http://dx.doi.org/10.5392/JKCA.2011.11.12.012

다양한 스마트폰 키패드 환경에서 유사 단어 검색을 위한 수정된 편집 거리 계산 방법

Modified Edit Distance Method for Finding Similar Words in Various Smartphone Keypad Environment

송영길, 김학수
강원대학교 IT대학 컴퓨터정보통신공학전공

Yeong-Kil Song(nlpyksong@kangwon.ac.kr), Hark-Soo Kim(nlpdrkim@kangwon.ac.kr)

요약

대부분의 스마트폰은 터치패드 기반의 가상 키패드를 사용한다. 가상 키패드는 기기의 화면 크기나 입력 방법의 물리적인 한계로 입력 오류가 자주 발생한다. 이 문제를 해결하기 위해 유사 단어를 찾기 위한 많은 연구가 있었다. 본 논문에서는 편집 거리 방법을 다양한 가상 키패드를 고려하여 수정하는 방법을 제안한다. 제안 방법은 다양한 키패드에서 발생하는 입력 오류를 효과적으로 해결하기 위해, 입력 문자열을 사용자가 실제 누르게 되는 입력열로 변환하고, 가상 키패드의 특성에 따라 편집 비용을 수정하였다. 다양한 키패드에서 실험한 결과 제안 방법이 일반적인 편집 거리 방법을 이용한 것 보다 좋은 성능을 보였다.

■ 중심어 : | 스마트폰 | 유사단어 | 편집 거리 |

Abstract

Most smartphone use virtual keypads based on touch-pad. The virtual keypads often make typographical errors because of the physical limitations of device such as small screen and limited input methods. To resolve this problem, many similar word-finding methods have been studied. In the paper, we propose an edit distance method (a well-known string similarity measure) that is modified to consider various types of virtual keypads. The proposed method effectively covers typographical errors in various keypads by converting an input string into a physical key sequence and by reflecting characteristics of virtual keypads to edit scores. In the experiments with various keypads, the proposed method showed better performances than a typical edit distance method.

■ keyword : | Smartphone | Similar Words | Edit Distance |

I. 서 론

전 세계적으로 스마트폰(smart-phone) 열풍이 불면서, 2011년 3월 24일 기준 국내 스마트폰 사용자는 1,000만 명을 돌파했고, 연내 2,000만을 돌파할 것이라는 분석이 있다[1]. 이런 스마트폰 열풍과 함께 다양한 스마트폰 어플리케이션(application)들이 개발되고 있다. 이러한 스마트폰 어플리케이션에서 키패드(keypad)를 이용한 문자입력은 가장 많이 사용되는 입력방법이다. 대부분의 스마트폰은 기기의 두께를 줄이기 위한

* 이 연구는 삼성전자 산학협력 과제의 지원을 받아 수행되었음.
* 이 논문은 2010년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (No.2010-0009875).
접수번호 : #110830-005
접수일자 : 2011년 08월 30일
심사완료일 : 2011년 09월 29일
교신저자 : 김학수, e-mail : nlpdrkim@kangwon.ac.kr

방편으로 물리적인 키패드를 제거하고, 터치스크린(touch screen) 방식의 가상 키패드(virtual keypad)를 탑재하고 있다. 가상 키패드는 키의 크기가 작아서 주변의 다른 키를 누를 가능성이 크고, 민감하게 반응하여 오타를 발생시킬 가능성이 크다는 단점이 있다. 이와 같은 단점은 스마트폰 사용자들이 가장 크게 느끼는 불만요소이다[2]. 그래서 오류를 포함한 입력과 유사한 단어를 찾는 연구가 필요하다. 입력 문자열과 유사한 단어를 찾는 연구에서 가장 중요한 것은 단어 사이의 유사도를 계산하는 방법이다. 기존의 연구들은 주로 개인용 컴퓨터 환경에서 이루어졌기 때문에 스마트폰의 입력 환경을 고려하지 못하였다. 본 논문에서는 스마트폰 입력환경을 고려하여 다양한 가상 키패드에 적용 가능한 유사도 계산 방법을 제안한다.

