http://dx.doi.org/10.5392/JKCA.2012.12.03.034

PAM 행렬 모델을 이용한 음소 간 유사도 자동 계산 기법

Automatic Inter-Phoneme Similarity Calculation Method Using PAM Matrix Model

김성환, 조환규
부산대학교 컴퓨터공학과

Sung-Hwan Kim(sunghwan@pusan.ac.kr), Hwan-Gue Cho(hgcho@pusan.ac.kr)

요약

두 문자열 간의 유사도를 계산하는 문제는 정보 검색, 오타 교정, 스팸 필터링 등 다양한 분야에 응용될 수 있다. 동적 계획법 기반의 유사도 계산 방법을 통하여 한글 문자열의 유사도 계산을 위해서는 우선 음소 간의 유사도에 대한 정의가 필요하다. 그러나 기존의 방법들은 수동적 설정에 의한 유사도 점수를 사용하고 있다는 한계점이 있다. 본 논문에서는 PAM(Point Accepted Mutation) 행렬과 유사한 확률 모델을 이용하여 변형 단어 집합으로부터 음소 간의 유사도를 자동적으로 계산하는 기법을 제안한다. 제안 기법은 주어진 변형 단어의 집합 내 유사한 단어 쌍을 찾아 문자열 정렬(Text Alignment)을 수행함으로써 음소 변형 규칙을 도출하고, 이로부터 각 음소 쌍의 상호 변형 빈도에 따른 유사도 점수를 계산한다. 실험 결과 특이도(Specificity) 77.2~80.4% 수준에서 불일치 여부에 따른 단순 점수 부여 방식에 비해서는 10.4~14.1%, 수동으로 음소 간 유사도를 직접 설정하는 방식에 비해서는 8.1~11.8%의 민감도(Sensitivity) 향상이 있음을 확인하였다.

■ 중심어 : | 음소 유사도 | PAM 행렬 | 문자열 정렬 | 단어 필터링 |

Abstract

Determining the similarity between two strings can be applied various area such as information retrieval, spell checker and spam filtering. Similarity calculation between Korean strings based on dynamic programming methods firstly requires a definition of the similarity between phonemes. However, existing methods have a limitation that they use manually set similarity scores. In this paper, we propose a method to automatically calculate inter-phoneme similarity from a given set of variant words using a PAM-like probabilistic model. Our proposed method first finds the pairs of similar words from a given word set, and derives derivation rules from text alignment results among the similar word pairs. Then, similarity scores are calculated from the frequencies of variations between different phonemes. As an experimental result, we show an improvement of 10.1%~14.1% and 8.1%~11.8% in terms of sensitivity compared with the simple match-mismatch scoring scheme and the manually set inter-phoneme similarity scheme, respectively, with a specificity of 77.2%~80.4%.

■ keyword : | Inter-phoneme Similarity | PAM Matrix | Text Alignment | Word Filtering |

* 본 연구는 2009년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행되었습니다(2011-0003157)

접수번호 : #111229-003
접수일자 : 2011년 12월 29일
심사완료일 : 2012년 02월 13일
교신저자 : 조환규, e-mail : hgcho@pusan.ac.kr

I. 서 론

1. 문자열 유사도 판정 문제

서로 다른 문자열 간의 유사도를 판정하는 문제는 다양한 분야에서 응용될 수 있다[1]. 정보 검색 시 오타나 외래어 음역 등의 문제로 잘못 입력된 질의어를 색인된 올바른 질의어로 교정할 수 있으며, 생물정보학에서는 진화추적이나 유전자의 기능을 발견하기 위하여 서열 간의 유사성을 비교하기도 한다. 이처럼 정보와 관련된 모든 분야에서 문자열의 유사도 판정 문제는 빠질 수 없는 핵심적인 역할을 한다. 특히 불건전, 불필요 정보를 차단하기 위한 단어 필터링 시스템에서 문자열 간의 유사도 판정은 매우 중요하다. [표 1]에서 확인할 수 있듯이 본래 필터링 대상이 되는 단어를 중심으로 다양한 형태의 유사한 변형이 존재한다. 기존의 완전 일치(Exact Matching) 탐색 기법으로 단어 필터링을 수행하기 위해서는 파생 형태들을 데이터베이스에 수록하여야 한다[2][3]. 그러나 현실적으로 그 경우의 수가 매우 다양하며 또한 앞으로 어떠한 형태의 신형 변종 단어가 발생할지 예측할 수 없기 때문에 불가능하다. 만약 가능하다고 가정하더라도 수많은 유사 단어들을 필터링하기 위해 아무런 전처리 없이 데이터베이스에 수집하는 것은 매우 비효율적인 방법이다. 따라서 데이터베이스 내 금지어들 중 질의어와 가장 가까운 단어는 무엇인지, 또한 해당 단어들 간의 유사도는 어느 정도인지를 판단하는 것이 필수적이다.

