Journal of the Korea Academia-Industrial cooperation Society
Vol. 21, No. 7 pp. 699-707, 2020

https://doi.org/10.5762/KAIS.2020.21.7.699
ISSN 1975-4701 / eISSN 2288-4688

한국어 특성 기반의 STT 엔진 정확도를 위한 정량적 평가방법 연구

민소연[1*], 이광형[2], 이동선[3], 류동엽[4]
[1]서일대학교 정보통신공학과, [2]서일대학교 소프트웨어공학과, [3]숭실대학교 컴퓨터학과, [4]넥스트지

A Study on Quantitative Evaluation Method for STT Engine Accuracy based on Korean Characteristics

So-Yeon Min[1*], Kwang-Hyong Lee[2], Dong-Seon Lee[3], Dong-Yeop Ryu[4]
[1]Dept. of Information and Communication Engineering, Seoil University
[2]Dept. of Computer Software, Seoil University
[3]Dept. of Computer Science, Soongsil University, [4]NEXTG

요 약 딥러닝 기술의 발전으로 STT(Speech To Text), TTS(Text To Speech), 챗봇(ChatBOT), 인공지능 비서 등 다양한 분야에 음성처리 관련 기술이 적용되고 있다. 특히, STT는 음성 기반 관련 서비스의 기반이며, 인간의 언어를 텍스트로 변환시키기 때문에 IT관련 서비스에 대한 다양한 응용을 할 수 있다. 따라서 최근 일반 사기업, 공공기관 등 여러 수요처에서 관련 기술에 대한 도입을 시도하고 있다. 하지만 정량적으로 수준을 평가할 수 있는 일반적인 IT 솔루션과는 달리 STT엔진에 대한 정확성을 평가하는 기준과 방법이 모호하며 한국어의 특성을 고려하지 않기 때문에 정량적인 평가 기준 적용이 어렵다. 따라서 본 연구에서는 한국어의 특성에 기반한 STT엔진 변환 성능 평가에 대한 가이드를 제공함으로써 엔진제작사는 한국어 특성에 기반한 STT변환을 수행 할 수 있으며, 수요처에서는 더 정확한 평가를 수행할 수 있다. 실험 데이터에서 기존 방식에 비해 35% 더 정확한 평가를 수행할 수 있다.

Abstract With the development of deep learning technology, voice processing-related technology is applied to various areas, such as STT (Speech To Text), TTS (Text To Speech), ChatBOT, and intelligent personal assistant. In particular, the STT is a voice-based, relevant service that changes human languages to text, so it can be applied to various IT related services. Recently, many places, such as general private enterprises and public institutions, are attempting to introduce the relevant technology. On the other hand, in contrast to the general IT solution that can be evaluated quantitatively, the standard and methods of evaluating the accuracy of the STT engine are ambiguous, and they do not consider the characteristics of the Korean language. Therefore, it is difficult to apply the quantitative evaluation standard. This study aims to provide a guide to an evaluation of the STT engine conversion performance based on the characteristics of the Korean language, so that engine manufacturers can perform the STT conversion based on the characteristics of the Korean language, while the market could perform a more accurate evaluation. In the experiment, a 35% more accurate evaluation could be performed compared to the existing methods.

Keywords : Speech To Text, Text To Speech, Evaluation, Measure, Korean Characteristics

*Corresponding Author : So-Yeon Min(Seoil Univ.)
email: symin@seoil.ac.kr
Received June 11, 2020 Revised July 2, 2020
Accepted July 3, 2020 Published July 31, 2020

1. 서론

STT(Speech To Text)는 음성인식의 한 분야로서 사람의 음성언어를 컴퓨터의 해석으로 문자데이터로 변환하는 처리를 의미한다. 키보드나 기타 입력장치에 의한 입력이 아니라 사람의 음성을 통한 입력이 되기 때문에 HCI(Human Computer Interaction), 텔레메틱스(Telemetics), 인공지능 비서, 챗봇(ChatBot) 등 다양한 기술의 기반이 된다. STT는 발화자의 음성을 기계적인 알고리즘을 통해 텍스트로 변환을 수행한다.

최근 STT를 위한 음성인식 엔진은 딥러닝(Deep Learning)알고리즘을 통해 음향과 언어모델을 이용해 정확도를 높이고 있다. 전통적인 음성인식 알고리즘인 HMM(Hidden Markov Model) 이외에 딥러닝 기반알고리즘으로 널리 사용되는 DNN(Deep Neural Network)과 RNN(Recurrent Neural Network)기법을 적용함으로써 과거에 비해 높은 정확도를 보이고 있다.

STT를 수행하는 과정은 입력음성을 전처리한 후 많은 음성 데이터에 의해 트레이닝된 모델과 비교해 텍스트 결과를 출력한다. 따라서 얼마나 다양한 음성 데이터로 트레이닝을 했는지, 얼마나 특화된 알고리즘을 통해 모델을 생성했는지 등 다양한 기술에 따라 출력되는 결과물이 다르게 나타날 수 있다. 따라서 정확한 성능 평가 및 테스트를 위해서는 입력된 오디오가 얼마나 정확하게 텍스트로 변환이 되었는가에 대한 평가를 수행하는데 이때, 필요한 기술이 텍스트 유사도(similarity) 평가이다.

