The Journal of the Convergence on Culture Technology (JCCT)
Vol. 5, No. 4, pp.407-412, November 30, 2019. pISSN 2384-0358, eISSN 2384-0366

http://dx.doi.org/10.17703/JCCT.2019.5.4.407
JCCT 2019-11-51

한글 단어의 음성 인식 처리에 관한 연구

A Study on Processing of Speech Recognition Korean Words

남기훈*

Kihun Nam*

요 약 본 논문에서는 한글 단어 단위의 음성 인식 처리 기술을 제안한다. 음성 인식은 마이크와 같은 센서를 사용하여 얻은 음향학적 신호를 단어나 문장으로 변환시키는 기술이다. 대부분의 외국어들은 음성 인식에 있어서 어려움이 적은 편이다. 그에 반면, 한글의 모음과 받침 자음 구성이어서 음성 합성 시스템으로부터 얻은 문자를 그대로 사용하기에는 부적절하다. 기존 구조의 음성 인식 기술을 개선해야만 보다 정확하게 단어를 인식할 수 있다. 이러한 문제를 해결하기 위해 기존 방식의 음성 인식구조에 새로운 알고리즘을 추가하여 음성 인식률을 높이게 하였다. 먼저 입력된 단어를 전처리 과정을 수행한 후 결과를 토큰 처리한다. 레벤스테인 거리 알고리즘과 해싱 알고리즘에서 처리된 결과 값을 조합한 후 자음 비교 알고리즘을 거쳐 표준 단어를 출력한다. 최종 결과 단어를 표준화 테이블과 비교하여 존재하면 출력하고 존재하지 않으면 테이블에 등록하도록 하였다. 실험 환경은 스마트폰 응용 프로그램을 개발하여 사용하였다. 본 논문에서 제안된 구조는 기존 방식에 비해 인식률의 성능이 표준어는 2%, 방언은 7% 정도 향상되었음을 보였다.

주요어 : 음성 인식, 음성 합성 시스템, 레벤스테인 거리, 해싱, 인식률

Abstract In this paper, we propose a technique for processing of speech recognition in korean words. Speech recognition is a technology that converts acoustic signals from sensors such as microphones into words or sentences. Most foreign languages have less difficulty in speech recognition. On the other hand, korean consists of vowels and bottom consonants, so it is inappropriate to use the letters obtained from the voice synthesis system. That improving the conventional structure speech recognition can the correct words recognition. In order to solve this problem, a new algorithm was added to the existing speech recognition structure to increase the speech recognition rate. Perform the preprocessing process of the word and then token the results. After combining the result processed in the Levenshtein distance algorithm and the hashing algorithm, the normalized words is output through the consonant comparison algorithm. The final result word is compared with the standardized table and output if it exists, registered in the table dose not exists. The experimental environment was developed by using a smartphone application. The proposed structure shows that the recognition rate is improved by 2% in standard language and 7% in dialect.

Key Words : Speech Recognition, STT(Speech to Text), Levenshtein Distance, Hashing, Recognition Rate

*정회원, 서경대학교 컴퓨터공학과 조교수
접수일: 2019년 9월 13일, 수정완료일자: 2019년 10월 7일
게재확정일자: 2019년 10월 15일

Received: September 13, 2019 / Revised: October 7, 2019
Accepted: October 15, 2019
*Corresponding Author: namkh@skuniv.ac.kr
Dept. Computer Engineering, SeoKyeong Univ, Korea

Ⅰ. 서 론

음성 인식 기술은 인간의 자연어 발화를 컴퓨터가 자동으로 이해하고 처리하는 알고리즘을 연구하는 분야이다. 음성 인터페이스 기능을 확장한다면 모바일을 통한 다양한 소통형 서비스가 이루어질 것으로 기대된다. 애플의 '시리(Siri)', 마이크로소프트의 '코타나(Cortana)', 구글의 '어시스턴트(Assistant)' SKT의 '누구', KT의 'GIGA지니' 등이 현재 서비스를 진행하고 있다[1].