표 1. 단어 변종 파생 형태의 예. 하나의 단어로부터 파생되는 모든 변형된 단어를 관리하기에는 그 경우의 수가 지나치게 다양하다.

원래 단어	변형 형태
멍청이	멍청이, 멍충이, 멍충이, 멍청ㅇㅣ
대출	대출, 대출, 뎨출, 대출, ㄷH출
새끼	색히, 새퀴, ㅅH끼, 색퀴, 색휘

2. 음소 간 유사도 문제

한글은 음소 문자이기 때문에 한글로 이루어진 문자열 간의 유사성을 판단하기 위해서는 우선 음소 간의 유사도에 관한 문제가 해결되어야 한다. 특히 단어 필터링 시스템을 우회하고자 하는 악의적인 사용자는 유사한 형태의 음소를 이용하여 기존의 금지어를 교묘하게 변형해나간다. 따라서 음소 간의 유사도를 통하여 유사 단어를 판별하는 것이 매우 중요하다. 기존 시스템에서는 직접 직관에 의하여 유사도를 책정하는 방식들이 주로 사용되었다[4-6]. 그러나 [표 2]에서 볼 수 있듯이 실제 변형에 이용되는 음소는 한글의 음소에만 국한된 것이 아니라 유사한 형태의 외국어 문자, 특수문자들이 포함되어 있으며, 변형 형태 역시 매우 다양하다. 변형 형태의 가짓수가 증가할수록 음소간의 유사도를 직접 설정해야 하는 분량이 증가하고, 이는 음소 간 유사도를 일관적으로 구성 및 유지하기 위해 지나치게 큰 비용이 필요하게 됨을 의미한다.

표 2. 음소 변형 형태의 예. 각 변형 형태에 따른 수동 유사도 설정은 작업이 까다롭고 유사도 점수에 대한 일관성 문제를 수반한다.

음소	변형 형태
ㄱ	ㄱ ㄲ ㅋ ㉠ ㄱ つ づ G K
ㄴ	ㄴ L ㄴ ㉡ ㄴ じ (L) N
ㅐ	ㅐ ㅐ ㅔ ㅖ ㅙ ㅞ H
ㅣ	ㅣ ㅢ ㅟ i 1 1 I ! !

3. 본 논문의 기여

본 논문에서는 주어진 변형 단어들의 집합으로부터 유사 단어 쌍을 찾아내고, 해당 단어 쌍들 간의 문자열 정렬을 통하여 음소 변형 규칙을 도출하여, 음소 간의 변형 규칙의 빈도를 통하여 최종적으로 음소 간의 정량적인 유사도를 계산하는 기법을 제안한다. 생물정보학에서 흔히 사용되는 문자 간 점수 행렬인 PAM 행렬의 모델을 목적에 맞게 다소 변형하여 사용하였다.

II. 관련 연구

1. 한글 문자열 및 음소 유사도 측정 방식

한글 문자열의 유사도 측정 방식은 주로 음소를 해체하여 나열한 후 잘 알려진 문자열 간 유사도 측정 알고리즘을 이용한다[5-7]. 또한 한글의 특성에 맞게 음소

및 음절 편집거리를 새로 정의하여 측정하는 방식도 제안되었으며[4], 모바일 환경과 같이 특수한 환경에서의 편집거리 알고리즘을 제안한 연구도 있다[9].