텍스트 유사도 측정은 자연어처리(NLP, Natural Language Processing) 분야에서 중요한 연구 분야이며 최근 인공지능(AI, Artificial Intelligence)기술의 발전과 더불어 적용 분야가 많다. 문자열의 유사도 측정은 두 문자열 간의 의미적 유사성을 점수화하며 두 데이터가 얼마나 같은지 나타내주는 척도이다. 데이터 과학 분야에서 데이터 간의 유사도를 측정하는 것은 데이터의 분류(classification) 및 군집화(clustering)의 기반이며 향상된 알고리즘을 적용하기 위한 기반이 된다.

데이터에 대한 유사도를 비교할 때 단순 1:1 매칭이면 간단할 수도 있지만 음성을 통해 변환된 텍스트는 여러 변수가 존재한다. 영어를 텍스트로 변환하는 경우, 영어는 알파벳으로 이루어져 있고 띄어쓰기도 명확하기 때문에 유사도를 측정하기가 상대적으로 유리하다. 한국어의 경우는 초성, 중성, 종성이 하나의 글자를 이루고 있으며, 정확한 띄어쓰기도 어렵다. 또한 두음법칙, 연음법칙 등으로 인해 충분한 의미전달은 되지만 정확한 한글 표기와 발음상의 다른 점이 많은 경우가 많다. 따라서 한글에 대한 유사도 평가는 영어와는 다른 식으로 접근을 해야 좀 더 정확한 평가를 할 수 있다.

전통적인 유사도 측정 방법은 언어학적 지식, 언어에 따른 구조적 분석 및 복잡한 연산 과정이 필요하다[1].

STT엔진의 정확도 평가를 위한 과정은 일반적으로 다음과 같다.

첫 번째 테스트 오디오 음성파일과 정확하게 매칭되는 원본 텍스트 파일을 준비한다.

두 번째 STT 엔진을 통해 나온 텍스트 결과파일과 원본 오디오의 텍스트 파일을 비교해 유사도를 비교한다.

평가과정은 매우 단순하지만 인식율을 평가하는데 몇몇 오차가 존재할 수 있다.

우선 원본 텍스트 파일을 생성할 때 오류가 존재할 수 있다. 예를 들면 원본 오디오가 '산 토끼 토끼야 어디를 가느냐'가 녹음된 경우 이를 원본 텍스트로 생성할 때 두음법칙과 연음법칙 등에 따라 '산 톡끼 톡끼야 어디를 가는야'라고 발음된다. 이 경우 발음 그대로 원본 텍스트를 생성하는게 좋은지 아니면 발음 그래도 원본 텍스트를 생성하는게 좋은지에 대한 판단이 필요하다. 어떤 원본 텍스트를 생성했는가에 따라서 유사도가 달라질 수 있기 때문이다.

Table 1은 STT엔진 변환 결과에 대한 예를 보여주고 있다. 일반적으로 평가 기준으로 하는 CER(Character Error Rate) 또는 LER(Letter Error Rate)나 WER(Word Error Rate)을 적용하면, A의 경우 정확하게 변환이 되어 인식율이 100%이지만 B와 C의 경우는 12글자 중 5글자만 일치하기 때문에 인식율은 동일하게 약 41%이다. 하지만 B의 경우 이해 하는데는 큰 무리가 없지만 C의 경우는 이해하기 몹시 어렵다. 현재 STT엔진은 100% 정확도를 보이기 어렵고 외래어나 트레이닝되지 않은 음을 음향모델에 따라 텍스트로 변환하는 경우도 매우 빈번하기 때문에 이와 같은 이슈에 대한 고려가 되어야 한다.

Table 1. STT engine results comparison

Engine	Result														
A	산		토	끼		토	끼	야		어	디	를		가	느	냐
B	샨		톡	끼		톡	끼	야		어	데	를		갸	는	야
C	냠		벙	끼		벙	끼	야		걸	디	를		황	야	가

본 논문에서는 일반적인 평가방법으로 사용되는 CER 또는 LER이나 WER(Word Error Rate) 방법을 개선해 한국어 유사도 비교에 적용함으로서 더 정확한 STT 엔

진의 성능을 평가할 수 있는 방법에 대해 제안한다.

2. 관련연구

2.1 문자열 유사도 측정 방법

문자열은 데이터처리의 기준이 되며 정보검색, 문서 비교와 분류, 요약, 번역 등 다양한 분야에 적용된다. 특히 문자열 간의 유사도 측정 및 평가는 여러 전 처리의 과정이 수반되며 다양한 방법이 적용되고 있다.

문자열 유사도는 크게 문자기반 유사도와 단어기반 유사도로 나누어진다. 문자기반 유사도는 두 문자열 중 다른 문자가 얼마나 있는가를 평가하는 방식이며 단어기반 유사도는 단어가 다른 단어가 얼마나 있는가를 평가하는 방식이다. 일반적으로 문자기반 유사도 평가는 CER이며 단어기반은 WER로 평가한다.

비교할 데이터를 n차원 상의 벡터로 표현하면 데이터 간의 유사도는 두 벡터 사이의 거리(distance)에 따라 유사도를 판단하는 방식이 주로 이용된다[2].