II. 관련 연구

철자 유사도를 계산하는 기존연구는 대표적으로 유사 단어 그룹핑(grouping)[3-5], n-그램 방법, 편집 거리(edit distance)에 기반한 방법[6][7]이 있다.

유사 단어 그룹핑 방법은 외래어 음차표기, 발음 등으로 인해 다르게 쓰인 유사한 단어들을 그룹핑하는 방법이다. 강병주(1999)[3]는 외국어의 음차표기의 개인차에 따른 오류를 처리하기 위해서 대표적인 영어의 음성적 유사도 비교 알고리즘인 Soundex 알고리즘을 한국어의 특성에 맞게 수정한 Kodex 알고리즘을 제안하였다. 예를 들어 “sound”가 “싸운드/사운드”로 표기될 경우 두 가지 음차 표기에 동일한 코드를 부여한다. 이 방법은 최소한의 제약사항으로 최대한의 재현율을 보였지만 정확률이 낮은 단점이 있다. 고숙현(2007)[4]은 Kodex 알고리즘의 정확률이 낮은 단점을 보완하기 위해 한국어의 발음 특성에 기반하여 세부적인 규칙을 정하고, 기존 알고리즘의 조건을 수정하는 방법을 제안하였다. 김효경(2006)[5]은 한국어 표기의 개인차에 따른 오류를 처리하기 위해서 한국어 표준어 규정의 규칙들을 이용한 방법을 제안하였다. 이 방법은 기존의 입력 문자열을 한국어 표준어 규정의 규칙들을 이용하여 변환한 뒤 코드를 부여한다. 이와 같은 철자 그룹핑 방법들은 오류 사전 없이 유사한 단어를 찾을 수 있다는 장점이 있다. 그러나 이 방법들은 표기/발음법 차이로 인한 문제를 대상으로 하므로 키 입력 오류에 대해서는 정확한 추천이 어렵고, 유사도 순위에 따른 다양한 추천이 불가능하다는 단점이 있다.

n-그램 방법은 언어 독립적으로 가장 간단하게 사용할 수 있는 방법으로 간단하게 단어 간의 유사도를 계산할 수 있다는 장점이 있지만, 길이가 짧은 단어에 대해서는 낮은 정확률을 보인다는 단점이 있다.

편집 거리에 기반을 둔 방법으로 NHN(2008)[6], 노강호(2010)[7]가 있다. NHN(2008)[6]은 기존의 편집 거리 방법을 수정하여, 치환 비용을 주변에 위치한 키에 따라 가변적으로 계산함으로써 인접키 입력 오류를 고려하였다. 노강호(2010)[7]는 한글의 특성인 하나의 음절이 2~5개의 자소로 구성된다는 점을 편집 거리 방법에 반영하였다. 이 방법은 하나의 음절이 다른 경우와, 음절의 일부 자소만 다른 경우를 구분하여 편집 비용을 다르게 계산하였다. 그러나 스마트폰의 키패드로 입력된 문자열은 그 특성상 입력 단위가 하나의 자소가 아닐 수 있다. 스마트폰의 특성상 입력 공간이 협소하기 때문에 많은 단말기에서 자소를 분리하여 조합하는 방식이나 하나의 키에 몇 개의 자소를 할당하는 방식을 취하고 있다[8]. 예를 들어 천지인 키패드의 경우 ‘ㅏ’를 입력하기 위해서는 ‘ㅣ’와 ‘·’을 눌러야 한다.

본 논문에서는 유사 단어의 순위화에 유리한 편집 거리 방법을 기반으로 다양한 스마트폰 키패드 방식에 적용 가능한 유사도 계산 방법을 제안한다. 제안 방법은 편집 거리 계산 단위를 기존의 한글 자소, 음절 단위에서 가상 키패드의 키 하나로 본다. 이 방법은 실제 키패드를 입력하면서 발생하는 인접키, 연속 입력, 입력 생략, 입력 순서 등의 오류를 직접적으로 반영할 수 있다.