문자열 유사도를 측정하는 방법은 주로 동적 계획법(Dynamic Programming)을 이용한 서열 정렬이나 편집 거리 알고리즘들이 사용된다. 전역 정렬(Global Alignment)을 응용한 알고리즘의 경우 주어진 유사도 함수 $S: \Sigma \times \Sigma \rightarrow Z$ (단, Z는 정수의 집합)에 대하여 두 문자열 x, $y \in \Sigma^*$의 유사도 점수 $AlignScore(x,y;S)$는 다음과 같이 계산된다.

$$AlignScore(x,y;S) = GA(|x|,|y|;x,y,S) \quad (1)$$

$$GA(0,0;x,y,S) = 0 \quad (2)$$

$$GA(0,j;x,y,S) = i \times Gap \quad (3)$$

$$GA(0,j;x,y,S) = j \times Gap \quad (4)$$

$$GA(i,j;x,y,S) = \max \begin{Bmatrix} GA(i-1,j-1;x,y,S) + S(x[i],y[j]) \\ GA(i-1,j;x,y,S) + Gap \\ GA(i,j-1;x,y,S) + Gap \end{Bmatrix} \quad (5)$$

수식 (5)에서도 알 수 있듯이 두 문자(음소)간의 점수를 결정하는 유사도 함수 S에 따라 유사도 점수의 결과값이 바뀐다. 문자열 유사도 계산 방법들이 도달한 공통의 과제는 이러한 음소 간의 유사도를 어떻게 반영할 것인가에 관한 문제이다. 유사 음소 군을 정의한 후 같은 음소 군 내의 음소 간에는 서로 다른 음소 군 간의 음소들보다 낮은 페널티를 주는 방법[4]이 있으며, 음소 쌍 별로 유사도 점수를 별도로 부여하는 방법[5][6]도 제안되어 왔다.

1장에서도 언급했듯이 기존의 방법들은 관리자의 직관에 의한 점수 부여 과정이 반드시 필요하기 때문에 음소의 변형 가짓수가 증가하는 경우 과다한 작업량, 점수의 일관성 등의 문제가 수반된다.

2. PAM(Point Accepted Mutation)행렬

생물정보학에서는 염기서열의 비교를 위하여 일찍부터 염기 서열을 구성하는 문자들 간의 유사성을 비교하기 위한 노력이 진행되어 왔다[8]. 대표적인 것이 PAM(Point Accepted Mutation) 행렬이다. PAM 행렬은 가까운 거리의 염기서열 간의 비교를 위해 만들어진 유전자 간의 유사도 점수 행렬이다. 그러나 PAM행렬에서 학습을 위하여 사용되는 유사 구간은 갭이 없는 정렬(Ungapped Alignment)를 이용하여 서로 대응시킬 수 있는 충분히 긴 길이의 서열임에 비하여, 단어 필터링 시스템에서의 문자열들은 길이가 매우 짧기 때문에 변형, 특히 삽입이나 삭제가 일어나는 경우 대응관계의 변형이 상대적으로 크게 발생한다. 따라서 본 논문에서는 PAM 행렬의 기법을 기반으로 하되 소기의 목적에 적합하도록 일정 부분 변형하여 응용하도록 한다.

III. 제안 기법

1. 입출력 정의

본 논문에서 다룰 음소 간 유사도 점수를 계산하는 문제는 [표 3]과 같이 음소 집합 Σ와 Σ의 원소로 이루어진 문자열들의 집합 W가 주어졌을 때, 이로부터 $|\Sigma+1| \times |\Sigma+1|$ 크기의 유사도 점수 행렬 $Score$를 구하는 것이 목적이다. 유사도 점수 행렬 $Score$의 i행 j열 성분은 문자열 정렬 시에 Σ_i와 Σ_j가 대응되는 경우 부여되는 점수이다. 이 때, 음소 집합 Σ의 원소 외에 Gap을 위한 특수 문자와 아직 알려지지 않은 문자에 대한 점수가 필요한데, 이들을 하나로 취급하여 1개의 추가 문자가 필요하므로, 행과 열의 수가 $|\Sigma+1|$이 된다.

표 3. 음소 간 유사도 계산 문제의 입출력.

항목	내용
입력	Σ : 음소 집합 W: 문자열 집합
출력	$Score$: $\|\Sigma+1\| \times \|\Sigma+1\|$ 크기의 행렬

한편 W의 각 원소는 실제로는 Σ의 원소들로 이루어진 하나의 벡터이지만, 편의를 위하여 논문에 W의 원소를 표기할 때에는 완성된 단어를 사용하도록 한다. 즉, 실제로 입력되는 형태는 "ㅋㅏㅈㅣㄴㅗ"이더라도 논문상에서는 편의상 "카지노"로 표기한다.