코사인 유사도(Cosine Similarity)는 생성된 두 벡터의 각도를 비교해 유사도를 측정한다. 유사도가 높다면 두 벡터는 예각에 가까워지고 유사도가 낮다면 둔각으로 된다. 즉, 유사도가 높을수록 1에 가까워지고, 유사도가 낮을수록 -1에 가까워진다. 0은 서로 독립적인 경우를 의미한다. 이 방식은 벡터의 크기는 고려하지 않고 코사인 값에 따라 평가되기 때문에 유사도만 비교할 때 사용된다.

$$\sim ilarity = \cos(\Theta) \tag{1}$$

$$= \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\ \sqrt{\sum_{i=1}^{n} B_i^2}}$$

자카드 유사도(Jaccard Similarity)는 원본과 비교 대상 문자열에서 각각의 단어 집합을 생성한다. 두 집합의 교집합을 합집합의 개수로 나눠서 계산한다.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}, \tag{2}$$

A, B가 $empty$이면 $J(A,B) = 1$,

$0 \le J(A,B) \le 1$

코사인유사도나 자카드 유사도 같은 단어를 비교하는 유사도 비교는 문자열을 단어 단위로 추출한 후 연산 작업을 수행한다. 하지만 이와 같은 방법은 같은 의미를 갖는 유사 단어들이나 오탈자 등은 다른 단어로 처리되는 문제가 있다.

유클리디언 거리(Euclidean Distance)는 두 벡터 간의 직선 거리를 측정하는 방법으로 n차원의 두 개의 유클리디언 거리는 다음과 같이 계산된다.

$$d(p,q) = d(q,p) \tag{3}$$

$$= \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \ldots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

맨하탄 거리(Manhattan Distance)는 영역을 정사각형으로 나누고 그 구간에서 두 점 사이의 거리를 측정한다. 두 벡터의 차의 절대 값의 합을 의미한다. n 차원에 주어진 점 p, q에 대해 유클리디언 거리는 다음과 같이 계산된다.

$$d_1(p,q) = \|p - q\|_1 = \sum_{i=1}^{n} |p_i - q_i| \tag{4}$$

민코프스키 거리(Minkowski Distance)는 유클리드 거리와 맨해튼 거리를 일반화 한 것으로 n 차원 점 X, Y에 대한 거리는 다음과 같이 계산된다.

$$D(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}} \tag{5}$$

p = 1일 경우 맨해튼 거리와 같은 결과이며, p = 2일 경우 유클리드 거리와 같다. p = ∞일 경우는 체비쇼프 거리(Chebyshev Distance)와 같다.

레벤스타인 거리(Levenshtein Distance)는 두 개의 텍스트 문자열의 유사도를 비교할 때 문자열을 변환하는데 필요한 비용을 산출함으로서 얼마나 많은 비용이 들어가느냐에 따라 유사도를 판단한다. 이때 사용되는 비용 척도는 삽입(Insertion), 삭제(Deletion), 대체(Substitution)로 거리를 계산한다.

원본에는 없고 비교 대상 문자열에만 존재하는 경우는 삽입이라 한다. 원본 문자열에는 있고 비교 문자열에는 없는 경우는 삭제이다. 원본과 비교 문자가 다른 경우는

대체로 정의 된다. 이 같은 척도를 이용해 원본과 비교대상의 두 배열이 같아지는 비용 거리를 계산한다.

$$D_{(i,j)} = \min\begin{cases} D(i-1,j-1), & s_i = t_i \\ D(i-1,j-1)+\alpha, & s_i \neq t_i \\ D(i-1,j)+\beta, & insertion \\ D(i,j-1)+\beta, & deletion \end{cases} \quad (6)$$

si와 ti는 비교 문자열의 위치를 나타내며, α, β 는 대체 오류와 삽입, 삭제 오류 시 추가되는 비용을 의미한다. 레벤스타인 거리는 편집 유사도라고도 하며 일반적으로 문자기반 유사도를 비교하는데 널리 사용되지만 “우리나라 대한민국”과 “대한민국 우리나라”와 같은 예처럼 단어의 순서가 바뀐다면 의미적으로 비슷한 문자열임에도 불구하고 찾아내지 못하는 단점이 있다. 따라서 이 단점을 해결하는 혼합 유사도[3]가 제안되었지만 이 방식은 유사도를 계산하는데 소요되는 오버헤드가 크다는 단점을 갖는다.

이외에 해밍 거리(Hamming Distance)도 두 개의 단어에서 몇 개의 문자를 대체해야 같아지는가를 계산하는데 두 단어가 비교 전부터 길이가 동일해야 한다는 전재가 있기 때문에 일반적인 상황에서는 적용하기 어렵다.

2.2 문서 간 유사도 측정 방법

문서는 일반적인 문자열 보다 더 많은 단어와 문자의 집합이기 때문에 문서에 대해 특징을 추출하여 문서간의 유사도를 구하는 방법을 이용하며 TF-IDF(Term Frequency - Inverse Document Frequency) 방법을 많이 이용한다. TF-IDF는 문서 간의 유사도, 문서 내의 단어 중요도, 검색엔진에서 검색 결과의 순위 등 널리 이용되며 이때, TF는 문서 내에서 사용된 특정 단어의 빈도수를 의미한다[4].