III. 유사 단어 순위화

1. 시스템 개요

스마트폰에서 많이 사용되는 터치스크린 기반의 가

상 키패드를 이용한 문자열 입력은 물리적인 키패드를 이용한 입력에 비해 오타가 포함될 가능성이 크다. 그래서 스마트폰에서 연락처, 주소 등을 검색할 때 사용자가 원하는 결과를 얻지 못하는 일이 종종 발생한다. 본 논문에서는 이와 같은 경우에 적용할 수 있도록 오류가 포함된 사용자 입력에 대하여 유사한 단어를 순위화하기 위한 유사도 계산 방법을 제안한다.

2. 시스템 활용 분야

제안하는 방법은 기기 내부에 저장된 정보와 같은 이미 존재하는 정보를 검색할 때 사용할 수 있다. 예를 들어 SMS나 노트, 이메일(e-mail) 등의 어플리케이션에서 이전에 사용된 단어를 찾는 모듈에 사용될 수 있고, 내비게이션 어플리케이션에서는 지명 정보를 검색하는 모듈에 사용될 수 있다. [그림 1]은 제안 방법을 사용하여 유사 단어 검색을 지원하는 연락처 검색 어플리케이션을 구현했을 때의 구성도이다. 사용자 입력과 후보를 3.4절의 입력 문자열 변환 방법으로 키 입력열로 변환을 하고, 3.5절의 유사도 계산 방법으로 점수를 계산하여 순위화 한다.

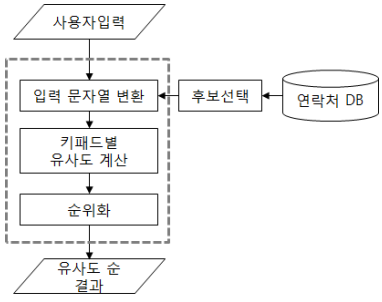


그림 1. 유사 단어 지원 연락처 검색 어플리케이션 구성도

제안 방법은 기본적으로 두 개의 단어가 입력됐을 때 두 단어의 유사도를 계산하는 방법이다. 제안 방법의 기본이 되는 편집거리 방법은 계산량이 많은 연산이기 때문에 모든 단어에 대하여 편집거리 연산을 하는 것은 문제가 있다. 그러므로 제안 방법을 이용하여 어플리케이션을 구현할 때 유사도 계산을 할 후보를 선택하는 부분이 필요하다. 그래서 본 논문에서는 비교적 계산량이 적은 음절 바이그램(bigram) 기반 정보검색 시스템을 이용하는 것으로 가정하고, 검색 결과 상위 n개의 단어 후보를 추출하여 유사도를 계산하였다.

3. 유사도 계산 알고리즘

3.1 스마트폰의 가상 키패드 분류

제안 방법에서는 입력 문자열을 가상 키패드의 실제 입력 키열로 변환하여 유사도 계산을 한다. 이때 대상이 되는 가상 키패드는 각각에 맞게 구현하기 위해서는 많은 종류가 존재한다. 그래서 본 논문에서는 가상 키패드의 종류를 키의 배치/할당 방식에 따라 크게 세 가지로 나누었다.

그림 2. FQ방식 키패드의 예

그림 3. 천지인 키패드

그림 4. 모아키 키패드

첫 번째는 [그림 2]의 아이폰(iPhone)의 키패드와 같이 키보드의 키 배열을 그대로 옮겨서 사용한 방식(full qwerty, 이하 FQ)이고, 두 번째는 [그림 3]의 천지인 키패드와 같이 연속으로 키를 누름으로써 “ㄱ/ㅋ/ㄲ”과 같이 다양한 키를 입력하도록 한 방식(key combination, 이하 KC)이다. 마지막으로 세 번째는 [그림 4]의 모아키 키패드와 같이 키를 누르면 팝업으로 모음의 획과 같은 추가 입력을 할 수 있도록 한 방식(pop up, 이하 PU)이다. FQ 방식의 키패드는 키보드의 배열을 그대로 옮겨왔기 때문에 처음 사용하는 사람도 쉽게 사용할 수