2. 유사 단어 쌍 검출

먼저 주어진 문자열 집합 W 내에는 서로 유사한 문자열도 존재하지만 그렇지 않은 경우도 존재한다. 예를 들어 W={“바카라”, “박하롸”, “카지노”, “카ㅈI노”, “카쥐no”}인 경우 이는 두개의 부분 집합 W_1={“바카라”, “박하롸”}와 W_2={“카지노”, “카ㅈ1노”, “카쥐no”}로 분할할 수 있고, 이 때 같은 집합 내의 문자열 간에는 서로 유사하지만, W_1과 W_2간의 문자열들은 서로 상이하다는 사실을 알 수 있다.

문제는 이들을 어떻게 군집화 할 것이며, 군집화 하더라도 각각의 부분 집합 내에서의 문자열 변형 관계를 어떻게 정의할 것인가이다. 문자열이 변형되는 행태에 관하여 다음과 같은 가정을 내림으로써 문제를 단순화 시킬 수 있다.

가정 1. 주어진 문자열 집합 W내의 모든 문자열 w에 대하여 w와 유사한 다른 문자열이 적어도 하나 이상 존재한다.

가정 2. 각각의 문자열은 자신과 가장 가까운 다른 문자열로부터 변형된다.

가정 1과 가정 2를 통하여 문자열 $x \in W$에 대하여 x와 유사한 문자열들의 집합 $S_W(x)$를 다음과 같이 정의할 수 있다.

$$S_W(x) = \arg\min_{y \in W-\{x\}} \mathrm{Ed}(x,y) \qquad (6)$$

이 때, $\mathrm{Ed}(x,y)$는 두 문자열 x와 y 간의 Levenshtein 거리(편집 거리)를 의미한다. 즉, 각각의 문자열에 대하여 자신을 제외하고 편집 거리가 가장 짧은 문자열들의 집합이 해당 문자열이 변형될 수 있는 문자열 집합이라는 것을 의미한다. 앞서 든 예에서 각각의 단어 x에 대하여 $S_W(x)$를 구성해보면 [그림 1]과 같이 일종의 군집화가 되는 것을 확인할 수 있다. 이 때, S_W(“카쥐no”) = {“카지노”,“카ㅈ1노”}, S_W(“바카라”)={“박하롸”}이다.

이 때, 살펴볼 것은 [그림 1]의 그래프가 방향성을 가지기 때문에 S_W(“카쥐no”)에는 “카지노”가 있는 반면 S_W(“카지노”)에는 “카쥐no”가 존재하지 않는다. 이는 가정 2를 반영한 것으로 “카쥐no”가 “카지노”로부터 변형될 수는 있어도 “카쥐no”가 변형되어 “카지노”로 되지는 않음을 의미한다. 이를 통해 상호간에 변형이 빈번하게 일어나는 “카지노”-“카ㅈ1노”보다는 일방적인 변형관계인 “카지노”-“카쥐no”에 대한 가중치가 상대적으로 적게 작용하는 결과를 얻을 수 있다.

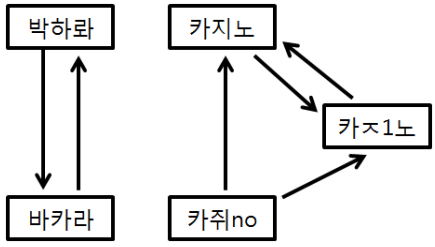

그림 1. 집합 W내의 가까운 단어들을 연결한 결과. 유사한 단어끼리 군집화 되는 효과를 얻었다.