$tf(t,d) = f_{(t,d)}$ 에서 $f_{(t,d)}$는 문서 d에 나타난 단어 t의 수이다.

하지만 이 경우 매우 긴 문서에서 3번 나타난 단어와 매우 짧은 문서에서 3번 나타난 단어 수치를 같다고 보기 때문에 로그 스케일 빈도수 (Logarithmically Scaled Frequency), 불리언 빈도수 (Boolean Frequency), 증가 빈도수 (Augmented Frequency)등을 이용해 보완한다[5].

불리언 빈도는 단어 t가 문서에 있으면 1아니면 0으로 표시한다. 로그 스케일 빈도는 많이 등장하는 단어에 대한 가중치를 줄이기 위해 다음과 같이 사용된다.

$$tf(t,d) = \log(f_{(t,d)} + 1) \quad (7)$$

증가 빈도는 단어 빈도값을 조절해 긴 문서에서 편향되는 것을 막기 위해 사용한다.

$$tf(d,f) = 0.5 + \frac{0.5 \times f_{(t,d)}}{\max f_{(t',d)}, t' \in d} \quad (8)$$

DF는 해당 단어가 나타난 문서의 수를 의미한다.

$$df(t,d) = \frac{|d \in D : t \in d|}{|D|} \quad (9)$$

$|D|$: 문서 집합 크기이거나 전체 문서의 수

$|d \in D : t \in d|$: 단어 t가 있는 문서의 수

IDF는 DF의 역변환이며, 공통 단어가 나타나는 빈도수는 문서 간 유사도 측정에 중요 지표가 된다[6]. 비교하는 문서의 수가 많으면 많아질수록 IDF의 값이 급격하게 커지므로 IDF는 log를 취하게 된다. 그리고 특정 단어가 문서에 존재하지 않는 경우는 분모가 0이 되기 때문에 1을 더함으로서 분모를 0이상이 되게 한다.

$$idf(t,D) = \log\left(\frac{|D|}{|d \in D : t \in d| + 1}\right) \quad (10)$$

특정한 문서 중 출현하는 단어 빈도수가 높고 전체 문서에서 해당 단어를 포함한 문서의 수가가 작을수록 TF-IDF 값은 높아진다. 따라서 계산된 TF-IDF값을 이용해 문서 중 의미 없이 흔하게 나타나는 단어를 걸러낼 수 있다. IDF의 로그 값은 항상 1 이상이므로 IDF값과 TF-IDF값은 항상 0 이상이다. 즉, 특정 단어를 포함하는 문서들이 많을수록 로그 함수 안의 값이 1에 가까워지게 되고, 이 경우 IDF값과 TF-IDF 값은 0에 가까워지게 된다.

$$tfidf(t,d,D) = tf(t,d) \times idf(t,D) \quad (11)$$

TF-IDF 값은 특정 문서 내에서 출현한 단어의 빈도수가 많거나 전체 문서 중에서 특정 단어를 포함한 문서가 적을수록 TF-IDF 값이 높아지게 된다. 따라서 이 값을 이용해 문서에 나타나는 흔한 단어들을 걸러낸다. 너무 흔한 단어들은 중요도가 낮다고 평가하며 특정 단어가 갖는 중요도를 통해 문서의 유사도를 비교하는데 사용된다.

이외에 대용량의 문서에서 독립적으로 산출된 정보를 활용해 유사도 측정하는 Corpus-used similarity 방식[7]이나 지식기반 네트워크(semantic network)를 이용한 유사도 측정방법 등이 연구되었다.

2.3 의미기반 유사도 측정을 위한 방법

워드 임베딩은 단어를 벡터로 변환해주는 알고리즘으로 Neural Network Language Model(NNLM)을 통해 학습 속도와 성능을 비약적으로 끌어올려 자연어 처리 분야 등에 널리 사용되고 있는 알고리즘이다[8].

$$p(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w=1}^{W} \exp(u_w^T v_c)} \quad (12)$$

o 는 문맥단어 c는 중심단어를 의미한다. 즉, p(o|c)는 중심단어(c)가 주어졌을 때 문맥단어(o)가 등장할 조건부확률을 의미하기 때문에 이 식을 최대화하는 것은 중심단어로 문맥단어를 맞춘다는 의미가 된다.

Word2Vec에서는 p(o|c)을 위 식 우변과 같이 정의했다. u와 v는 단어벡터를 의미하며 엄밀히 얘기하면 u와 v는 다른 벡터들이다.

벡터로 표현된 단어 간의 거리를 비교해 유사도 및 문장 구성 등에는 유리하지만, STT정확도를 비교하는 용도로 사용하기에는 매우 많은 말뭉치를 대상으로 학습이 선행 되어야하기 때문에 무리가 있다. 또한 Word2Vec은 기존 카운트기반 알고리즘과 같이 자주 같이 등장하는 단어들의 정보를 보존하기 때문에 본질적으로 기존 카운트 기반의 접근방식과 크게 다르지 않다[9].

딥러닝의 RNN을 사용하여 문장의 유사도를 측정하는 방법도 연구되었다[10]. Seq2Seq는 2개의 RNN을 연결하여 텍스트를 생성하는 모델로 입력과 출력이 같은 문장이 되도록 학습을 시키면 인코더의 출력인 벡터는 문장을 압축한 정보를 나타낸다. 이 값의 코사인 유사도를 측정하여 두 문장의 유사도를 판단한다. 하지만 문장의 구조가 다른 경우에는 성능이 좋지 않다는 문제를 갖는다.