IT 시장 조사 기관 가트너(Gartner)에서 2019년에는 스마트폰과 사용자 간의 상호 작용 중 20%가 가상 개인 비서(Virtual Personal Assistants)를 통해 이루어질 것으로 예측하였으며 많은 기기 및 IoT 장비가 누르지 않고 제어할 수 있는 제로터치 사용자 인터페이스(UI) 기반으로 작동할 것으로 전망하고 있다[2].

영어와 같은 받침이 없는 언어의 경우는 음성 인식률이 받침이 많은 한국어에 비해 정확도가 상대적으로 높은 편이다. 한국어는 형태소 분석에 기반 한 어휘 선정의 문제와 함께 인식 대상 어휘의 제한으로 인한 인식 오류 발생이 많은 편이다. 또한 표준어가 아닌 사투리(방언)의 음성도 인식하여야 하기에 한국어 특성에 적합한 음성 인식 처리 기술의 개선이 필요하다[3].

본 논문에서는 이러한 문제점을 개선하고자 한국어 단어 단위 음성 언어 처리 기술을 제안한다. 2장에서 Google의 Cloud Speech API: STT(Speech To Text)를 거쳐 나온 Text data를 입력받아 어휘 분석 과정을 기술하고 3장에서는 어휘 선정 및 표준어로 변환시키는 알고리즘들을 설명하고 4장과 5장은 실험 및 결과와 결론 순으로 구성된다.

Ⅱ. 관련 연구

음성 언어 기술은 자동적 수단에 의하여 음성으로부터 언어적 의미 내용을 식별하는 기술이다. 마이크와 같은 소리 입력 센서를 통해 얻은 음향학적 신호(Acoustic Speech Signal)를 음성 분석, 음소 인식, 단어 인식, 문장 해석, 의미 추출 등으로 분류하고. 특징을 추출하여 음성 모델 데이터베이스와 비교하는 방식으로 음성 인식을 하게 된다. 음성 분석은 단어 인식까지를 말하는 경우가 있다. 음성 인식의 목표는 자연스러운 발성에 의한 음성을 인식하여 완전한 단어 또는 문장으로 정확하게 변환시키는 것이다[4-7].

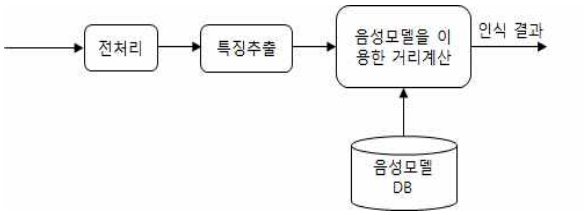

그림 1. 음성 인식 기술의 구조

Fig 1. Structure of speech recognition technology

그림 1은 음성 인식 기술의 기본적인 원리 구조이다[8]. 음성 인식률을 높이기 위해 그림 1의 구조를 개선시켰다.

단어 단위의 음성을 입력 받아 처리하는 방식으로 입력 결과는 한 음씩 명사에 대한 처리로 진행한다.

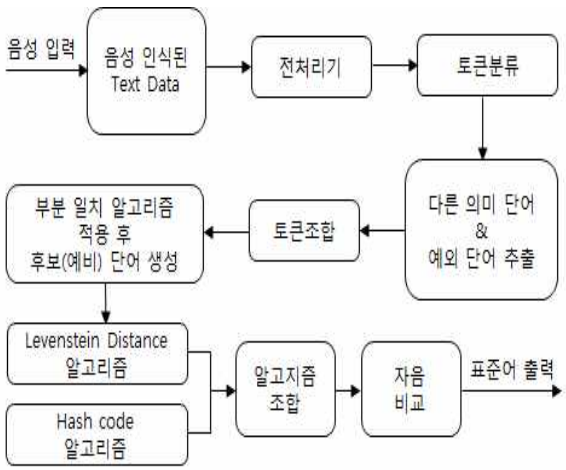

그림 2. 제안하는 음성 인식 기술

Fig 2. Proposal of speech recognition technology

그림 2에서의 전처리 과정은 입력 값에 띄어쓰기가 있는지를 확인하고 띄어쓰기가 있을 시에는 입력 단어를 분류하는 토큰 과정을 거친다. 분류와 동시에 다른 의미 단어가 있는지와 예외 단어의 여부를 동시에 확인한다.