있다는 장점이 있지만, 하나의 화면에 많은 키를 배치해서 키 하나하나의 크기가 작다. 그래서 키를 입력할 때 인접한 다른 키가 눌릴 가능성이 크다. KC 방식의 키패드는 키 하나하나의 크기는 크지만, 하나의 자소를 입력하기 위해서 1~4번의 키를 입력해야 하기 때문에, 그 과정에서 키가 추가로 눌리거나 생략되어 다른 자소가 입력될 수 있다. PU 방식의 키패드는 모음 입력 방식이 KC 방식과 유사하여 모음에서 KC 방식과 같은 오류가 발생할 수 있다.

3.2 키를 입력할 때 발생하는 오류 유형

3.1절에서 살펴본 가상 키패드를 이용한 입력에서 발생할 수 있는 오류의 종류를 크게 묶으면 [표 1]과 같이 분류할 수 있다.

표 1. 키 입력 오류 종류

오류 종류	실제 예 (모아키 키패드)
연속 입력 오류	등잔산 → 등잔ㅅ산 (ㅅ 2번 입력)
입력 생략 오류	효성기업사 → 효싱기업사 (. 안눌림)
입력 순서 오류	장호원유치원 → 장호원유치위.ㄴ (., ㅣ 순서 바뀜)
인접키 오류	삼일전력공사 → 삼일전렬공사 (ㄱ/ㄹ 인접키)

가상 키패드의 특성상 입력 공간이 작고, 키 입력에 대한 반응속도가 물리적인 키와 다르기 때문에 키를 연속으로 누르거나, 입력이 인식되지 않는 상태에서 다음 키를 누를 수 있고, 누르는 순서를 잘못하거나, 옆의 다른 키를 누르기 쉽다. 이와 같은 오류들은 키패드의 종류에 따라 그 발생 빈도의 차이가 있을 뿐 대부분의 오류는 위의 4가지로 분류된다.

3.3 입력 문자열 변환

[표 1]의 키 입력 오류들을 살펴보면 한글의 자소 단위의 문제가 아닌 키 입력 단위의 문제임을 알 수 있다. 그러므로 이러한 오류를 처리하기 위해서는 입력 문자열을 키 입력열(key sequence)로 변환하는 과정이 필요하다. 키 입력열은 각각의 자소를 입력하기 위해 눌러야 하는 키 열을 말한다. 예를 들어 "감기"를 입력할 때, [그림 2]의 FQ방식은 "ㄱ-ㅏ-ㅁ-ㄱ-ㅣ"의 위치를 눌러야 한다. [그림 3]의 천지인 키패드의 경우 "4-1-2-0-0-4-1"의 위치를 눌러야 하고, [그림 4]의 모아키 키패드의 경우 "ㄱ-오른쪽드래그-ㅁ-ㄱ-팝업(ㅣ)"를 눌러야 한다. 이와 같이 실제로 누르게 되는 키에 각각의 코드를 할당하여, 입력 문자열을 키 입력열로 변환할 수 있다. 이와 같이 입력 문자열을 실제 입력 동작을 나타내는 키 입력으로 변환함으로써 다양한 키패드에서 나타나는 오류에 대하여 직접적인 표현이 가능하다. 예를 들어 "감기"와 "캄기"는 자소 단위 비교에서는 'ㄱ'과 'ㅋ'이 대치되는 것으로 보지만, 천지인 키패드에서 "4-1-2-0-0-4-1"과 "4-4-1-2-0-0-4-1"로 "ㄱ/ㅋ"이 연속으로 눌린 것으로 볼 수 있다.

3.4 유사도 계산

유사도 계산은 편집 거리 방법을 기반으로 스마트폰에서 발생하는 입력 오류를 반영하여 계산한다.