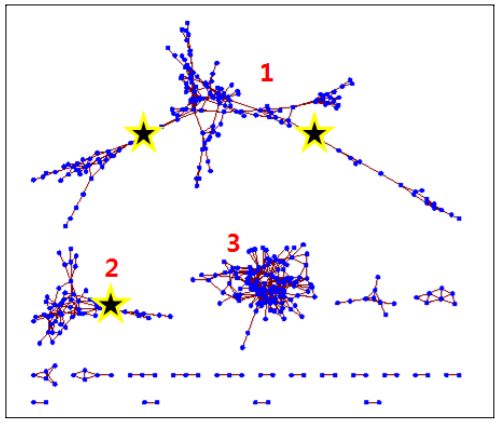

그림 2. 욕설 407 단어에서 가까운 단어 쌍을 찾아낸 결과. 우연히 중간 형태를 가지는 단어(1과 2의★표)들에 의하여 길게 늘어지는 모습을 보이며, 이들의 수가 증가할수록 성능에 악영향을 끼친다. 단어군 3은 한 클래스(“ㄱㅅㅋ”)의 단어만 군집화 되었다.

[그림 2]는 실제 변형 욕설 단어 407개에서 편집거리에 따른 가까운 단어 쌍을 찾아낸 결과이다. 욕설 단어의 선정성과 폭력성으로 인하여 구체적인 단어의 내용은 언급하지 않았지만 전체적으로 군집화가 되는 모습을 확인할 수 있다. 단어군 1과 2의 ★표는 “새끼”등과 같이 욕설이 가지는 일반적인 접미사로 인하여 서로 다

른 욕설들이 연결되는 모습을 나타낸다. 전체 욕설의 개수가 충분하기 때문에 성능에는 큰 영향을 끼치지는 않지만 이러한 단어가 증가하게 되면 유사하지 않아야 하는 두 단어가 서로 연결되는 결과를 초래하여 유사도 점수를 계산함에 있어서 성능에 악영향을 미칠 수 있다. 단어군 3의 경우에는 같은 클래스(“ㄱㅅㄲ”)의 단어가 군집화된 것을 볼 수 있는데, 같은 클래스 내에서는 단어 간 연결 횟수가 많을수록 유사도 점수의 성능이 높아질 것으로 예측할 수 있다.

3. 대응 음소 쌍 검출 및 출현 빈도 계산

유사 단어 집합을 구한 이후에는 각각의 유사 단어 쌍에 대하여 문자열 정렬(Alignment)을 수행하여 대응되는 음소의 쌍을 검출할 수 있다. 예를 들어 “카지노”와 “카쥐no”를 정렬하면 [그림 3]과 같은 결과를 얻을 수 있다. 정렬 결과를 통하여 불일치하는 음소 쌍 “ㅣ”-“ㅟ”, “ㄴ”-“n”, “ㅗ”-“o” 간의 대응 관계를 알 수 있다. 여러 단어 쌍에 걸쳐서 대응되는 횟수가 많은 음소 쌍일 수록 두 음소 간에 변형이 빈번하게 발생함을 의미한다. 이러한 음소 간의 대응 횟수를 카운트하여 최종적으로 음소 간의 유사도 점수를 정량적으로 계산하는데 사용한다.

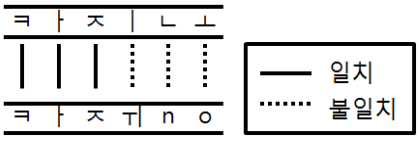

그림 3. “카지노”와 “카쥐no”의 정렬(Alignment)결과. 불일치 음소 쌍 간의 변형이 이루어졌음을 알 수 있다.

편집 거리를 이용하여 정렬할 단어 쌍을 선별한 이후이므로 두 문자열을 구성하는 음소의 개수가 서로 같다면 정렬 결과는 유일하며 각 음소가 차례대로 대응된다. 그러나 “바카라”(음소 6개), “박하라”(음소 7개)와 같이 음소의 개수가 서로 다른 경우에는 하나 이상의 정렬 결과가 발생할 수 있다. “바카라”와 “박하롸”를 정렬한 결과는 [그림 4]와 같다.

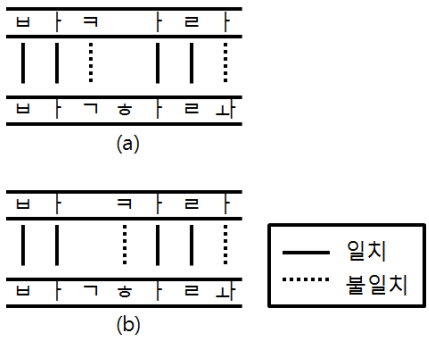

그림 4. “바카라”와 “박하라”의 정렬 결과. 여러 개의 정렬 결과를 얻을 수 있다. (a)에서는 “ㅋ”-“ㄱ”이, (b)에서는 “ㅋ”-“ㅎ”이 대응되었다.