다른 의미 단어 확인은 입력받은 값이 어떤 단어의 다른 의미인 경우 해당하는 원래 단어를 저장하고 아닐 경우는 무시한다. 입력 값에 다른 의미(표준어, 사투리) 테이블의 컬럼과 일치하는 단어가 있는지를 확인한다. 만약 일치하는 단어가 있을 경우 해당하는 원래 의미 단어를 저장하여 후보 단어 필터링에 표준어

확인 단계에서 표준어로 바꾼 후 표준 단어로 출력된다. 예외 단어가 있는지 확인하고 있을 경우, STT의 DB내에 존재하지 않는 단어를 예외(표준어, 사투리)테이블의 단어와 비교하여 원래의 단어로 변경한다. 예외 처리는 그림 3과 같이 입력이 예외(표준어, 사투리) 테이블내의 예외 결과 컬럼과 비교하여 일치하는 경우 해당하는 원래 단어로 변환하여 출력한다.

토큰 조합은 토큰화된 단어가 다른 의미, 예외처리, 알파벳 표기 변환을 거쳐 변경 되었을 때 표1과 같이 각각 변경된 입력 결과의 순서를 기준으로 변경된 단어를 조합하고 띄어쓰기와 중복된 단어는 삭제함으로써 후보 단어를 만든다.

표 1. 토큰 조합 예
Table 1. Example of token combination

입력 값	{일칠 차}, {하나일곱 차}, {십칠 차}, {열일곱 차}, {17 차}
토큰 조합	{일칠}, {차}, {하나일곱}, {차}, {십칠}, {차}, {열일곱}, {차}, {17}, {차}, {일칠차}, {하나일곱차}, {십칠차}, {열일곱차}, {17차}
중복 제거	{일칠}, {하나일곱}, {십칠}, {열일곱}, {17}, {차}, {일칠차}, {하나일곱차}, {십칠차}, {열일곱차}, {17차}

이때 글자 길에 따라 초기 근사치 값을 생성하는데 "(입력받은 글자 수) - (후보입력결과 글자 수) = 초기 근사치" 같은 방법으로 생성하면 이것을 Score라 부른다. 초기 근사치 점수는 해당하는 후보 입력단과 함께 저장한다. 단, 띄어쓰기가 없어 토큰 분류 과정을 거치지 않는 경우 초기 근사치는 0이다.

III. 새로운 알고리즘을 추가로 적용한 음성 인식 기술

후보 단어 선정에 앞서 전처리 단어는 1개 혹은 그 이상일 수 있다. 후보 단어는 부분 일치 확인을 통해 최종 입력 단어와 부분 또는 모두 일치하는 단어를 찾아 초기 후보 단어를 생성한다. 만약 초기 후보 단어가 있으며 그 단어 내에서 후보 단어를 선출하고 초기 후보 단어가 없으면 최종 입력 단어를 이용해 후보 단어를 선출한다. 후보 단어 선출 방법은 앞서 전처리 때 만들어진 초기 근사치 값이 0일 경우에 Hashing과 Levenshtein Distance 알고리즘의 결과로 나온 단어들을 매칭시켜 같은 단어가 있을 경우 그 단어의 근사치에 -1 값을 더해주고 만약 0이 아닐 경우 Hashing 알고리즘은 사용하지 않으며 근사치 값을 그대로 저장한다. 그 후 근사치 비교 과정을 거쳐 후보 단어의 수를 줄이고 만약 후보 단어의 수가 2개 이사일 경우 자음 비교를 거쳐 한번 더 후보 단어의 수를 줄임으로써 최종 후보 단어를 선출한다.

부분 일치 확인은 그림 3에서 최종 입력 단어가 표준어 테이블, 사투리 테이블의 단어 내에 부분적 또는 모두 일치하는 단어를 색출한다. 이때 부분 일치하는 단어가 있을 경우 true 값을 전달하고 없을 경우 false 값을 전달한다.