ED-Algorithm

$$edit(0,0)=0$$
$$edit(i,0)=i$$
$$edit(0,j)=j$$
$$edit(i,j)=\min\begin{bmatrix} edit(i-1,j)+1, \\ edit(i,j-1)+1, \\ edit(i-1,j-1)+1 \end{bmatrix}$$

그림 5. 편집 거리 알고리즘

[그림 5]는 편집 거리 알고리즘이다. 일반적으로 편집 거리 방법은 삽입/삭제/치환에 대하여 각각 1점씩 주어 계산한다. 본 논문에서는 일반적인 편집 거리 방법에서 삽입/삭제/치환 비용을 사용자 입력 오류를 고려하여 수정하였다. [그림 6]은 [표 1]의 오류들을 고려한 제안 알고리즘이다.

mED-Algorithm

$$edit(i,j)=\min\begin{bmatrix} edit(i-1,j)+insert(s_{i-1},s_i,t_j) \\ edit(i,j-1)+delete(s_i,t_j,t_{j+1}) \\ edit(i-1,j-1)+r(s_i,s_{i+1},t_j,t_{j+1}) \end{bmatrix}$$

$$insert(s_{i-1},s_i,t_j)=\begin{cases} if\ \ s_{i-1}=s_i\ \&\ s_i=t_j\ \ then\ \ 연속\ 키입력\ \ 오류\ \ 비용 \\ else\ \ 10 \end{cases}$$

$$delete(s_i,t_j,t_{j+1})=\begin{cases} if\ \ s_i=t_{j+1}\ \&\ s_i=t_j\ \ then\ \ 키\ 안눌림\ \ 오류\ \ 비용 \\ else\ \ 10 \end{cases}$$

$$r(s_i,s_{i+1},t_j,t_{j+1})=\begin{cases} if\ \ s_i=t_j\ \ then\ \ 0 \\ if\ \ s_is_{i+1}=t_{j+1}t_j\ \ then\ \ 입력순서\ \ 오류\ \ 비용, \\ if\ \ 인접키배열\ \ O\ \ then\ \ 인접\ 키배열\ \ 비용 \\ if\ \ 인접키배열\ \ X\ \ then\ \ 10 \end{cases}$$

그림 6. 제안 알고리즘

제안 알고리즘에서는 기본적인 편집 거리 알고리즘에 [표 1]의 오류들을 판단하는 부분을 추가하였다. 연속 눌림 오류는 삽입 연산에서 정답과 동일한 키가 입력된 것으로 판단하였다. 예를 들어 천지인 키에서 'ㅇ'을 입력하는데 'ㅁ'이 입력되는 경우, 즉 입력 문자열 변환에서 '0'이 '00'으로 눌린 경우이다. 키 생략 오류는 삭제 연산에서 연속 눌림과 반대의 경우로 판단하였다. 예를 들어 'ㅁ'이 'ㅇ'으로 입력되는 '00'이 '0'으로 눌린 경우이다. 입력 순서 오류는 치환 연산에서 입력과 정답이 서로 반대인 경우로 판단하였다. 예를 들어 'ㅏ'가 'ㅓ'로 눌린 경우, 즉 '12'가 '21'로 눌린 경우이다. 인접 키 눌림 오류는 정답 키에서 인접한 다른 키가 눌린 경우로, 정답 키로부터 거리 1의 키가 눌렀을 경우로 판단하였다. 이와 같은 오류들의 경우에 대하여 기본 삽입, 삭제, 치환의 비용보다 더 적은 비용을 주도록 하였다. 삽입/삭제 비용의 계산은 수식 (1)과 같이 하였다.

$$\text{if}(P(type) \geq 0.1)\ then \lfloor 10/MAX(-\log_2 P(type), 1) \rfloor \\ else \quad 10 \qquad (1)$$