워드넷(WordNet)은 명사, 동사, 부사, 형용사 등을 구분하여 동의어에 대해서 계층적 네트워크로 구성한다. 따라서 단어와 단어 간의 관계를 파악하기 쉽기 때문에 자연어처리 분야에서 널리 활용된다. 단어들 간 다양한 의미 관계를 이용해 단어와 단어의 유사도 측정을 위해 활용 될 수 있지만 텍스트 정확도를 평가하는 데는 적합하지 않다. 즉, 문서 간의 유사도는 두 문서 간의 유사성 판단 등에 매우 유용하지만 단순 STT 결과에 대한 정확성을 평가하기에는 부적합하다.

이외 같은 어절이 사용되었는지 계산하는 방법과 문장 특성 추출 결과를 이용해 구조적 유사성을 분석하는 방법도 연구되었다[11].

기존의 기법들은 본 연구에서 다루고자 하는 문제를 해결하기 위해 부족한 부분들이 있다. 언어에 의해 생성된 결과물이 사람이 얼마나 잘 이해하는가에 대한 평가 방법이 부족하다. 본 논문에서는 주어진 문제에 더욱 적합한 기법을 제안하고자 한다. 본 연구에서는 단순 데이터의 일치성 여부보다는 의미적으로 더 적합한 유사도를 측정하는 방법을 제안한다.

3. 제안 방법

일반적으로 사용하는 성능확인 평가방법은 다음과 같다.

(원본 글자 수 - 잘못 변환된 글자 수) / 원본 글자 수 * 100

$$errrate = \frac{n_o - n_e}{n_o} \times 100 \quad (13)$$

이때, n_o는 음성 변환 대상 글자 중 특수기호(띄어쓰기, 느낌표, 물음표 등 문장 부호)를 제외한 글자의 수이며, n_e는 잘못 변환된 글자 수를 의미하며 원본의 내용이 빠져 있거나 잘못 변환된 글자, 원본에는 없는 글자가 추가된 글자를 의미한다. 따라서 레벤스타인 편집거리를 이용해 원본과 STT의 결과가 얼마나 다른가를 평가한다. 이와 같은 평가는 1:1매핑이기 때문에 자연스러운 평가 방법일 수는 있지만 한국어의 특성이 고려되지 않았기 때문에 몇몇 문제가 존재한다.

첫 번째, 정확한 원본 텍스트에 대한 신뢰가 필요하다. 두음법칙, 연음법칙, 외래어 변환, 숫자 변환 등에 대해서 어떻게 원본 텍스트를 생성하느냐에 따라 STT엔진의 평가가 달라질 수 있다.

두 번째, 한글은 영어처럼 알파벳으로 이루어지지 않고 초,중,종성이 하나의 글자를 이루며, 뜻은 다르지만 동일한 발음을 내는 문자가 많다. 예를 들면 '게'와 '개'같은 경우는 STT변환이 난해하다.

세 번째, 한글은 정확한 맞춤법이 맞지 않더라도 읽는 사람들은 대부분 이해할 수 있다. 최근 인터넷상의 댓글놀이나 언어유희가 많은데 예를들면, '이 하우스의 청소상태는 매우 불량합니다.' 라는 표현을 '잉 항웅승잉 청송

상탱는 맹웅 불량합닝당.'라고 적었거나 '한쿡인뜰만아라 뽈쑤있께쩍을껭용'라고 적었더라도 기계적인 번역 등은 안되지만 글을 읽는 사람은 충분히 의미를 이해할 수 있다.

단어 우월효과(WSE, Word Superiority Effect)는 한 단어의 첫 번째와 마지막 음절이 정확한 위치에 있을 때 해당 단어를 구성하는 나머지 음절은 틀리게 배열이 되어있어도 사람은 단어를 정확히 인지할 수 있는 사실을 설명한다[12].

'한 단어 안에서 글자의 배순열서 보다는 첫 번 째와 마지막 글자가 정확한 위치에 있는 것이 중요하다. 이장문은 글자 순서가 틀렸지만 큰 문제없이 문장을 이해할 수 있다.

네 번째, 숫자, 외래어, 영문단어에 대한 변환기준이 필요하다. 예를 들면, '2020년입니다'의 원본텍스트가 '이천이십년입니다', '이공이공년입니다'라고 변환된 경우 기존 평가척도를 적용하면 인식율은 50%로 측정된다.

완벽한 발음을 가진 화자를 통해 음성입력이 되고 성능 평가를 위한 원본 텍스트가 정확한 문법에 맞게 작성이 되고, STT엔진이 정확한 문법에 맞는 음성과 레이블 데이터로 트레이닝이 되었다면 이상적인 형태의 STT변환이나 테스트 성능 평가가 이뤄지겠지만 한글은 이런 경우의 모두 고려하기가 매우 어렵다. 가장 정확한 결과가 나오는 경우가 아니라면 STT결과를 읽는 사람의 이해도가 높을수록 STT의 성능이 좋다고 판단할 수 있다. 즉, 동일한 70%의 성능을 보인다고 해도 내용을 이해하는데는 지장이 없을 수도 있고, 이해하기 몹시 어려운 경우가 있을 수도 있기 때문에 사람이 이해하기 쉬운 형태로 변환된 엔진 정확도가 더 좋다고 판단을 해야한다.