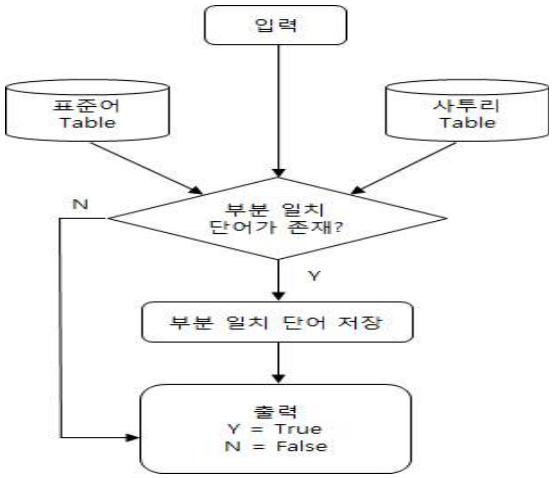

그림 3. 부분 일치 알고리즘
Fig 3. Algorithm of partial accordance

Levenshtein Distance 알고리즘은 표준어 테이블과 사투리 테이블에 존재하는 단어들 중 어떤 단어가 전처리를 거친 입력 단어와 얼마나 유사한지 검사하는 알고리즘으로 그림 4와 같이 후보 단어를 색출한다[9][10]. 이때 입력 받은 단어와 같아지기 위해서 몇 번의 글자 변경이 일어나야 하는지 기록한 값을 근사치로 정한다. 근사치는 그 값이 적을 수 록 가장 근사함을 의미하며 Levenshtein Distance 알고리즘을 통해 얻은 예비 단어의 근사치에 해당 입력 단어의 초기 근사치 값을 더한다. 후보 선정에 있어서 근사치 값이 5 이하일 경우에만 후보에 올리고 각 후보 단어의 근사치에 앞서 전처리를 거친 각각의 입력 단어가 가지는 근사치를 더한다.

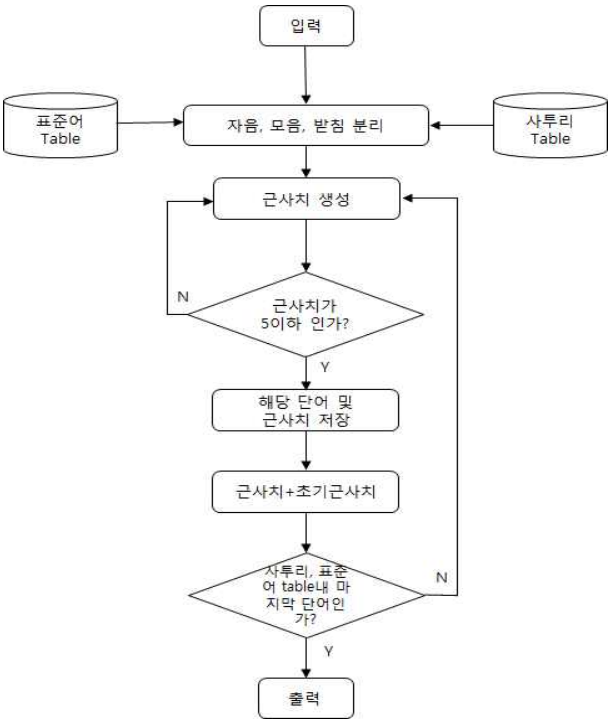

그림 4. Levnsthtein distance 알고리즘

Fig 4. Algorithm of Levnsthtein distance

Hashing Algorithm은 그림 5와 같은 형태로 실행된다. 먼저, 발음 유사도를 기준으로 만든 그림 6의 가중치 테이블을 기준으로 입력단어의 hash code를 구하여 표준어 테이블과 사투리 테이블에 존재하는 단어들의 hash code값과 비교하여 같은 값을 가지는 단어를 색출한다. 원래 입력 단어와 글자 수가 같을 경우이면서 숫자가 포함되지 않은 경우에만 실행한다.

