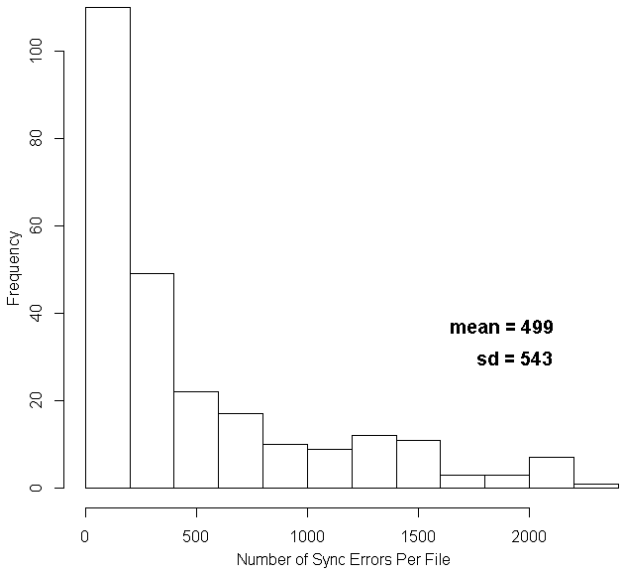

그림 13. 파일 당 오류 개수 히스토그램
Figure 13. Histogram of the number of errors per file

레이블 파일 하나 당 평균 1392개의 단어가 있으므로 레이블 파일 하나에 존재하는 전체 단어 중에 약 1/3에 못미치는 비율로 단어 경계 불일치 오류가 존재하고 있었다. 이러한 오류 수정을 통해 생성된 레이블 파일은 단어 경계가 변이음 경계와 정확하게 일치하기 때문에 본 연구 이외에도 후속 연구나 관련 연구에 유용하게 사용될 수 있을 것이다.

3.2 파열음의 VOT 측정 및 주변 정보 추출

스크립트를 활용하여 벅아이 코퍼스에 존재하는 파열음의

VOT와 주변 정보를 추출하면 <그림 14>와 같다. 첫줄에서 보듯이 파일이름, 변이음 구간 번호, 단어 구간 번호, 조음위치, VOT, 단어 내 위치, 이전 변이음의 [s] 여부, 직전과 직후 단어와 각각의 소요 시간 등의 정보가 텍스트 파일로 출력되었음을 알 수 있다.

File	iPhone	iWord	POA	VOT	LocWord	AfterS	preWord	Word	nexWord	preDur	Dur	nexDur
s0101a	99	48	k	60	initial	YES	it's	kind	of	140	180	94
s0101a	231	95	k	59	medial	no	i'm	working	down	337	284	202
s0101a	476	174	k	48	medial	no	service	looking	at	591	241	88
s0101a	552	197	k	65	medial	no	um	looking	at	272	240	91
s0101a	557	199	k	46	initial	no	at	control	of	91	347	103
s0101a	595	211	k	67	initial	no	spinal	cord	injury	372	165	433
s0101a	693	240	k	52	initial	no	spinal	cord	injury	259	151	367
s0101a	707	245	k	59	initial	no	they're	kind	of	109	188	42
s0101a	743	256	k	55	medial	YES	that's	kind	of	146	168	94
s0101a	916	310	k	51	initial	no	we	can	write	82	153	133
s0101a	942	317	k	55	medial	no	so	we	can	350	100	208
s0101a	1014	340	k	66	initial	no	to	come	in	62	207	127
s0101a	1019	342	k	44	initial	no	in	confirm	our	127	445	117
s0101a	1123	375	k	57	initial	no	gotta	couple	house	188	241	222
s0101a	1393	483	k	65	initial	no	were	kind	of	59	177	81
s0101a	2510	900	k	67	initial	no	i	can't	imagine	68	217	484
s0101a	2554	918	k	68	initial	no	i	can't	imagine	60	197	519
s0101a	2978	1065	k	124	initial	no	they	can	go	130	300	368
s0101a	3050	1094	k	94	initial	no	all	kind	of	170	372	87
s0101a	3269	1169	k	54	initial	no	doing	quite	well	190	109	459

그림 14. VOT 측정 결과

Figure 14. Result from VOT measurements

3.3 모음 포먼트 측정 및 주변 정보 추출

모음 포먼트와 주변 정보를 추출하는 스크립트를 실행하여 얻은 결과는 <그림 15>와 같다. 결과 파일에서 보듯이 맨 뒤의 F1, F2 컬럼 앞으로 여러 가지 주변 정보들이 추출되어 있다. 즉, 파일이름, 화자번호, 단어 구간 번호, 추출 단어 직전과 직후의 두 단어들을 포함 총 다섯 단어와 각각의 소요 시간, 모음 구간 번호, 직전과 직후의 변이음, 모음 길이, 단어 내 위치, 발화문장 내 위치 등의 정보가 출력된다.

File	Subj	cWrdInt	ppWord	preWord	cWord	nexWord	nnWord	ppWdDur	preWDur	cWDur	nexWDur	nnWDur
s0601a	s06	10	are	still	married	<SIL>	i'm	67	282	773	185	496
s0601a	s06	84	other	one's	fifteen	years	older	140	218	425	231	282
s0601a	s06	93	it	didn't	seem	that	much	213	228	315	213	388
s0601a	s06	103	<IVER>	i	mean	my	<SIL>	620	51	493	545	330
s0601a	s06	115	he	wasn't	very	social	in	144	328	220	432	105
s0601a	s06	127	i'm	a	pre	art	major	223	58	259	283	360
s0601a	s06	160	lot	of	people	are	really	197	40	483	114	338
s0601a	s06	162	people	are	really	narrow	minded	483	114	338	343	480
s0601a	s06	169	work	on	really	trying	to	286	325	378	332	64
s0601a	s06	176	in	college	people	are	more	198	869	459	232	243
s0601a	s06	180	more	open	idea	like	open	243	466	499	311	349
s0601a	s06	188	they're	more	willing	to	express	124	228	270	112	398
s0601a	s06	193	their	own	ideas	<IVER>	i	208	137	729	6805	157
s0601a	s06	207	you	don't	see	like	<SIL>	238	250	622	609	668
s0601a	s06	241	their	own	thing	<IVER>	um	181	158	402	6747	612
s0601a	s06	263	found	myself	being	<SIL>	myself	239	516	586	812	731

cVInt	prePhon	Vowel	nexPhon	VDur	locInWord	locInUtt	F1	F2
25	r	iy	d	174	word-medial	utterance-medial	340	1998
222	t	iy	n	59	word-medial	utterance-medial	282	2177
243	s	iy	m	83	word-medial	utterance-medial	436	2320
274	m	iy	n	122	word-medial	utterance-medial	399	798
313	r	iy	s	68	word-final	utterance-medial	306	2042
339	r	iy	aa	66	word-final	utterance-medial	363	2062
428	p	iy	p	103	word-medial	utterance-medial	316	2128
435	l	iy	n	119	word-final	utterance-medial	376	2340
457	l	iy	ch	75	word-final	utterance-medial	332	2133
476	p	iy	p	98	word-medial	utterance-medial	305	2310
489	d	iy	ih	151	word-medial	utterance-medial	303	1981
515	l	iy	ng	42	word-medial	utterance-medial	401	1934
532	d	iy	ah	124	word-medial	utterance-medial	362	1931
572	s	iy	l	415	word-final	utterance-medial	352	1661
685	th	iy	ng	109	word-medial	utterance-medial	347	1748
759	h	iy	ih	101	word-medial	utterance-medial	324	2077

그림 15. 모음 포먼트 측정 결과

Figure 15. Result from vowel formant measurements

3.4 단어 출현 빈도수 측정 결과

벅아이 코퍼스에는 레이블 기호를 포함하여 총 355,245개의 단어 혹은 토큰이 존재하고 있었고 중복 출현하는 토큰의 빈도수를 고려하면, 단어 혹은 토큰 유형은 총 13,704개였다. 13,000여개의 단어 유형별 빈도수를 로그축을 이용하여 나타내면 <그림 16>과 같다.