수식 (1)에서 $-\log_2 P(type)$는 오류 유형별 정보량이다. 정보량이 클수록 편집 비용이 작아지도록 기준 값 10(기본적인 삽입, 삭제, 치환 값)을 정보량으로 나눠서 편집 비용으로 사용하였다. $P(type)$은 해당 오류가 나타날 확률로 각각의 오류로 판단된 수를 오류가 속한 연산의 횟수로 나누어 계산한다. 예를 들어 연속 입력 오류는 삽입 연산에서 판단하므로 연속 입력 오류로 판단된 것의 수를 삽입 연산 횟수로 나눈다. 이때 출현 비율이 너무 적은 경우 해당 키패드에서 의미 있는 오류 유형으로 볼 수 없다고 판단되어, 출현 비율이 10% 이상이 되는 경우만 오류 판단을 수행하였다. 이때, 스마트폰의 특성상 소숫점 연산(floating-point operation) 하드웨어가 없기 때문에, 소숫점 연산을 없애기 위해서 정수화 하였다. 치환 연산의 경우 인접 키 입력 오류를 처리하기 위해서 키로부터 거리 1에 있는 키들에 대하여 더 적은 치환 비용을 갖도록 하였다. 이때 손가락 근육의 특성상 펴지기가 쉽기 때문에 쉽게 갈 수 있는 상단과 바깥쪽 방향의 값을 잘 가지 않는 하단, 안쪽 방향의 값보다 적은 값을 주었다. 치환 연산의 인접 키 값과 입력 순서 비용은 실험적으로 결정하였다.

표 2. 키패드별 편집 비용

키패드	연속 입력	입력 생략	입력 순서	인접키사용
쿼티(FQ방식)	10	10	10	O
천지인(KC방식)	4	3	10	X
모아키(PU방식)	10	10	3	O

IV. 실험 결과 및 분석

1. 실험 환경

실험을 위해 대학생 15명이 터치패드 방식의 스마트폰으로 입력한 POI 지명 데이터를 사용하였다. 수집된 데이터는 [표 3]과 같다. [표 3]에서 정답은 실제 지명데이터와 사용자 입력이 동일한 경우이고, 오답은 실제 지명데이터와 사용자 입력이 자소 하나라도 틀린 경우이다. 오답률은 전체 단어에서의 오답의 수로 계산하였다.

표 3. 실험 데이터

키패드	전체 (단어)	정답 (단어)	오답 (단어)	오답률(%)	비고
쿼티 FQ방식	1,000	604	396	39.6%	200단어 × 5명
천지인 KC방식	1,000	559	441	44.1%	200단어 × 5명
모아키 PU방식	1,000	699	301	30.1%	200단어 × 5명

수집된 데이터는 동일한 단어 1,000개를 각각의 키패드에서 입력한 것이다. 데이터 크기 문제로 실제와는 차이가 있겠지만, 수집된 데이터에서는 모아키의 경우가 30.1%의 오류율로 세 가지 방식의 키패드에서 가장 좋은 결과를 보였고, 천지인 방식이 44.1%로 가장 오류율이 높았다. 이것은 모음 입력시 세 가지 획(ㅡ, ., ㅣ)의 조합으로 입력하기 때문에, 복잡한 획의 모음(ㅐ, ㅖ, ㅘ, ㅙ 등) 입력에서 많은 오류가 발생한 것으로 보인다.

실험에 사용된 대상은 수집된 데이터의 정답을 포함한 총 25,000개 POI 지명 데이터를 대상으로 음절 바이그램 기반 정보검색 시스템을 이용하여 수집된 데이터 질의로 줬을 때의 결과 상위 n개이다.

제안 시스템의 성능을 평가하기 위해서 R@n(Recall at n)과 MRR(Mean Reciprocal Rank)을 이용하였으며 계산에 사용된 식은 각각 수식 (2)와 수식 (3)이다.

$$R@n = \frac{\text{상위 } n\text{개에 포함된 정답 수}}{\text{전체 질의 수}} \quad (2)$$

$$MRR = \sum_{i=1}^{N} \frac{1}{Rank(q_i)} \quad (3)$$

수식 (3)에서 N은 전체 질의 수이고, q_i는 i번째 질의이다.