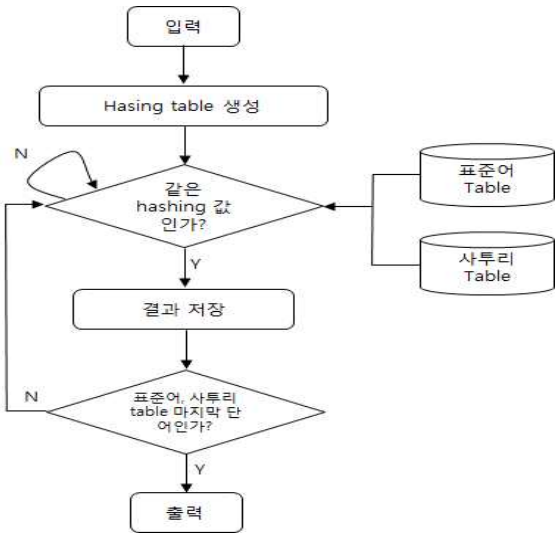

그림 5. Hashing 알고리즘

Fig 5. Algorithm of hashing

그림 6. 가중치 테이블

Fig 6. Table of weighted

두 알고리즘의 조합은 그림 7과 같이 Hashing 알고리즘으로 인해 나온 결과와 Leveshtein Distance 알고리즘에 나온 결과 단어 중 같은 단어가 있을 때 전처리를 거친 입력 단어와 글자 수가 같은 경우에만 그 단어의 근사치 값에 -1을 더해준다.

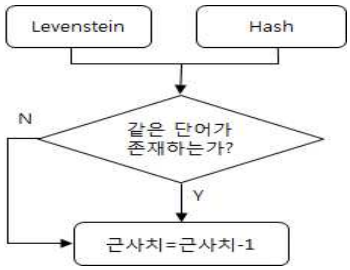

그림 7. 두 알고리즘의 조합

Fig 7. Combination of two algorithms

근사치 비교는 그림 8과 같이 각 후보 단어에 근사치를 비교하여 가장 적은 근사치 값을 가지는 후보 단어를 최종 결과로 저장한다. 이때 같은 근사치를 갖는 단어도 함께 저장한다.

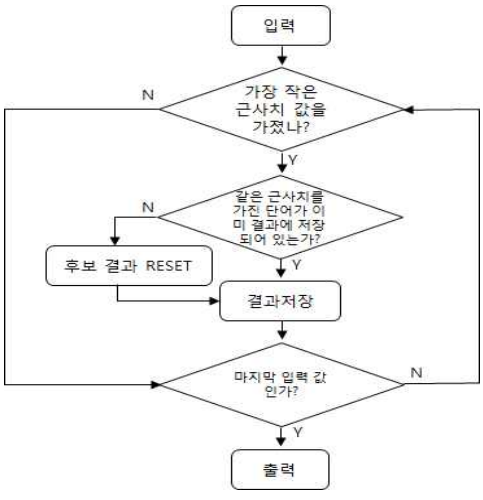

그림 8. 근사치 비교 알고리즘

Fig 8. Algorithm of approximate comparison

자음 비교는 근사치를 비교한 두에도 단어가 2개 이상일 경우 그림 9와 같이 최종 입력 단어의 자음 순서를 비교하여 가장 비슷한 단어를 최종 결과로 저장한다. 단, 원래 단어의 글자 수와 같지 않은 입력단어의 후보 단어일 경우 자음의 순서가 모두 같아야만 후보 결과에 저장된다.