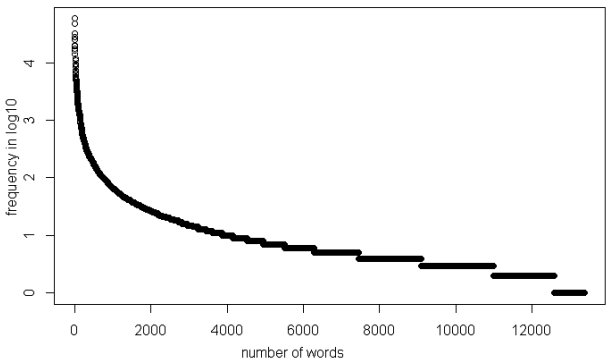

그림 16. 벅아이 코퍼스 단어의 빈도수

Figure 16. Frequency plot for words in the Buckeye Corpus

<그림 17>에서는 벅아이 코퍼스에 존재하는 단어를 가장 자주 출현하는 토큰 순서대로 나열한 출력 파일의 일부를 나타내고 있다. 가장 빈번하게 나타나는 토큰은 휴지기를 나타내는 레이블 기호인 <SIL>로 총 27,042번 반복되어 나타나고, 가장 빈번하게 나타나는 영어 단어는 인칭대명사인 I로 나타났다. 그 다음으로는 접속사 and와 정관사 the 등 기능어들 대부분이 그 뒤를 따랐다.

NUMBER	TYPE	FREQUENCY	NUMBER	TYPE	FREQUENCY
1	sil	27042	29	my	1863
2	vocnoise	22144	30	for	1816
3	i	12258	31	know	1788
4	and	10947	32	think	1773
5	iver	9751	33	he	1708
6	the	8338	34	we	1707
7	to	6410	35	there	1634
8	a	6136	36	do	1617
9	that	5820	37	on	1549
10	it	4583	38	or	1540
11	of	4350	39	not	1511
12	you	4317	40	mean	1499
13	like	4064	41	be	1497
14	uh	3594	42	with	1465
15	yknow	3591	43	because	1454
16	in	3577	44	that's	1366
17	they	3514	45	what	1305
18	was	3062	46	well	1302
19	but	2954	47	people	1301
20	it's	2833	48	really	1301
21	so	2701	49	i'm	1246
22	yeah	2651	50	all	1226
23	just	2446	51	if	1221
24	um	2435	52	this	1167
25	don't	2226	53	out	1161
26	have	2115	54	at	1153
27	laugh	1906	55	get	1133
28	is	1879	56	about	1112

그림 17. 벅아이 코퍼스 단어의 빈도수에 따른 출력

Figure 17. Buckeye Corpus words and their frequency

3.5 발화속도 측정 결과

초당 음절수로 정의된 발화속도를 계산하기에 앞서 우선 CMU 사전의 모든 항목에 대하여 음절수를 계산한 데이터베이스를 구축하였다. <그림 18>에서 보듯 강세 종류에 따른 수치를 합하여 해당 단어의 음절수로 저장하였다. 이 중에서 첫째와 마지막 컬럼 즉, 단어 리스트와 대응하는 음절수를 텍스트파일로 만들어 <부록 10>의 입력 파일 중의 하나로 지정한

다. 그렇게 하면 <부록 10>의 스크립트가 실행 중에 자동으로 이 데이터베이스를 참조하여 발견된 단어의 음절수를 찾게 되고 걸린 시간을 이용하여 발화속도를 자동으로 산출하게 된다.

CMU index	CMU entry	primary	secondary	no	totalNumStress
a	x0	0	0	1	1
a's	e1z	1	0	0	1
a(2)	e1	1	0	0	1
a.	e1	1	0	0	1
a.'s	e1z	1	0	0	1
a.s	e1z	1	0	0	1
aaa	trI2px0le1	1	1	1	3
aaberg	a1bX0g	1	0	1	2
aachen	a1kx0n	1	0	1	2
aaker	a1kX0	1	0	1	2
aalseth	a1lsE0T	1	0	1	2
aamodt	a1mx0t	1	0	1	2
aancor	a1nkc2r	1	1	0	2
aardema	a0rdE1mx0	1	0	2	3
aardvark	a1rdva2rk	1	1	0	2
aaron	E1rx0n	1	0	1	2
aaron's	E1rx0nz	1	0	1	2
aarons	E1rx0nz	1	0	1	2
aaronson	E1rx0nsx0n	1	0	2	3
aaronson's	E1rx0nsx0nz	1	0	2	3

그림 18. CMU 사전의 음절수 계산
Figure 18. Number of syllables for CMU dictionary entries

음절수 데이터베이스를 참조하여 입력된 단어 목록에 대하여 주변 단어를 고려한 발화속도 출력 결과가 <그림 19>에 제시되어 있다. 출력된 주변 정보는 파일이름, 화자, 성별, 연령, 단어 구간 번호, 해당 단어를 포함한 다섯 단어, 각각의 소요 시간, 각각의 음절수, 세 종류의 발화속도 등이다.

File	Subj	Gender	Age	wIntrvl	ppWord	preWord	word	nexWord	nnWord
s0101a_fy	s01	female	young	160	on	three	different	<SIL>	um
s0101a_fy	s01	female	young	168	with	three	different	physicians	<VOCNOISE>
s0101a_fy	s01	female	young	1032	um	<VOCNOISE>	different	swim	centers
s0101a_fy	s01	female	young	1037	hunter	<SIL>	different	rec	centers
s0101a_fy	s01	female	young	474	wise	is	probably	within	a
s0101a_fy	s01	female	young	528	the	westside	probably	couple	miles

durPP	durPrev	durCur	durNex	durNN	#SylPP	#SylPre	#SylCur	#SylNex	#SylNN	Rate1	Rate3	Rate5
0.165	0.19	0.597	0.597	0.597	1	1	3	0	1	5.03	2.89	2.8
0.104	0.128	0.298	0.298	0.298	1	1	3	3	0	10.07	9.67	7.11
0.352	0.37	0.287	0.287	0.287	1	0	3	1	2	10.44	4.23	4.42
0.482	0.181	0.29	0.29	0.29	2	0	3	1	2	10.36	5.26	5.22
0.23	0.084	0.247	0.247	0.247	1	1	3	2	1	12.16	10.39	7.59
0.077	0.397	0.412	0.412	0.412	1	2	3	2	1	7.28	5.73	5.26

그림 19. 발화속도 추출 결과
Figure 19. Calculation of speech rates

4. 결론

본 논문은 자연발화 음성 코퍼스인 벅아이 코퍼스에 대한 연구 저변을 넓히고 관련 연구를 촉진시키기 위한 의도를 가지고, 코퍼스의 레이블 파일을 프랏 포맷으로 변환시키는 방법을 소개하였고, 또 변환된 레이블 파일들을 이용하여 음성학 연구를 수행할 수 있는 여러 가지 프랏 스크립트 툴을 소개하였다.

레이블 포맷 변환 중에 발견된 코퍼스의 오류 유형은 단어층과 변이음층 사이의 경계 불일치 오류, 레이블 시작과 끝을 알리는 구간 레이블 명칭이 존재하지 않거나 단어 경계가 사라진 경우, 음성이 없는 구간에 레이블이 존재하는 경우 등이였고 스크립트를 이용하거나 수작업으로 이러한 오류를 다수 수정하였다.

또한 이렇게 수정된 벅아이 코퍼스 레이블 파일을 바탕으로 음성학 연구에 활용하는데 도움이 될 프랏 스크립트를 개발하여 소개하였다. 자음 연구에 활용할 수 있는 VOT 측정, 모음 연구에 활용할 수 있는 포먼트 측정, 코퍼스 내에서 각 단어의 출현 빈도수를 측정, 발화속도 측정 스크립트 등이다.

특히 VOT와 포먼트 측정에 있어서는 단순 측정값 이외에도 주변의 음성학적 환경이나 단어 혹은 문장 내의 위치와 관련된 정보 및 발화속도도 추출할 수 있어, 특정 조건에 맞는 측정값만을 취하여 분석할 수 있게 하였다. 자연발화 코퍼스는 특성상 기존의 녹음 방식에서 수행할 수 있는 통제를 가할 수 없으므로 이러한 주변 정보를 뽑아내어 간접적으로 통제하는 효과를 내도록 할 수 밖에 없다.