2. 실험 결과

[표 4]는 각 키패드 방식에서 오류처리로 얻는 성능 향상을 보여준다. 표의 B는 기본적이 편집 거리 방법, T는 키 입력 열 변환, C는 연속 눌림・키 생략 처리, O는 입력 순서 처리, N은 인접키 처리이다. 실험에 사용된 데이터는 바이그램 정보검색 시스템을 이용한 상위 30개의 결과를 사용하였다. 아래 계산된 값들은 정보검색 시스템으로 후보를 뽑았을 때, 정답이 있는 경우에 대하여 계산하였다.

표 4. 오류 처리 실험 결과

키패드	조합	MRR	R@1	R@2	R@3
FQ	B	0.9427	0.9204	0.9511	0.9755
	B+T	0.9435	0.9204	0.9541	0.9786
	B+T+N	0.9572	0.9419	0.9695	0.9878
KC	B	0.8930	0.8366	0.9058	0.9502
	B+T	0.9234	0.8864	0.9307	0.9640
	B+T+C	0.9237	0.8864	0.9307	0.9668
PU	B	0.9172	0.8716	0.9396	0.9660
	B+T	0.9161	0.8755	0.9396	0.9547
	B+T+O	0.9213	0.8830	0.9434	0.9623
	B+T+N	0.9149	0.8755	0.9358	0.9547
	B+T+O+N	0.9222	0.8868	0.9396	0.9623

모든 방식에서 공통적으로 처리한 키 입력열 변환은 키패드의 특성상 FQ방식에서는 큰 성능 향상이 없었지만 KC방식과, PU방식에서는 비교적 큰 성능향상이 있었다. 인접키 처리의 경우 FQ방식과, PU방식에서 사용되었지만, 키패드 방식의 특성상 FQ방식에서 좋은 성능을 보였고, PU방식에서는 단독으로 사용될 경우 오히려 성능을 떨어뜨렸다. 그러나 입력 순서 오류 처리와 함께 사용할 경우 MRR과 R@1에서 성능 향상이 있었다. 각 키패드 방식별 처리에 사용되지 않는 처리 방법들은 성능 향상이 없거나, 오히려 성능을 떨어뜨리는 결과를 보였다. 이러한 결과는 오류를 판단하기 위한 알고리즘이 완벽히 오류만 고르는 것이 아니라, 오류가 아닌 경우도 고르기 때문인 것으로 보인다.

[표 5]는 제안방법과 일반적인 편집 거리를 이용한 방법의 비교 실험 결과이다.

표 5. 비교 실험 결과

		MRR	R@1	R@2	R@3
FQ	Res_IR	0.7791	0.7033	0.7859	0.8471
	Res_ED	0.9427	0.9204	0.9511	0.9755
	Res_Proposal	0.9572	0.9419	0.9694	0.9878
KC	Res_IR	0.7336	0.6427	0.7258	0.7950
	Res_ED	0.8930	0.8366	0.9058	0.9502
	Res_Proposal	0.9237	0.8864	0.9307	0.9668
PU	Res_IR	0.7932	0.7283	0.8000	0.8415
	Res_ED	0.9172	0.8716	0.9396	0.9660
	Res_Proposal	0.9222	0.8868	0.9396	0.9622

[표 5]에서 Res_IR은 정보검색 시스템 결과를 그대로 사용한 결과이고, Res_ED는 단어를 초성, 중성, 종성으로 분해하여 일반적인 편집 거리 방법을 이용하여 재순위화한 결과, Res_Proposal은 제안 방법을 이용하여 재순위화한 결과이다. [표 5]의 결과를 보면 PU 방식의 R@3을 제외한 모든 경우에서 제안 시스템이 정보검색 시스템의 결과, 일반적인 편집 거리 방법을 사용하는 것 보다 좋은 성능을 보였다. 성능 향상 정도를 보면 MRR의 경우 FQ에서 0.0145, KC에서 0.0307, PU에서 0.0050의 성능 향상이 있었고, R@n의 경우 n=1에서 평균 0.0288, n=2에서 평균 0.0144, n=3에서 0.0083의 성능 향상이 있었다. 비록 성능이 크게 높아진 것은 아니지만 간단한 전처리로 성능향상이 있었다는 점에서 의미 있는 결과라고 생각된다.