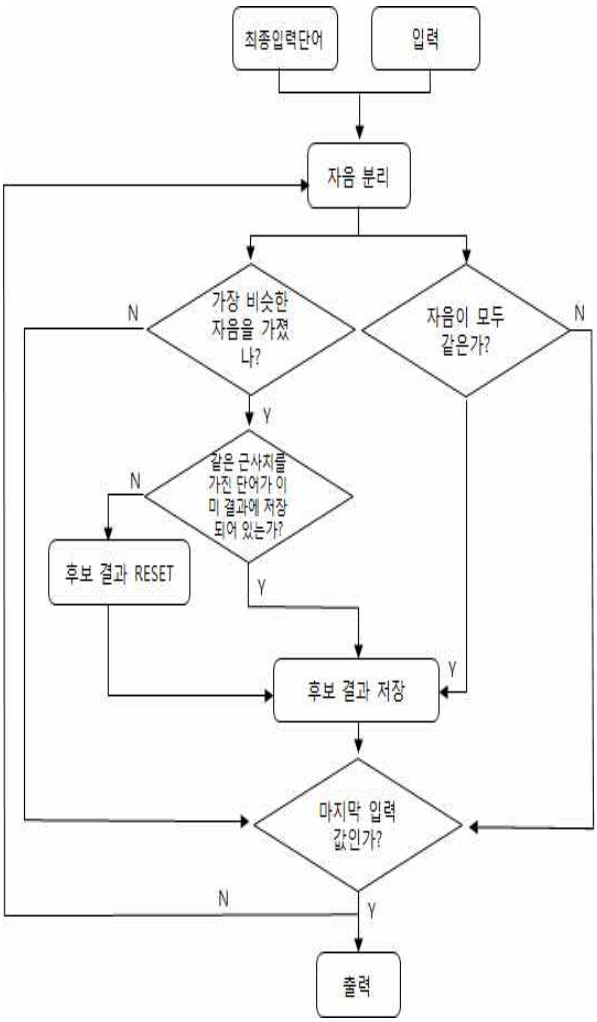

그림 9. 자음 비교 알고리즘
Fig 9. Algorithm of consonant comparison

최종 후보 필터링에서 표준어 확인 그림 10과 같이 최종 결과 단어와 전처리 단계에서 찾아낸 다른 의미 단어의 원래 단어 중 SRD 테이블과 사투리 컬럼과 일치하는 단어가 있는지 확인한다. 만약 일치하는 단어가 있을 겨우 해당하는 표준어를 최종 결과 단어와 함께 출력하고 없을 경우 최종 결과 단어만 출력한다.

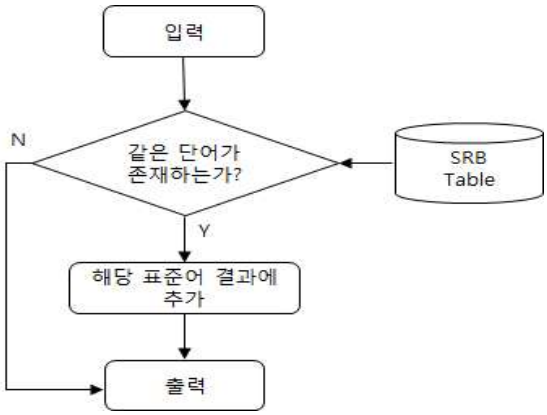

그림 10. 표준어 확인 알고리즘
Fig 10. Algorithm of confirm standard word

IV. 실험 및 결과

단어 단위 실험을 하기위한 테스트 앱 UI를 표2와 같은 환경에서 개발하였다. 사용법은 마이크에 3번 발화를 통해 테스트 단어와 동일하지 않을 경우 저장 및 3번 이내에 나오지 않은 단어들을 확인한다.

표 2. 개발 환경
Table 2. Environment of Development

제작도구	내용
Android Studio 3.2	Android 프로그래밍 개발 환경
Samsung Note9	프로그래밍 어플을 적용한 디바이스
Android 9, APPI 28	Android AVD

테스트 앱을 통해 원본 단어는 표준 테이블과 사투리 테이블에 존재하는 단어를 하나씩 출력하고 원본 단어를 읽은 결과를 출력한다. 처리결과 부분에서는 저장 또는 저장취소 버튼을 누른 후 잘 처리되었는지를 확인하며 마이크 버튼을 눌러 사용자 음성을 입력받는다.

표 3. 표준어와 불일치된 단어
Table 3. Mismatches between standard and word

단어	결과	비고
나노	가나초콜릿->가나쵸콜릿	부분일치
딸기썬대	딸기선테이, 딸기, 선데이->딸기	
밤팥밥	단팥빵(사전에 존재)	
	밤밥(사전에 존재)	
살치찌개	참치찌개(사전에 존재)	

표 4. 사투리와 불일치된 단어
Table 4. Mismatches between standard and dialect

단어	결과	원래단어	비고
댕추	땡큐->땅콩	고추	자음일치
따리	다리->가다리->가다랭이	딸기	부분일치
묶	묵->어묵	무	부분일치
새구	삼구, 3구->살구	살구	
오앳	오예->오이	자두	
…	…	…	…
콩기럼	콩기름(사전에 존재)	콩나물	
뭉우	무무->열무무침	무	부분일치
무짐치	무김치->열무김치	깍두기	부분일치

표준어는 1578개 단어 중 4개가 불일치하여 약 99.7%의 적중률을 보였으며 사투리는 661개 단어 중 21개가 불일치하여 약 97%의 음성 인식 적중률을 보였다.