무엇보다도 이러한 연구 도구를 통해서 얻을 수 있는 가장 큰 장점 중의 하나는 수천 개에서 수만 개에 이르는 측정값을 컴퓨터를 통하여 자동 혹은 반자동으로 추출하게 하여 연구의 효율과 분석 결과의 신빙성을 높혀 줄 수 있다는 점이다. 인위적인 설정과 녹음에서 얻어내었던 많은 연구 결과들을, 비교적 자유롭게 발화되고 녹음된 자료를 통해 그 진위나 변이를 검증 확인하는 것이 보다 더 중요해지는 추세를 감안할 때, 본 연구에서 제시된 도구들은 그 가치가 클 수 있다고 믿는다.

물론 자연발화 음성 코퍼스가 기존의 스튜디오 녹음 자료에 비하여 완벽한 요인 통제가 어렵다는 점의 단점이 있긴 하지만, 저자가 아는 한 지금껏 수행해 왔거나 공개된 음성 코퍼스 중에서 가장 자연스러운 발화 상황에서 실제 대화 상황에서 발생하는 음성 현상을 가장 잘 나타내고 있기 때문에 그런 단점은 충분히 상쇄하고도 남을 것이라 생각한다. 또한 통제되지 않은 요인은 측정값 추출시 동시에 기록되는 주변의 음성학적 환경 등 부가 정보를 통해, 분석 시에 특정 요건에 적합한 측정치만을 분석 대상으로 삼아 간접적으로 통제할 수 있다. 끝으로 본 논문에 공개된 프랏 스크립트와 결과 파일 등은 저자의 누리집을 통해 배포하고 있다.

참고문헌

[1] Pitt, M.A., Dilley, L., Johnson, K., Kiesling, S., Raymond, W., Hume, E. and Fosler-Lussier, E. (2007). Buckeye Corpus of Conversational Speech (2nd release) [www.buckeyecorpus. osu. edu] Columbus, OH: Department of Psychology, Ohio State University (Distributor).

[2] Boersma, Paul & Weenink, David (2012). Praat: doing phonetics by computer [Computer program]. Version 5.3.04, retrieved 12 January 2012 from http://www.praat.org/

[3] Sjölander, Kåre & Beskow, Jonas. (2000). Wavesurfer - An Open Source Speech Tool.

[4] The CMU Pronouncing Dictionary, URL: http://www.speech.cs.cmu.edu/ cgi-bin/cmudict.

[5] R Development Core Team. (2008). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

• **윤규철 (Yoon, Kyuchul)**
영남대학교 영문과
경북 경산시 대동 214-1
Tel: 053-810-2130
Email: kyoon@ynu.ac.kr
관심분야: 음성학, 음운론

부 록

(아래에 제공된 스크립트의 파일 버전은 저자의 개인 홈페이지 http://ling.osu.edu/~kyoon에서 구할 수 있습니다.)

1. Xwaves 형식의 레이블 파일을 프랏의 레이블 형식으로 변환하는 프랏 스크립트

```
#########################
# xlabel2textgrid.praat #
#########################
form Specify files and folders
  word inFolder problem
  word inFileType_(with_dot) .log
  word outFileType_(with_dot) .TextGrid
  word outFolder_(to_be_created) problem.fixed
endform

system_nocheck mkdir 'outFolder$'

Create Strings as file list... fileListObj
  'inFolder$'/*'inFileType$'
numFiles = Get number of strings
pause 'numFiles' files identified. Continue?

for iFile to numFiles
  select Strings fileListObj
  fileName$ = Get string... iFile
  outFileName$ = fileName$ + outFileType$

  Read IntervalTier from Xwaves...
    'inFolder$'/'fileName$'
  Rename... xwavesIntervalTierObj
  Into TextGrid

  Write to text file... 'outFolder$'/'outFileName$'
    plus IntervalTier xwavesIntervalTierObj
    Remove
endfor
select Strings fileListObj
Remove
############ END OF SCRIPT ############
```

2.두 개의 레이블 파일을 하나로 통합하는 프랏 스크립트

```
############################
# mergeTextGridTiers.praat #
############################
form Specify files and folders
  word inWordsFolder wordsFolder
  word inWordsFileType_(with_dot) .words.TextGrid
  natural wordsTierNum 1
  word wordsTierName words
  word inPhonesFolder phonesFolder
  word inPhonesFileType_(with_dot) .phones.TextGrid
    natural phonesTierNum 2
    word phonesTierName phones
    word outFolder_(to_be_created) mergedFolder
    word outFileType_(with_dot) .TextGrid
endform

system_nocheck mkdir 'outFolder$'
Create Strings as file list... fileListObj
  'inWordsFolder$'/*'inWordsFileType$'
numFiles = Get number of strings
pause 'numFiles' files identified. Continue?

for iFile to numFiles
  select Strings fileListObj
  fileName$ = Get string... iFile
  filePrefix$ = fileName$ - inWordsFileType$
  phonesFileName$ = filePrefix$ + inPhonesFileType$
    outFileName$ = filePrefix$ + outFileType$

  Read from file... 'inWordsFolder$'/'fileName$'
    Rename... wordsFileObj
  Read from file...
    'inPhonesFolder$'/'phonesFileName$'
  Rename... phonesFileObj
  plus TextGrid wordsFileObj
  Merge
  Set tier name... 3 log
  Write to text file... 'outFolder$'/'outFileName$'
    plus TextGrid wordsFileObj
    plus TextGrid phonesFileObj
    Remove
endfor
select Strings fileListObj
Remove
############ END OF SCRIPT ############
```

3. 단어층과 변이음층 사이에 경계가 일치하지 않는 부분을 자동으로 수정하는 프랏 스크립트

```
##############################################
# correctUnsynchronizedPhoneBoundaries.praat #
##############################################
form Parameters
  word soundFolder wav
  word labelFolder label
  natural wordTier 1
  natural phoneTier 2
  real windowSize_(whole_size_in_sec) 1.5
  word fixedFolder label.synced
  word logFolder label.syncError
  choice showStep: 2
           button Yes
           button No
  choice showByFile: 2
           button Yes
           button No
  choice saveAsNewFile: 1
           button Yes
           button No
  comment Fixed data to  be output with "fixed-"
endform

system_nocheck mkdir 'fixedFolder$'
system_nocheck mkdir 'logFolder$'

Create Strings as file list... fileListObj
  'labelFolder$'/*.TextGrid
numFiles = Get number of strings
pause 'numFiles' files identified. Continue?

for iFile to numFiles
  select Strings fileListObj
  labelName$ = Get string... iFile
```

```
prefix$ = labelName$ - ".TextGrid"
outFile$ = "fixed-" + prefix$ + ".txt"
fileappend 'logFolder$'/'outFile$'
...file'tab$'wordInterval#
...'tab$'word'tab$'startError(+/-)'newline$'
soundName$ = prefix$ + ".wav"
Read from file... 'soundFolder$'/'soundName$'
Read from file... 'labelFolder$'/'labelName$'
numIntervals = Get number of intervals... wordTier
totalDur = Get total duration
plus Sound 'prefix$'
Edit

for iInterval from 2 to numIntervals
  select TextGrid 'prefix$'
  wordIntervalTime = Get start point...
  ...wordTier iInterval
  word$ = Get label of interval...
  ...wordTier iInterval
  left = wordIntervalTime - windowSize/2
  right = wordIntervalTime + windowSize/2
  intervalNum = Get interval at time...
  ...phoneTier wordIntervalTime
  startInterval = Get start point...
  ...phoneTier intervalNum
  endInterval = Get end point...
  ...phoneTier intervalNum
  # Find it
  diffFront = abs(startInterval-wordIntervalTime)
  diffBack = abs(endInterval-wordIntervalTime)
  if (diffFront <> 0 and diffBack <> 0)
    if diffFront < diffBack
      if showStep = 1
          pause Front smaller
      endif
      editor TextGrid 'prefix$'
        if left < 0
          Select... 0 right
          Zoom to selection
           Move cursor to... wordIntervalTime
        elsif right < 0
           Select... left totalDur
           Zoom to selection
        else
           Select... left right
           Zoom to selection
        endif
      endeditor

      fixedTime = wordIntervalTime
      select TextGrid 'prefix$'
      Insert boundary... phoneTier fixedTime
      oldLabel$ = Get label of interval...
      ...phoneTier intervalNum
      newPhoneIntervalNum = intervalNum + 1
      Set interval text...
      ...phoneTier newPhoneIntervalNum 'oldLabel$'
      Set interval text... phoneTier intervalNum
      if intervalNum <> 1
          Remove left boundary...
          ...phoneTier intervalNum
      endif
      fileappend 'logFolder$'/'outFile$'
      ...'prefix$''tab$''iInterval''tab$'
      ...'word$''tab$'-'diffFront''newline$'
    elsif diffFront > diffBack
      if showStep = 1
          pause Front bigger
      endif
      editor TextGrid 'prefix$'
          if left < 0
            Select... 0 right
            Zoom to selection
          elsif right < 0
            Select... left totalDur
            Zoom to selection
          else
            Select... left right
            Zoom to selection
            Move cursor to... wordIntervalTime
          endif
        endeditor

        fixedTime = wordIntervalTime
        select TextGrid 'prefix$'
        Insert boundary... phoneTier fixedTime
        newPhoneIntervalNum = intervalNum + 2
        if newPhoneIntervalNum <= numIntervals
           Remove left boundary...
           ...phoneTier newPhoneIntervalNum
        endif
        fileappend 'logFolder$'/'outFile$'
        ...'prefix$''tab$''iInterval''tab$'
        ...'word$''tab$'+'diffBack''newline$'
      elsif diffFront = diffBack
          # Already synced, so do nothing
      endif
    endif
  endfor
  editor TextGrid 'prefix$'
    Close
  endeditor
  if showByFile = 1
    pause Go on with next file?
  endif
  if saveAsNewFile = 1
    select TextGrid 'prefix$'
    Save as text file... 'fixedFolder$'/'labelName$'
    filedelete 'labelFolder$'/'labelName$'
  endif
  select TextGrid 'prefix$'
  plus Sound 'prefix$'
  Remove
endfor
select Strings fileListObj
Remove
############ END OF SCRIPT ############
```