V. 결론 및 향후 과제

본 논문에서는 스마트폰 입력 환경에서 사용자 입력

오류를 고려하여 유사한 단어순으로 순위화하는 방법을 소개하였다. 스마트폰 입력 환경을 고려하기 위해 가상 키패드의 입력 방식을 FQ, KC, PU의 세 가지 방식으로 분류하고, 각각의 방법을 코드화하는 방법을 소개하였다. 코드화된 문자열간의 유사도 비교를 위해 편집 거리 방법을 각각의 입력 방식에서 자주 발생하는 오류를 반영하여 수정하였다. 모아키, 천지인, 쿼티 입력 방식에서 실험한 결과 MRR과 R@n에서 작은 성능 향상을 확인할 수 있었다. 비록 큰 성능 향상은 아니지만 간단한 전처리만으로도 성능을 향상시킬 수 있기 때문에 의미 있는 결과라고 생각된다. 향후 과제로 실험 데이터를 확장 구축하고, 오류 유형의 판단을 좀 더 명확하게 판단하는 알고리즘을 연구할 예정이다. 또한 편집 비용의 결정을 이론적으로 명확히 하여 연구의 완성도를 높일 예정이다.

참 고 문 헌

[1] http://www.kcc.go.kr/user.do?mode=view&page=P05030000&dc=K05030000&boardId=1042&boardSeq=30991

[2] 오형용, "모바일기기를 위한 소프트키보드 인터페이스 디자인", 한국콘텐츠학회논문지, 제7권, 제6호, pp.79-88, 2007.

[3] 강병주, 이재성, 최기선, "외국어 음차 표기의 음성적 유사도 비교 알고리즘", 정보과학회논문지(B), 제26권, 제10호, pp.1237-1246, 1999.

[4] 고숙현, 이재성, "문맥을 고려한 유사 외래어 검출 알고리즘의 성능 향상", 제19회 한글 및 한국어 정보처리 학술대회, pp.114-121, 2007.

[5] 김효경, *한글 철자 검사를 위한 음성적 유사도 계산 알고리즘*, 성균관대학교 석사학위 논문, 2006.

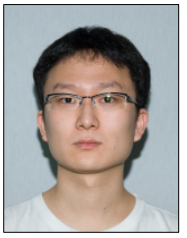
[6] NHN, *키 배열 정보를 이용한 단어 추천 방법 및 그 시스템*, 대한민국특허청, KR-A-10-2008-0045530, 2008.

[7] 노강호, 김진욱, 김은상, 박근수, 조환규, "한글에 대한 편집 거리 문제", 정보과학회논문지:시스템 및 이론, 제37권, 제2호, pp.103-109, 2010.

[8] 홍성용, 이진영, 정태의, "콘텐츠 저작용 다국어 입력기", 한국콘텐츠학회논문지, 제2권, 제1호, pp.50-62, 2004.

저 자 소 개

송 영 길(Yeong-Kil Song) 정회원



- 2008년 2월 : 강원대학교 컴퓨터학부(공학사)
- 2010년 2월 : 강원대학교 컴퓨터정보통신공학과(공학석사)
- 2010년 2월 ~ 현재 : 강원대학교 컴퓨터정보통신공학과 박사과정

<관심분야> : 유사 단어 검색, 멀티모달 정보검색, 한글 자동 띄어쓰기

김 학 수(Hark-Soo Kim) 정회원

- 1996년 2월 : 건국대학교 전자계산학과(공학사)
- 1998년 2월 : 서강대학교 컴퓨터학과(공학석사)
- 2003년 2월 : 서강대학교 컴퓨터학과(공학박사)
- 2004년 ~ 2005년 : CIIR in UMass, Amherst (박사후연구원)
- 2005년 ~ 2006년 : 한국전자통신연구원(선임연구원)
- 2006년 ~ 현재 : 강원대학교 컴퓨터정보통신공학전공 교수

<관심분야> : 자연어처리, 대화시스템. 정보검색, 질의응답시스템