V. 결 론

본 논문에서 한글 단어의 음성 인식 기술을 구현하였다. 기존의 음성 인식 시스템보다 제안한 구조의 인식률이 높았으며 사투리(방언) 단어의 인식률도 개선되었다. 통상적인 구조에서는 표준 단어의 경우는 97%의 인식률을 99.7%까지 향상되었으며 사투리의 경우는 91%에서 97%로 인식률이 개선되었다[11]. 표준어는 대략 50만개의 단어들을 데이터베이스 구축하고 서비스를 제공하고 있는 시스템으로 비교하기에는 물리적 시간적 제약이 따르기에 상대적으로 적은 약 2천개의 단어를 가지고 샘플 비교한 결과이다. 그러나 안정적으로 음성 인식을 하기위해 세 번 발화를 하여야하는 사용자의 불편한 점이 문제로 남는다. 차후 연구로는 3회에 걸친 발화 방식을 1회 발화로 음성을 정확하게 인식할 수 있는 방법을 개발하는 것을 목표로 두고 있다.

References

[1] Kyung Nim Lee, “Speech language processing technology, how far”, National Institute of Korean Language New Language Life, Vol. 27, No. 4, pp. 99-116, 2017.
[2] https://www.gartner.com/en/newsroom/press-releases/2019-01-09-gartner-predicts-25-percent-of-digital-workers-will-u
[3] G. Hinton et al., "Deep Neural Networks for Acoustic Modeling in Speech Recognition", The IEEE Signal Processing Magazine, Vol. 29, No. 6, pp. 18-27, 2012.
[4] Thomas F. Quatieri, "Discrete-Time Speech Signal Processing Principles and Practice", Prentice Hall, 2001.
[5] Dosik Moon, "Development and Evaluation of an Englich Speaking Task Using Smartphone and Text-to Speech", The Journal of The Institute of Internet, Broadcasting and Communication(JIIBC), Vol. 16, No. 5, pp. 13-20, 2016.
Doi.org/10.7236/JIIBC.2016.16.5.13
[6] Hyeong-Joon Kwon, Tetsuo Kinoshita, "Novel Speech Web Architecture Based on Information Selection Agent", International Journal of Advanced Culture Technology(IJACT), Vol 1, No 1, pp. 11-14, 2013.
[7] Seung Joo Choi, Jong-Bae Kim, "Comparison Analysis of Speech Recognition Open API's Accuracy", Asia-pacific Journal of Multimedia Services Convergent with Art, Humanities, and sociology, Vol. 7, No. 8, pp. 411-418, 2017.
[8] Hyun Shin Park, Sung woong Kim, Min-Ho Jin, Chan Dong Yoo, "Current trend of speech recognition base machine learnig", IEIE, pp. 18-27, 2014.
[9] Jong-Sub Lee, Sand-Yeob Oh, "Vocabulary Retrieve System using Improve Levenshtein Distance algorithm", The Journal of Digital Policy & Management, Vol. 11, No. 11, pp. 367-372, 2013.
Doi.org/10.14400/JDPM.2013.11.11.367
[10] Eiichi Tanaka, Tamotsu Kasai, "Synchronization and Substitution Error-correcting codes for the Levenshtein Metric", IEEE Trans. Information Theory, Vol. IT-22, No. 2, pp. 156-162, 1976.
[11] Hee-Kyung Roh, Kang-Hee Lee, "A Basic Performance Evaluation of the Speech Recognition APP of Standard Language and Dialect using Goolgle, Naver, and Daum KAKAO APIs", Asia-pacific Journal of Multimedia Services Convergent with Art, Humanities, and Sociology, Vol. 7, No. 12, pp. 819-829, 2017.

※ This work was supported by Seokyeong University in 2019.