4. VOT와 주변 정보를 추출하는 프랏 스크립트

```
####################
# VOT-marker.praat #
####################
form Parameters
  word targetPhone_(Buckeye_phoneset) k
  natural phoneTier_(tier_of_phones) 2
  natural wordTier_(tier_of_words) 1
  natural votTier_(tier_of_VOT) 3
  word inputSound s08
  word inputTG s08
  real windowSize_(total_size_in_sec) 1.0
  comment New folder with "-VOT" prefix to be created
  comment New result text file to be created
  comment Number of words centered around target is 3
endform

outputFolder$ = inputTG$ + "-VOT"
system_nocheck mkdir 'outputFolder$'

outputFile$ = "result-" + targetPhone$ + ".txt"
fileappend 'outputFile$' file'tab$'intervalPhone
...'tab$'intervalWord'tab$'targetPhone'tab$'VOT
...'tab$'wordLocation'tab$'prevPhoneS'tab$'
...prevWord'tab$'targetWord'tab$'nextWord'tab$'
...prevDur'tab$'targetDur'tab$'nextDur'newline$'

Create Strings as file list...
...fileListObj 'inputSound$'/*.wav
numFiles = Get number of strings
pause 'numFiles' identified. Continue?

for iFile to numFiles
```

```
select Strings fileListObj
soundName$ = Get string... iFile
prefix$ = soundName$ - ".wav"
tgName$ = prefix$ + ".TextGrid"
Read from file... 'inputSound$'/'soundName$'
Read from file... 'inputTG$'/'tgName$'
numIntervals = Get number of intervals... phoneTier
totalDur = Get total duration
Insert interval tier... votTier VOT

intervalCount = 0
while (intervalCount < numIntervals)
  intervalCount = intervalCount + 1
  intervalText$ = Get label of interval...
  ...phoneTier intervalCount
  startPhone = Get start point...
  ...phoneTier intervalCount

  if intervalText$ = targetPhone$
    # Get the previous phone label
    prevPhone$ = Get label of interval...
    ...phoneTier (intervalCount-1)
    # Mark the VOT
    start = Get start point...
    ...phoneTier intervalCount
    end = Get end point... phoneTier intervalCount
    dur = end - start
    # Get interval number of word tier and its info
    intervalWord = Get interval at time... wordTier start
    wordLabel$ = Get label of interval...
    ...wordTier intervalWord
    startWord = Get start point... wordTier intervalWord
    endWord = Get end point... wordTier intervalWord
    durWord = endWord - startWord
    durWordMSEC = durWord*1000
    # Get the info of the previous word
    prevLabel$ = Get label of interval...
    ...wordTier (intervalWord-1)
    startPrev = Get start point...
    ...wordTier (intervalWord-1)
    endPrev = Get end point...
    ...wordTier (intervalWord-1)
    durPrev = endPrev - startPrev
    durPrevMSEC = durPrev*1000
    # Get the info of the next word
    nextLabel$ = Get label of interval...
    ...wordTier (intervalWord+1)
    startNext = Get start point...
    ...wordTier (intervalWord+1)
    endNext = Get end point...
    ...wordTier (intervalWord+1)
    durNext = endNext - startNext
    durNextMSEC = durNext*1000

    # Check word-initial or word-medial
    # If start of word interval equals start of phone
    # interval, then initial
    if startWord = startPhone
         wordLoc$ = "initial"
    else
         wordLoc$ = "medial"
    endif

    # Check if previous phone is [s]
    if prevPhone$ = "s"
         prevS$ = "YES"
    else
         prevS$ = "no"
    endif
    # Window size to display
    leftOfWindow = start - windowSize/2
    if leftOfWindow < 0
         leftOfWindow = 0
    endif
    rightOfWindow = start + windowSize/2
    if rightOfWindow > totalDur
         rightOfWindow = totalDur
    endif
    select Sound 'prefix$'
    plus TextGrid 'prefix$'
    Edit
    # VOT
    editor TextGrid 'prefix$'
         Zoom... leftOfWindow rightOfWindow
         Select... start end
         pause Select VOT. Otherwise just leave it
         startMarked = Get start of selection
         endMarked = Get end of selection
         durMarked = endMarked - startMarked
         votMSEC = durMarked*1000
         if (durMarked <> dur and durMarked
         ...<> 0 and durMarked <> durWord)
           Add on tier 'votTier'
         endif
    endeditor
    select TextGrid 'prefix$'
    if (durMarked <> dur and durMarked <> 0
    ...and durMarked <> durWord)
         votInterval = Get interval at time...
         ...votTier startMarked
         Set interval text... votTier
         ...votInterval 'targetPhone$'
         Save as text file...
         ...'outputFolder$'/'tgName$'
         # Output the info.
         fileappend 'outputFile$' 'prefix$''tab$'
         ...'intervalCount''tab$''intervalWord'
         ...'tab$''intervalText$''tab$''votMSEC:0'
         ...'tab$''wordLoc$''tab$''prevS$''tab$'
         ...'prevLabel$''tab$''wordLabel$''tab$'
         ...'nextLabel$''tab$''durPrevMSEC:0''tab$'
         ...'durWordMSEC:0''tab$''durNextMSEC:0'
         ...'newline$'
    endif
    editor TextGrid 'prefix$'
         Close
    endeditor
   endif
  endwhile
endfor
####### END OF SCRIPT #######
```