원본 텍스트의 내용과 완벽하게 동일한 변환을 수행하면 좋지만 반드시 완전한 변환이 되지 않더라도 사람이 이해할 수 있는 텍스트 변환에 대한 평가가 고려되어야 한다.

따라서 본 연구에서는 이와 같은 특성을 적용해 좀 더 정확한 STT성능 평가를 수행 할 수 있는 방법을 제안한다.

띄어쓰기, 느낌표, 물음표, 마침표, 쉼표 등 모든 문장부호와 키보드로 입력되는 특수기호 등은 모두 제외한다. 단, %, $ 등 일상적으로 사용되는 기호는 포함한다.

숫자나 날짜 등의 경우 한글로 변환된 경우가 많기 때문에 오인식 처리를 하면 안 된다. 예를 들면 "2020년"이라 표기된 원본 텍스트의 경우 "이천이십년"이라고 변환된 엔진이 틀렸다고 하지 않는다.

초,중,종성에 따른 고려가 필요하다. 즉, 더 중요한 부분에 가중치를 부여함으로써 좀 더 정확한 평가를 수행할 수 있다.

a. 초성의 가중치는 높다.
b. 중성의 가중치는 보통이다.
c. 종성의 가중치는 낮다.

한글은 유니코드로 44032번부터이며 초성은 19개, 중성은 21개, 종성은 28개로 구성된다.

Table 2. Weight composition

classification	contents	weight
onset	ㄱ, ㄲ, ㄴ, ㄷ, ㄸ, ㄹ, ㅁ, ㅂ, ㅃ, ㅅ, ㅆ, ㅇ, ㅈ, ㅉ, ㅊ, ㅋ, ㅌ, ㅍ, ㅎ	H
nucleus	ㅏ, ㅐ, ㅑ, ㅒ, ㅓ, ㅔ, ㅕ, ㅖ, ㅗ, ㅘ, ㅙ, ㅚ, ㅛ, ㅜ, ㅝ, ㅞ, ㅟ, ㅠ, ㅡ, ㅢ, ㅣ	M
coda	ㄱ, ㄲ, ㄳ, ㄴ, ㄵ, ㄶ, ㄷ, ㄹ, ㄺ, ㄻ, ㄼ, ㄽ, ㄾ, ㄿ, ㅀ, ㅁ, ㅂ, ㅄ, ㅅ, ㅆ, ㅇ, ㅈ, ㅊ, ㅋ, ㅌ, ㅍ, ㅎ, ' '	L

단어의 경우 단어우월 효과를 고려해 첫 글자와 마지막 글자에 대한 가중치를 상대적으로 높게 부여한다.

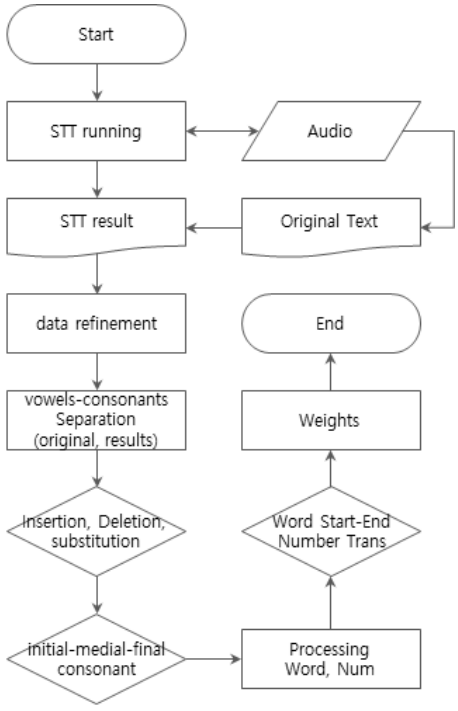

Fig. 1. Experimental Process

$$\left(\sum_{i=0}^{n_{jm}}\right)_{lev}+\sum_{j=0}^{w_f}+\sum_{k=0}^{w_m}+\sum_{l=0}^{w_l}+\sum_{m=0}^{w_w} \quad (14)$$

1. 모든 문장에 자음을 넣는다. 이때 쌍자음을 넣어도 된다.
2. 자음과 모음을 분리해서 작성한다.
3. 단어 우월 효과를 이용해 글자 앞뒤 배열을 바꾼다.

4. 실험결과

테스트에 사용된 환경은 Table 3과 같다.

Table 3. Test environment

classification	contents
CPU	Intel I9
RAM	64G(16*4)
VGA	RTX 2080 Ti D6 11GB
OS	Ubuntu 18.04LTS
Language	Python 3.7

실험에 사용된 오디오 데이터는 다양한 시간, 코덱 등을 사용했다. STT에 사용된 엔진의 출력결과가 실험 환경에 적합한 데이터를 생성하기 위해 텍스트 레이블링 시 변환작업을 수행한 후 새로운 각각의 모델을 생성했다.