5. 모음 포먼트와 주변 정보를 추출해주는 프랏 스크립트

```
############################
# vowelExtractor-v8.praat #
############################
form Parameters
  word targetVowel_(BuckeyePhoneset) uh
  natural phoneTier_(tier_of_phones) 2
  natural wordTier_(tier_of_words) 1
  word inputSound_(wav_files) wav
  word inputLabel_(TextGrid_files) label
  real windowSize_(total_display_size_in_sec) 1.0
  comment Result text file: "result-"+targetVowel$.txt
  comment Processed labels to be moved to "processed" folder
  choice showByVowel_(whether_to_show_steps_by_vowel): 1
          button No
          button Yes
  choice showByFile_(whether_to_show_steps_by_file): 1
          button No
          button Yes
  choice showInfo_(whether_to_show_Praat_info_window): 1
          button No
          button Yes
endform

# Create a folder where processed label files are moved
processed$ = "processed"
path$ = inputLabel$ + "\" + processed$
system_nocheck mkdir 'path$'
```

```
# Clean the info window. For debuggin purpose
if showInfo = 2
          clearinfo
endif
outputFile$ = "result-" + targetVowel$ + ".txt"

# Print header info
fileappend 'outputFile$' file'tab$'subject'tab$'sex'tab$'
...age'tab$'curWordInt'tab$'ppWord'tab$'prevWord'tab$'
...curWord'tab$'nexWord'tab$'nnWord'tab$'ppWordDur'tab$'
...prevWordDur'tab$'curWordDur'tab$'nexWordDur'tab$'
...nnWordDur'tab$'ppNumSyl'tab$'prevNumSyl'tab$'curNumSyl
...'tab$'nexNumSyl'tab$'nnNumSyl'tab$'conentWord'tab$'
...curVowInt'tab$'prevPhone'tab$'vowel'tab$'nexPhone
...'tab$'vowDur'tab$'sylStress'tab$'locInWord'tab$'
...locInUtt'tab$'speechRate'tab$'wordFreq'tab$'F1'tab$'
...F2'newline$'

# For script debugging
if showInfo = 2
  printline file'tab$'subject'tab$'sex'tab$'
  ...age'tab$'curWordInt'tab$'ppWord'tab$'prevWord'tab$'
  ...curWord'tab$'nexWord'tab$'nnWord'tab$'ppWordDur'tab$'
  ...prevWordDur'tab$'curWordDur'tab$'nexWordDur'tab$'
  ...nnWordDur'tab$'ppNumSyl'tab$'prevNumSyl'tab$'curNumSyl
  ...'tab$'nexNumSyl'tab$'nnNumSyl'tab$'conentWord'tab$'
  ...curVowInt'tab$'prevPhone'tab$'vowel'tab$'nexPhone
  ...'tab$'vowDur'tab$'sylStress'tab$'locInWord'tab$'
  ...locInUtt'tab$'speechRate'tab$'wordFreq'tab$'F1'tab$'
  ...F2'newline$'
endif

# Make a list of sound files
Create Strings as file list...
...fileListObj 'inputLabel$'/*.TextGrid
numFiles = Get number of strings
pause 'numFiles' identified. Continue?

# Loop through each sound file
for iFile to numFiles
  select Strings fileListObj
  labelName$ = Get string... iFile
  prefix$ = labelName$ - ".TextGrid"
  # Get the speaker ID, sex, age
  subject$ = left$(prefix$,3)
  sexType$ = mid$(prefix$,8,1)
  ageType$ = mid$(prefix$,9,1)
  if sexType$ = "m"
          sex$ = "male"
  else
          sex$ = "female"
  endif
  if ageType$ = "y"
          age$ = "young"
  else
          age$ = "old"
  endif

  # Set the maximu formant in Editor Window
  if sex$ = "male"
          maxFormantHz = 5000
  else
          maxFormantHz = 5500
  endif

  # Read the files
  soundName$ = prefix$ + ".wav"
  Read from file... 'inputSound$'/'soundName$'
  Read from file... 'inputLabel$'/'labelName$'
  numPhoneInt = Get number of intervals... phoneTier
  numWordInt = Get number of intervals... wordTier
  totalDur = Get total duration
  plus Sound 'prefix$'
  Edit

  # Loop through each phone tier interval
for iPhoneInt to numPhoneInt
  # Flag for target vowel. For debugging purpose
  flagFound = 0
  select TextGrid 'prefix$'
  phoneLabel$ = Get label of interval...
  ...phoneTier iPhoneInt

  # See if it's the target vowel
  if phoneLabel$ = targetVowel$
    flagFound = 1
    # Get the vowel duration
    vowelStart = Get start point... phoneTier iPhoneInt
    vowelEnd = Get end point... phoneTier iPhoneInt
    vowelDur = (vowelEnd - vowelStart)*1000
    vowelLabel$ = phoneLabel$
    prevVowelLabel$ = Get label of interval...
    ...phoneTier (iPhoneInt-1)
    nexVowelLabel$ = Get label of interval...
    ...phoneTier (iPhoneInt+1)

    # Decide the display size
    leftOfDisplay = vowelStart - windowSize/2
    rightOfDisplay = vowelStart + windowSize/2

    # Point of F1, F2 measurement
    centerOfVowel = vowelStart + (vowelEnd-vowelStart)/2
    # Target word containing the target vowel
    wordInterval = Get interval at time...
    ...wordTier centerOfVowel

    # Current word
    wordStart = Get start point... wordTier wordInterval
    wordEnd = Get end point... wordTier wordInterval
    wordDur = (wordEnd - wordStart)*1000
    wordLabel$ = Get label of interval...
    ...wordTier wordInterval

    # Previous previous word
    # Only if it's not at the beginning of file
    if (wordInterval-2) = 0
         ppWordDur = 0
         ppWordLabel$ = "<INTERVAL_NUMBER_IS_ZERO>"
    else
         ppWordStart = Get start point...
         ...wordTier (wordInterval-2)
         ppWordEnd = Get end point...
         ...wordTier (wordInterval-2)
         ppWordDur = (ppWordEnd - ppWordStart)*1000
         ppWordLabel$ = Get label of interval...
         ...wordTier (wordInterval-2)
    endif

    # Previous word
    prevWordStart = Get start point...
    ...wordTier (wordInterval-1)
    prevWordEnd = Get end point...
    ...wordTier (wordInterval-1)
    prevWordDur = (prevWordEnd - prevWordStart)*1000
    prevWordLabel$ = Get label of interval...
    ...wordTier (wordInterval-1)

    # Next word
    nexWordStart = Get start point...
    ...wordTier (wordInterval+1)
    nexWordEnd = Get end point... wordTier (wordInterval+1)
    nexWordDur = (nexWordEnd - nexWordStart)*1000
    nexWordLabel$ = Get label of interval...
    ...wordTier (wordInterval+1)

    # Next next word
    # Only if it's not at the end of file
    if (wordInterval+2) > numWordInt
         nnWordDur = 0
         nnWordLabel$ =
         ..."<INTERVAL_NUM_EXCEEDS_ACTUAL_NUM_OF_INTVL>"
    else
```

```
      nnWordStart = Get start point...
      ...wordTier (wordInterval+2)
      nnWordEnd = Get end point...
      ...wordTier (wordInterval+2)
      nnWordDur = (nnWordEnd - nnWordStart)*1000
      nnWordLabel$ = Get label of interval...
      ...wordTier (wordInterval+2)
endif

# Decide word-initial/-medial/-final
if vowelStart = wordStart
      locInWord$ = "word-initial"
elsif vowelEnd = wordEnd
      locInWord$ = "word-final"
else
      locInWord$ = "word-medial"
endif

# Decide utt-initial/-medial/-final
if locInWord$ = "word-initial"
      if startsWith(prevWordLabel$,"{B_TRANS") = 1
        locInUtt$ = "utternace-initial"
      elsif startsWith(prevWordLabel$,"<VOCNOISE") = 1
        locInUtt$ = "utternace-initial"
      elsif startsWith(prevWordLabel$,"<IVER") = 1
        locInUtt$ = "utternace-initial"
      elsif startsWith(prevWordLabel$,"<SIL") = 1
        locInUtt$ = "utternace-initial"
      endif
elsif locInWord$ = "word-final"
      if startsWith(nexWordLabel$,"{E_TRANS") = 1
        locInUtt$ = "utternace-final"
      elsif startsWith(nexWordLabel$,"<VOCNOISE") = 1
        locInUtt$ = "utternace-final"
      elsif startsWith(nexWordLabel$,"<IVER") = 1
        locInUtt$ = "utternace-final"
      elsif startsWith(nexWordLabel$,"<SIL") = 1
        locInUtt$ = "utternace-final"
      endif
elsif locInWord$ = "word-medial"
      locInUtt$ = "utterance-medial"
endif

# Display and measure F1, F2 after formant setting
editor TextGrid 'prefix$'
      # Adjust the formant setting
      Formant settings... maxFormantHz 5 0.025 30 1
      Select... vowelStart vowelEnd
      Zoom... leftOfDisplay rightOfDisplay
      Move cursor to... centerOfVowel
      f1 = Get first formant
      f2 = Get second formant
endeditor

# Output all the info
fileappend 'outputFile$' 'prefix$''tab$''subject$'
...'tab$''sex$''tab$''age$''tab$''wordInterval'
...'tab$''ppWordLabel$''tab$''prevWordLabel$'
...'tab$''wordLabel$''tab$''nexWordLabel$''tab$'
...'nnWordLabel$''tab$''ppWordDur:0''tab$'
...'prevWordDur:0''tab$''wordDur:0''tab$'
...'nexWordDur:0''tab$''nnWordDur:0''tab$'
...'tab$''tab$''tab$''tab$''tab$''tab$''iPhoneInt'
...'tab$''prevVowelLabel$''tab$''vowelLabel$''tab$'
...'nexVowelLabel$''tab$''vowelDur:0''tab$'
...'tab$''locInWord$''tab$''locInUtt$''tab$'
...'tab$''tab$''f1:0''tab$''f2:0''newline$'

if showInfo = 2
  printline 'prefix$''tab$''subject$'
  ...'tab$''sex$''tab$''age$''tab$''wordInterval'
  ...'tab$''ppWordLabel$''tab$''prevWordLabel$'
  ...'tab$''wordLabel$''tab$''nexWordLabel$''tab$'
  ...'nnWordLabel$''tab$''ppWordDur:0''tab$'
  ...'prevWordDur:0''tab$''wordDur:0''tab$'
  ...'nexWordDur:0''tab$''nnWordDur:0''tab$'
  ...'tab$''tab$''tab$''tab$''tab$''tab$''iPhoneInt'
  ...'tab$''prevVowelLabel$''tab$''vowelLabel$''tab$'
  ...'nexVowelLabel$''tab$''vowelDur:0''tab$'
  ...'tab$''locInWord$''tab$''locInUtt$''tab$'
  ...'tab$''tab$''f1:0''tab$''f2:0''newline$'
 endif

 # Show steps by vowel?
 if showByVowel = 2
      if flagFound = 1
        pause Next target vowel?
      endif
 endif
 endif
endfor
editor TextGrid 'prefix$'
      Close
endeditor

# Show steps by file?
if showByFile = 2
   pause Next File?
endif
# Move processed label files
select TextGrid 'prefix$'
Save as text file... 'path$'/'labelName$'
filedelete 'inputLabel$'/'labelName$'
# Remove objects
select Sound 'prefix$'
plus TextGrid 'prefix$'
Remove
endfor
select Strings fileListObj
Remove
################ END OF SCRIPT ##################
```

6. 벅아이 코퍼스 레이블 파일에서 단어를 추출하는 프랏 스크립트

```
#######################
# wordExtractor.praat #
#######################
form Parameters
  word inputFolder label
endform

outputFile$ = "wordList.txt"
Create Strings as file list...
...fileListObj 'inputFolder$'/*.TextGrid
numFiles = Get number of strings
pause 'numFiles' identified.

for iFile to numFiles
  select Strings fileListObj
  file$ = Get string... iFile
  Read from file... 'inputFolder$'/'file$'
  numIntervals = Get number of intervals... 1
  for i to numIntervals
    intervalText$ = Get label of interval... 1 i
    fileappend 'outputFile$' 'intervalText$'
  endfor
  fileappend 'outputFile$' .'newline$'
  Remove
endfor
```

7. 벅아이 코퍼스 내 단어의 빈도수를 세는 프랏 스크립트

```
#########################################################
# tokenFrequency.praat (Written by Kyuchul Yoon)
# Given a text file, this script takes each line,
# tokenizes words by the space, removes non-letter symbols,
# lists the words and counts the token frequency of each
# word.
#########################################################
form Specify parameters
  word inFile_(with_.txt) wordList.txt
  word outFile_(to_be_created) tokenFrequency.txt
```

```
  word progressFile_(to_be_created) progress.txt
  natural progressLines_(report_every_th_line) 200
  natural progressTokens_(report_every_th_token) 500
endform

# Check the start time and print the header for the output
timeStarted$ = date$()
fileappend 'outFile$' 'timeStarted$''newline$'
fileappend 'outFile$' tokenType'tab$'tokenFreq'newline$'

# Read the file to process
Read Strings from raw text file... 'inFile$'
Rename... fileObj
numLines = Get number of strings
#pause 'numLines' lines identified. Continue?

################
### TOKENIZE ###
################
# Read all the lines of the file
totalTokenCount = 0
for iLine to numLines
  # Block for identifying the progress of the loop
  progress = iLine/progressLines
  progressFloor = floor(progress)
  diffProgress = progress - progressFloor
  if diffProgress = 0
          fileappend 'progressFile$' 'iLine'th
          ...line of 'numLines' lines'newline$'
  endif
  # Now the line tokenizing begins here
  select Strings fileObj
  lineText$ = Get string... iLine
  # Do the tokenization only if it's not the blank line
  if length(lineText$) <> 0
    # Tokenize the lineText by the space and
    # fill the array string variable rawTokenized$
    lenLineText = length(lineText$)
    indexOfSpace = index(lineText$," ")
    while (indexOfSpace <> 0)
          totalTokenCount = totalTokenCount + 1
          rawTokenized'totalTokenCount'$ =
          ...left$(lineText$,(indexOfSpace-1))
          lineText$ =
          ... right$(lineText$,(lenLineText-indexOfSpace))
          lenLineText = length(lineText$)
          indexOfSpace = index(lineText$," ")
    endwhile
    # Handle the last token
    totalTokenCount  = totalTokenCount + 1
    rawTokenized'totalTokenCount'$ = lineText$
  endif
endfor
# Now, we know the total number of tokens(totalTokenCount)
# and their identities(rawTokenized$)

##################
### PREPROCESS ###
##################
# Preprocessing. The preprocessed tokens are stored
# in an array string variable tokenized$
# Make each raw token lowercased
for iToken to totalTokenCount
  dummyRawToken$ = rawTokenized'iToken'$
  lenDummyRawToken = length(dummyRawToken$)
  # For each character of the raw token, make it lowercase
  for iChar to lenDummyRawToken
    singleChar$ = mid$(dummyRawToken$,iChar,1)
    call makeLowerCase 'singleChar$'
    # Store each lowercased character
    # in an array string variable
    lowercaseSingleChar'iChar'$ = lowercasedChar$
  endfor
  # When all the letters are in lowercase,
  # retrieve the lowercased raw token
  lowercasedRawToken$ = ""
  for iLetter to lenDummyRawToken
    dummy$ = lowercaseSingleChar'iLetter'$
    lowercasedRawToken$ = lowercasedRawToken$ + dummy$
  endfor
  # Block for identifying the progress of the loop
  progress = iToken/progressTokens
  progressFloor = floor(progress)
  diffProgress = progress - progressFloor
  if diffProgress = 0
    fileappend 'progressFile$' 'iToken'th lowercased
    ...out of 'totalTokenCount' tokens'newline$'
  endif
  # Now that we've got the lowercased version of
  # the raw token, start the actual preprocessing
  lenLowercasedRawToken = lenDummyRawToken
  alphabeticToken$ = ""
  for iChar to lenLowercasedRawToken
    singleLetter$ = mid$(lowercasedRawToken$,iChar,1)
    call deleteNonAlphabets 'singleLetter$'
    alphabeticToken$ = alphabeticToken$ + alphabeticChar$
  endfor
  # When the preprocessing is over, store the token
  # in an array string variable only if it's a word
  lenAlphabeticToken = length(alphabeticToken$)

  tokenized'iToken'$ = alphabeticToken$
endfor
#pause 'totalTokenCount' tokens lowercased. Continue?

###################
### COUNT TOKEN ###
###################
# Count the number of each token and store
# the token frequency in an array variable tokenFreq
# Initialize the total type count
typeCount = 0
# Loop through each token
for iToken to totalTokenCount
  # Block for identifying the progress of the loop
  progress = iToken/progressTokens
  progressFloor = floor(progress)
  diffProgress = progress - progressFloor
  if diffProgress = 0
    fileappend 'progressFile$' 'iToken'th processed
    ...out of 'totalTokenCount' tokens'newline$'
  endif
  token$ = tokenized'iToken'$
  # If not the first token, compare it to the processed
  # tokens, doing the counting
  if iToken = 1
    # For the first token, increase the typeCount to one
    typeCount = typeCount + 1
    # And store the processed token in an
    # array variable processedToken$
    processedToken'typeCount'$ = token$
    # The token frequency count is also one
    tokenFreq'typeCount' = 1
  # From the second token, start the comparisons
  else
    flagFoundMatch = 0
    numComparisons = 0
    # Repeat the loop until you find a match to existing types
    # & the number of comparisons fewer than the types found
    while (flagFoundMatch = 0 and numComparisons < typeCount)
      numComparisons = numComparisons + 1
      # dummy$ represents all the types found
      dummy$ = processedToken'numComparisons'$
      # Compare all the types against the target token
      if token$ = dummy$
            flagFoundMatch = 1
            # If found a match, there is no new type
            # Just increase the token frequency of the
            # existing type
            dummy = tokenFreq'numComparisons'
            tokenFreq'numComparisons' = dummy + 1
      endif
```

```
    endwhile
    # A new type found, add the new type to the array variable
    if flagFoundMatch = 0
      # Also, increase the type count
      typeCount = typeCount + 1
      processedToken'typeCount'$ = token$
      # And initialize the type count to one
      tokenFreq'typeCount' = 1
    endif
  endif
endfor
#pause 'typeCount' types found!

#######################
### PRINT FREQUENCY ###
#######################
for i to typeCount
  dummy$ = processedToken'i'$
  dummy = tokenFreq'i'
  fileappend 'outFile$' 'dummy$''tab$''dummy''newline$'
endfor