Table 4. Test set

classification	contents
Duration	3 ~ 60sec
Format	Wav, Mp3, AAC
Bitrate	64~192kbps
Channel	Mono, Stereo

아래의 결과는 원본과 A, B모델에 대한 테스트를 수행한 예이다. 기존 방식으로 정확도를 산정하면 A모델과 B모델 모두 56.25%로 동일하다.

Table 5. Same accuracy by letter

Engine	Result															
Org	산		토	끼		토	끼	야		어	디	를		가	느	냐
A	샨		톡	끼		톡	끼	야		어	데	를		갸	는	야
B	간		소	끼		소	끼	야		거	디	를		나	는	햐

자음과 모음을 분리한 경우도 83.33%로 동일한 인식율을 나타낸다. 수치상으로는 동일한 결과를 보이고 있지만 사람이 인식하기에는 A가 더 좋은 결과를 보여준다.

Table 6. Same accuracy for each character

Engine	Result
Org	ㅅㅏㄴㅌㅗㄲㅣ ㅌㅗㄲㅣ ㅇㅑ ㅇㅓ ㄷㅣ ㄹㅡㄹㄱㅏ ㄴㅡㄴㅑ
A	ㅅㅑㄴㅌㅗㄱㄲㅣ ㅌㅗㄱㄲㅣ ㅇㅑ ㅇㅓ ㄷㅔ ㄹㅡㄹㄱㅏ ㄴㅡㄴㅇㅑ
B	ㄱㅏㄴㅅㅗㄲㅣ ㅅㅗㄲㅣ ㅇㅑ ㄱㅓ ㄷㅣ ㄹㅡㄹㅏ ㄴㅡㄴㅎㅑ

다음 표의 결과를 기존의 방식으로 평가하면 A의 정확도는 33.33%이고 B는 72.22%의 결과를 나타낸다. 즉, B의 엔진이 더 정확하다고 평가된다.

Table 7. Different accuracy by letter

Engine	Result															
Org	자	가	격	리	자	는	1	층	에	서	체	크	인	함	니	다
A	작	아	경	니	자	는	일	층	애	서	채	크	잉	함	닝	당
B	삭	가	격	리	여	는	1	층	에	너	샤	크	인	갑	니	다

자모음별로 정확도를 계산하면 A는 75.26%이고 B는 82.22%로 B가 더 우수한 성능을 나타낸다. 하지만 10명의 사람에게 A와 B의 결과를 보여주고 어떤 문맥이 더 이해되는가를 조사했을 때 9명이 A의 결과가 더 이해하기 쉽다는 답을 했다. 즉, 수치상의 결과와 사람이 인식되는 결과가 다르다는 것이다.

Table 8. Different accuracy by letter

Engine	Result
Org	ㅈㅏ ㄱㅏ ㄱㅕㄱㄹㅣ ㅈㅏ ㄴㅡㄴㅊㅡㅇㅇㅖ ㅅㅓ ㅊㅖ ㅋㅡ ㅇㅣㄴㅎㅏㅂㄴㅣ ㄷㅏ
A	ㅈㅏㄱㅇㅏ ㄱㅕㅇㄴㅣ ㅈㅏ ㄴㅡㄴㅇㅣㄹㅊㅜㅇㅇㅐ ㅅㅓ ㅊㅐ ㅋㅡ ㅇㅣㅇㅎㅏㅁㄴㅣㅇㄷㅏㅇ
B	ㅅㅏㄱㄱㅏ ㄱㅕㄱㄹㅣ ㅇㅕ ㄴㅡㄴㅊㅡㅇㅇㅖ ㄴㅓ ㅅㅑ ㅋㅡ ㅇㅣㄴㄱㅏㅂㄴㅣ ㄷㅏ

실험 결과는 Table 9와 같다.

Table 9. Experiment result

classification	Model A	Model B
Conventional method accuracy (Apply CER for each character)	42.35%	78.16%
Accuracy after vowels/consonants separation (Apply CER for each character)	68.53%	72.18%
Accuracy by feeling	92.16%	7.84%
Proposed method	94.12%	5.88%

본 연구는 STT 인식율 향상이 아닌 인식 결과에 대한 정확도 평가이기 때문에 인식율 자체의 중요도 보다는 동일한 인식율을 갖는 경우 어떤 엔진이 한국어의 특성에 적합하게 사람이 이해하기 용이한가에 대한 방법이다.

자음과 모음을 분리한 상태에서만 평가해도 모델간의 차별성을 나타낼 수 있지만 사람이 느끼기에 더 낮은 모델이 더 높은 수치를 갖는 경우도 많다. 하지만 제안 방법은 사람이 느끼는 정확도와 유사한 결과를 나타냄을 확인 할 수 있다. 따라서 제안방법을 통해 STT 결과에 대한 정확도를 평가하는 경우 일반적인 방법보다 더 정확한 결과를 보여줌을 확인 할 수 있다.

5. 결론

STT에 의한 결과에 대한 정확도 평가를 수행하고 여러 업체의 엔진 중 더 적합한 엔진을 선택해 사업과 다양한 분야에 적용하는 경우가 빈번하다. 하지만 한국어의 특성에 기반하지 않은 일반적인 CER을 적용함으로써 더 낮은 성능을 갖는 STT 엔진이 동일한 정확도이거나 더 좋은 정확률 가지고 있다고 평가되는 경우가 발생한다.