timeEnded$ = date$()
fileappend 'outFile$' 'timeEnded$''newline$'

###############################
### PROCEDURE: makeLowerCase ###
###############################
procedure makeLowerCase singleCharacter$
  # If the character is not an alphabet,
  # leave it as is. Otherwise, make it lowercase
  if (singleCharacter$ = "A" or singleCharacter$ = "a")
          lowercasedChar$ = "a"
  elsif (singleCharacter$ = "B" or singleCharacter$ = "b")
          lowercasedChar$ = "b"
  elsif (singleCharacter$ = "C" or singleCharacter$ = "c")
          lowercasedChar$ = "c"
  elsif (singleCharacter$ = "D" or singleCharacter$ = "d")
          lowercasedChar$ = "d"
  elsif (singleCharacter$ = "E" or singleCharacter$ = "e")
          lowercasedChar$ = "e"
  elsif (singleCharacter$ = "F" or singleCharacter$ = "f")
          lowercasedChar$ = "f"
  elsif (singleCharacter$ = "G" or singleCharacter$ = "g")
          lowercasedChar$ = "g"
  elsif (singleCharacter$ = "H" or singleCharacter$ = "h")
          lowercasedChar$ = "h"
  elsif (singleCharacter$ = "I" or singleCharacter$ = "i")
          lowercasedChar$ = "i"
  elsif (singleCharacter$ = "J" or singleCharacter$ = "j")
          lowercasedChar$ = "j"
  elsif (singleCharacter$ = "K" or singleCharacter$ = "k")
          lowercasedChar$ = "k"
  elsif (singleCharacter$ = "L" or singleCharacter$ = "l")
          lowercasedChar$ = "l"
  elsif (singleCharacter$ = "M" or singleCharacter$ = "m")
          lowercasedChar$ = "m"
  elsif (singleCharacter$ = "N" or singleCharacter$ = "n")
          lowercasedChar$ = "n"
  elsif (singleCharacter$ = "O" or singleCharacter$ = "o")
          lowercasedChar$ = "o"
  elsif (singleCharacter$ = "P" or singleCharacter$ = "p")
          lowercasedChar$ = "p"
  elsif (singleCharacter$ = "Q" or singleCharacter$ = "q")
          lowercasedChar$ = "q"
  elsif (singleCharacter$ = "R" or singleCharacter$ = "r")
          lowercasedChar$ = "r"
  elsif (singleCharacter$ = "S" or singleCharacter$ = "s")
          lowercasedChar$ = "s"
  elsif (singleCharacter$ = "T" or singleCharacter$ = "t")
          lowercasedChar$ = "t"
  elsif (singleCharacter$ = "U" or singleCharacter$ = "u")
          lowercasedChar$ = "u"
  elsif (singleCharacter$ = "V" or singleCharacter$ = "v")
          lowercasedChar$ = "v"
  elsif (singleCharacter$ = "W" or singleCharacter$ = "w")
          lowercasedChar$ = "w"
  elsif (singleCharacter$ = "X" or singleCharacter$ = "x")
          lowercasedChar$ = "x"
  elsif (singleCharacter$ = "Y" or singleCharacter$ = "y")
          lowercasedChar$ = "y"
  elsif (singleCharacter$ = "Z" or singleCharacter$ = "z")
          lowercasedChar$ = "z"
  else
          lowercasedChar$ = singleCharacter$
  endif
endproc

###################################
### PROCEDURE: deleteNonAlphabets ###
###################################
procedure deleteNonAlphabets singleCharacter$
  if singleCharacter$ = "a"
          alphabeticChar$ = "a"
  elsif singleCharacter$ = "b"
          alphabeticChar$ = "b"
  elsif singleCharacter$ = "c"
          alphabeticChar$ = "c"
  elsif singleCharacter$ = "d"
          alphabeticChar$ = "d"
  elsif singleCharacter$ = "e"
          alphabeticChar$ = "e"
  elsif singleCharacter$ = "f"
          alphabeticChar$ = "f"
  elsif singleCharacter$ = "g"
          alphabeticChar$ = "g"
  elsif singleCharacter$ = "h"
          alphabeticChar$ = "h"
  elsif singleCharacter$ = "i"
          alphabeticChar$ = "i"
  elsif singleCharacter$ = "j"
          alphabeticChar$ = "j"
  elsif singleCharacter$ = "k"
          alphabeticChar$ = "k"
  elsif singleCharacter$ = "l"
          alphabeticChar$ = "l"
  elsif singleCharacter$ = "m"
          alphabeticChar$ = "m"
  elsif singleCharacter$ = "n"
          alphabeticChar$ = "n"
  elsif singleCharacter$ = "o"
          alphabeticChar$ = "o"
  elsif singleCharacter$ = "p"
          alphabeticChar$ = "p"
  elsif singleCharacter$ = "q"
          alphabeticChar$ = "q"
  elsif singleCharacter$ = "r"
          alphabeticChar$ = "r"
  elsif singleCharacter$ = "s"
          alphabeticChar$ = "s"
  elsif singleCharacter$ = "t"
          alphabeticChar$ = "t"
  elsif singleCharacter$ = "u"
          alphabeticChar$ = "u"
  elsif singleCharacter$ = "v"
          alphabeticChar$ = "v"
  elsif singleCharacter$ = "w"
          alphabeticChar$ = "w"
  elsif singleCharacter$ = "x"
          alphabeticChar$ = "x"
  elsif singleCharacter$ = "y"
          alphabeticChar$ = "y"
  elsif singleCharacter$ = "z"
          alphabeticChar$ = "z"
  # If it's the apostophe, leave it as is
  elsif singleCharacter$ = "'"
          alphabeticChar$ = "'"
  # If it's none of the alphabets,
  # then return null string
  else
          alphabeticChar$ = ""
  endif
endproc
```

```
########## END OF SCRIPT ############
```

8. CMU dictionary에서 단어별 음절수를 찾아주는 프랏 스크립트

```
######################
# getNumOfSyls.praat #
######################
form Parameters
  word inputFile pronun.txt
  comment output file with "numSyls-"
endform

outputFile$ = "numSyls-" + inputFile$
# Print header info
fileappend 'outputFile$' entry'tab$'numPriStress
...'tab$'numSecStress'tab$'numNoStresses'tab$'
...totalNumber'newline$'

Read Table from tab-separated file... 'inputFile$'
Rename... fileObj
numRows = Get number of rows

for iRow to numRows
  select Table fileObj
  rowEntry$ = Get value... iRow PRONUN

  # Print the entry
  fileappend 'outputFile$' 'rowEntry$''tab$'

  # Initialize the numPriCount
  numPriCount = 0
  sameFlag = 0
  # Check for primary stress, which is "1"
  # and can be more than one occurrences
  while (sameFlag = 0)
          rowEntryBefore$ = rowEntry$
          rowEntry$ = replace$(rowEntry$,"1","X",1)
          if rowEntryBefore$ <> rowEntry$
                    numPriCount=numPriCount+1
          else
                    sameFlag = 1
          endif
  endwhile

  # Print number of primary stresses
  fileappend 'outputFile$' 'numPriCount''tab$'
          # Check for secondary stress, which is "2"
          # and can be more than one occurrences
          # Initialize the numSecCount
          numSecCount = 0
          sameFlag = 0
          while (sameFlag = 0)
            rowEntryBefore$ = rowEntry$
            rowEntry$ = replace$(rowEntry$,"2","X",1)
            if rowEntryBefore$ <> rowEntry$
                    numSecCount=numSecCount+1
            else
                    sameFlag = 1
          endif
  endwhile

  # Print number of secondary stresses
  fileappend 'outputFile$' 'numSecCount''tab$'
  # Check for tertiary/no stress, which is "0"
  # and can be more than one occurrences
  # Initialize the numTerCount
  numTerCount = 0
  sameFlag = 0
  while (sameFlag = 0)
          rowEntryBefore$ = rowEntry$
          rowEntry$ = replace$(rowEntry$,"0","X",1)
          if rowEntryBefore$ <> rowEntry$
                    numTerCount=numTerCount+1
          else
                    sameFlag = 1
          endif
  endwhile

  numTotal = numPriCount + numSecCount + numTerCount

  # Print number of tertiary stresses
  fileappend 'outputFile$' 'numTerCount''tab$'
  ...'numTotal''newline$'
endfor
## END OF SCRIPT ##
```

9. 단어별 음절수 DB를 이용하여 입력 단어의 음절수를 찾아주는 프랏 스크립트

```
##################
# searchDB.praat #
##################
form Parameters
  word inputFile_(list_of_words) sylnum.txt
  word dataBase_(to_be_searched) db-sylnum.txt
  word column1_(first_column_name_in_DB) word
  word column2_(second_column_name_in_DB) numSyl
  comment Output file to be created with "result-"
endform

outputFile$ = "result-" + inputFile$
Read Strings from raw text file... 'inputFile$'
Rename... wordListObj
numWords = Get number of strings

Read Table from tab-separated file... 'dataBase$'
Rename... dbObj

for iWord to numWords
  select Strings wordListObj
  word$ = Get string... iWord
  select Table dbObj
  rowNum = Search column... 'column1$' 'word$'
  targetInfo = Get value... rowNum 'column2$'
  fileappend 'outputFile$' 'word$'
  ...'tab$''targetInfo''newline$'
endfor
select Strings wordListObj
plus Table dbObj
Remove
# END OF SCRIPT #
```

10. 관찰대상 단어의 발화속도 등 정보를 추출하는 스크립트

```
############################################
# calculateSpeechRate.praat (kyoon@ynu.ac.kr)
############################################
form Parameters
  word wordList freqWordList.txt
  word sound_(original_Buckeye_Corpus) data
  word label_(original_Buckeye_Corpus) data
  word dataBase_(num_of_syllables_for_CMU) numSyl-DB.txt
  natural wordTier_(tier_for_words) 1
  comment output text file with "result-" prefix
  comment output folder for each target word to be created
endform

outFile$ = "result-" + wordList$

Read Table from tab-separated file... 'wordList$'
Rename... wordListObj
numWords = Get number of rows

Read Table from tab-separated file... 'dataBase$'
Rename... dbObj

Create Strings as file list...
...fileListObj 'label$'/*.TextGrid
numLabels = Get number of strings

# Print header info
fileappend 'outFile$' file'tab$'subject'tab$'
```

```
...gender'tab$'age'tab$'wInterval'tab$'ppWord'tab$'
...prevWord'tab$'word'tab$'nexWord'tab$'nnWord'tab$'
...durPP'tab$'durPrev'tab$'durCur'tab$'durNex'tab$'
...durNN'tab$'numSylPP'tab$'numSylPrev'tab$'numSylCur
...'tab$'numSylNex'tab$'numSylNN'tab$'speechRate1
...'tab$'speechRate3'tab$'speechRate5'newline$'

# Loop through each word
for iWord to numWords
  select Table wordListObj
  word$ = Get value... iWord TOKEN
  system_nocheck mkdir 'word$'

  # Loop through each label/sound pair
  for iLabel to numLabels
    select Strings fileListObj
    labelName$ = Get string... iLabel
    prefix$ = labelName$ - ".TextGrid"
    # Get subject-related info
    subject$ = left$(prefix$,3)
    genderAge$ = right$(prefix$,2)
    if left$(genderAge$,1) = "m"
      gender$ = "male"
    else
      gender$ = "female"
    endif
    if right$(genderAge$,1) = "y"
      age$ = "young"
    else
      age$ = "old"
    endif

    soundName$ = prefix$ + ".wav"
    Read from file... 'label$'/'labelName$'
    numWordIntervals = Get number of intervals... wordTier
    Read from file... 'sound$'/'soundName$'

    # Loop through each interval searching for target word
    for iInterval to numWordIntervals
      select TextGrid 'prefix$'
      intervalText$ = Get label of interval...
      ...wordTier iInterval

      # If found, extract info/file
      if intervalText$ = word$
           wInterval = iInterval
           # Depending on where it is
           # For the first interval
           if iInterval = 1

           # For the second interval
           elsif iInterval = 2

           # For the one before the last interval
           elsif iInterval = (numWordIntervals-1)

           # For the last interval
           elsif iInterval = numWordIntervals

           # For all the other cases
           else
             ppWord$ = Get label of interval...
             ...wordTier (iInterval-2)
             interval = iInterval-2
             call getDuration interval
             durPP = duration
             call getNumSyl 'ppWord$'
             numSylPP = numSyl

             prevWord$ = Get label of interval...
             ...wordTier (iInterval-1)
             interval = iInterval-1
             call getDuration interval
             durPrev = duration
             call getNumSyl 'prevWord$'
             numSylPrev = numSyl

             interval = iInterval
             call getDuration interval
             durCur = duration
             call getNumSyl 'word$'
             numSylCur = numSyl

             nexWord$ = Get label of interval...
             ...wordTier (iInterval+1)
             interval = iInterval+1
             durNex = duration
             call getNumSyl 'nexWord$'
             numSylNex = numSyl

             nnWord$ = Get label of interval...
              ...wordTier (iInterval+2)
             interval = iInterval+2
             durNN = duration
             call getNumSyl 'nnWord$'
             numSylNN = numSyl

             speechRate1 = numSylCur/durCur
             speechRate3 = (numSylPrev+numSylCur+numSylNex)
             .../(durPrev+durCur+durNex)
            speechRate5 = (numSylPP+numSylPrev+numSylCur+
             ...numSylNex+numSylNN)
             .../(durPP+durPrev+durCur+durNex+durNN)

             partPrefix$ = prefix$ + "_" + "'iInterval'"
             startTarget = Get start point...
             ...wordTier (iInterval-2)
             endTarget = Get end point...
             ...wordTier (iInterval+2)

             select Sound 'prefix$'
             Extract part... startTarget
              ...endTarget rectangular 1 no
             Save as WAV file... 'word$'/'partPrefix$'.wav
             Remove
             select TextGrid 'prefix$'
             Extract part... startTarget endTarget no
             Save as text file...
             ...'word$'/'partPrefix$'.TextGrid
             Remove

             # Print info
             fileappend 'outFile$' 'prefix$''tab$''subject$'
             ...'tab$''gender$''tab$''age$''tab$''wInterval'
             ...'tab$''ppWord$''tab$''prevWord$''tab$'
             ...'word$''tab$''nexWord$''tab$''nnWord$''tab$'
             ...'durPP:3''tab$''durPrev:3''tab$''durCur:3'
             ...'tab$''durNex:3''tab$''durNN:3''tab$'
             ...'numSylPP''tab$''numSylPrev''tab$'
             ...'numSylCur''tab$''numSylNex''tab$''numSylNN'
             ...'tab$''speechRate1:2''tab$''speechRate3:2'
             ...'tab$''speechRate5:2''newline$'
           endif
       endif
     endfor
     select TextGrid 'prefix$'
     plus Sound 'prefix$'
     Remove
   endfor
#  pause NEXT WORD?
endfor
select Strings fileListObj
plus Table wordListObj
plus Table dbObj
Remove

procedure getDuration intervalNum
  start = Get start point... wordTier intervalNum
  end = Get end point... wordTier intervalNum
  duration = end - start
endproc
```

```
procedure getNumSyl targetWord$
  select Table dbObj
  rowNum = Search column... word 'targetWord$'
  if rowNum <> 0
    numSyl = Get value... rowNum numSyl
  else
    numSyl = 0
  endif
  select TextGrid 'prefix$'
endproc
###### END OF SCRIPT #######
```