본 연구에서는 한국어의 특성에 기반한 정확도 평가결과를 제공함으로써 여러 엔진 중 더 정확도가 높은 엔진을 평가하는 방법을 제안했다. 사람이 느끼는 정확도와 대부분 일치하는 결과를 얻을 수 있었다.

하나의 엔진에 대한 모델을 트레이닝하고 높은 인식율을 나타내는데는 많은 비용과 시간이 소모된다. 또한 엔진의 트레이닝 데이터와 특성에 따라 더 정확한 적용이 가능한 엔진이 존재한다.

본 연구에서는 비교적 짧은 문장에 대한 테스트와 제한된 인력을 통한 평가를 수행했다. 또한 사투리나 외래어에 대한 충분한 고려가 되어있지 않다. 따라서 향후 10분 이상의 장문과 사투리나 외래어 등에 대한 연구를 보완한다면 더 범용적이고 정확한 STT 정확도 평가 방법을 제공할 수 있을 것이다.

References

[1] P. Achananuparp, et al., "The evaluation of sentence similarity measures." Data Warehousing and Knowledge Discovery, Springer Berlin Heidelberg, pp. 305-316, 2008.
DOI : http://dx.doi.org/10.1007/978-3-540-85836-2_29

[2] T. Mikolov, et al., "Distributed representations of words and phrases and their compositionality," In Proc. of Advances in Neural Information Processing Systems, pp. 3111-3119, 2013.

[3] J. Wang, G. Li and J. Fe, "Fast-Join: An Efficient Method for Fuzzy Token Matching based String Similarity Join", In ICDE, 2011.

[4] Lee Mi-suk, "A copy detection system," Ph.D. dissertation, University of Dongguk, Seoul, Korea, 2005.

[5] Manning, C. D.; Raghavan, P.; Schutze, H. 《Introduction to Information Retrieval》. Cambridge University Press. 100~123. ISBN 9780521865715. Scoring, term weighting, and the vector space model

[6] Lee Mi-suk, "A copy detection system," Ph.D. dissertation, University of Dongguk, Seoul, Korea, 2005.

[7] Koopman B, Zuccon G, Bruza P, Sitbon L, Lawley M: An evaluation of corpus-driven measures of medical concept similarity for information retrieval. Proceedings of the 21st ACM International Conference on Information and Knowledge Management. New York: ACM, 2439-2442, 2012.
DOI : http://dx.doi.org/10.1145/2396761.2398661

[8] T. Mikolov et al., "Distributed Representations of Words and Phrases and their Compositionality", Int. Conf. NIPS, pp. 3111-3119, 2013.

[9] O. Levy and Y. Goldberg. Neural word embedding as implicit matrix factorization. In Advances in Neural Information Processing Systems, pages 2177—2185, 2014.

[10] DongKeonLee, O KyoJoongOh, Ho-Jin Choi, Measuring the Syntactic Similarity between Korean Sentences Using RNN, KCC, 2016.

[11] P. Achananuparp, et al., "The evaluation of sentence similarity measures." Data Warehousing and Knowledge Discovery, Springer Berlin Heidelberg, pp. 305-316, 2008.
DOI : http://dx.doi.org/10.1007/978-3-540-85836-2_29

[12] Wo Hyun Jung, Soo Jin Park, Word and coding-unit superiority effect in the perception of Korean Letter, The Korean Psychological Association. 18-2, pp.139-156, 2006.

민 소 연(So-Yeon Min) [종신회원]

- 1994년 2월 : 숭실대학교 전자공학과 (공학사)
- 1996년 2월 : 숭실대학교 전자공학과 (공학석사)
- 2003년 2월 : 숭실대학교 전자공학과 (공학박사)
- 2005년 3월 ~ 현재 : 서일대학교 정보통신공학과 부교수

〈관심분야〉
통신 및 신호처리, 정보통신

이 광 형(Kwang-Hyong Lee) [종신회원]

- 1998년 2월 : 광주대학교 컴퓨터공학과 (공학사)
- 2002년 2월 : 숭실대학교 컴퓨터학과 (공학석사)
- 2005년 2월 : 숭실대학교 컴퓨터공학과 (공학박사)
- 2005년 3월 ~ 현재 : 서일대학교 소프트웨어공학과 부교수

〈관심분야〉
멀티미디어 보안, 사물인터넷, 학습콘텐츠, 영상처리

이 동 선(Dong-Seon Lee) [정회원]

- 2007년 2월 : 한국산업기술대학교 (공학학사)
- 2009년 7월 : 연세대학교 공학대학원 (공학석사)
- 2018년 8월 ~ 현재 : 숭실대학교 컴퓨터학과 (박사과정)
- 2000년 5월 ~ 현재 : KTH

〈관심분야〉
빅데이터, 가상현실, IT서비스

류 동 엽(Dong-Yeop Ryu) [정회원]

- 2003년 2월 : 숭실대학교 컴퓨터공학과 (공학석사)
- 2007년 2월 : 숭실대학교 컴퓨터학과 (공학박사)
- 2014년 3월 : 엠텔로
- 2014년 ~ 현재 : 넥스트지, M1소프트, 메이슨인텔리전스, 숭의여대 겸임교수

〈관심분야〉
머신러닝, STT, 챗봇, 자연어처리