정렬 결과를 보면 알 수 있듯이 [그림 4]-(a)에서는 “ㅋ”과 “ㄱ”이 서로 대응되었고, [그림 4]-(b)에서는 “ㅋ”과 “ㅎ”이 서로 대응되었음을 알 수 있다. 혹자는 “ㅋ”은 “ㅎ”보다는 “ㄱ”과 유사하니 [그림 4]-(a)의 결과가 올바르지 않겠냐고 반박할 수 있겠지만 본 논문의 유사한 단어가 포함된 집합 이외의 아무런 지식이 주어지지 않았을 때 음소 간의 유사도를 자동적으로 계산하기 위함이므로 [그림 4]-(a)와 [그림 4]-(b) 두 경우 모두 타당한 결과이다. 따라서 이러한 경우 각각의 경우를 모두 음소 대응 횟수에 산정하되, 가중치를 총 경우의 수로 나누도록 한다. 이러한 방식을 이용하면 “ㅋ”은 “ㄱ”과 “ㅎ”에 대하여 각각 0.5회 대응된 것이며 “ㅏ”와 “ㅘ”는 1회 대응된 것이 된다.

본 절에서 토의한 결과를 정리하면 다음과 같이 나타낼 수 있다. 두 문자열 x, y를 정렬(Alignment)한 결과 대응되는 음소의 쌍들을 다중집합(Multiset)의 집합으로 나타낸 결과를 $Align(x,y)$라 하자. 예를 들어

$Align$(“바카라”,“박하롸”) =
{{<ㅂ,ㅂ>,<ㅏ,ㅏ>,<ㅋ,ㄱ>,<#,ㅎ>,<ㅏ,ㅏ>,<ㄹ,ㄹ>,<ㅏ,ㅘ> },
{<ㅂ,ㅂ>,<ㅏ,ㅏ>,<#,ㄱ>,<ㅋ,ㅎ>,<ㅏ,ㅏ>,<ㄹ,ㄹ>,<ㅏ,ㅘ>}}

이다. 단, “#”은 Gap을 나타내는 특수한 문자이다. $Align(x,y)$이 위와 같이 정의되면 음소 α와 β 간의 대

응 횟수 $F_{\alpha,\beta}$는 [그림 5]의 GetPhonemePairsFreq를 이용 하여 구할 수 있다. 모든 $Align(x,y)$의 원소에 대하여 대응되는 음소 α, β에 대하여 대응 가짓수(즉, $|Align(x,y)|$)의 역수만큼 카운트 된다. $F_{\alpha,\beta}$와 $F_{\beta,\alpha}$가 동시에 업데이트 되므로 $F_{\alpha,\beta}=F_{\beta,\alpha}$이다.

Algorithm GetPhonemePairsFreq
Input W : 문자열 집합
Output F : 음소 쌍 대응 횟수 $(\Sigma+1)\times(\Sigma+1)$ 행렬
$F \leftarrow O$
For each $x \in W$
  For each $y \in S_W(x)$
    $K \leftarrow |Align(x,y)|$
    For each $A \in Align(x,y)$
      For each $\langle \alpha,\beta \rangle \in A$
        $F_{\alpha,\beta} \leftarrow F_{\alpha,\beta} + 1/K$
        $F_{\beta,\alpha} \leftarrow F_{\beta,\alpha} + 1/K$

그림 5. 음소간 대응 빈도를 구하기 위한 알고리즘

4. 음소 간 유사도 계산

앞서 2장에서 언급하였듯이 음소 간 유사도 점수 계산을 위한 확률 모델을 기본적으로 생물정보학에서 주로 사용되는 PAM(Point Accepted Mutation) 행렬[8]을 구성하는 방법을 기반으로 하되 데이터 특성에 맞도록 다소 변형하여 적용한다.

먼저 각 음소 α가 등장하는 비율 p_α가 필요하다. p_α는 주어진 문자열 집합 W 내의 모든 단어 $x \in W$에 대하여 α가 등장하는 비율이다. 앞선 예에서 W={“바카라”, “박하롸”, “카지노”, “카ㅈI노”, “카쥐no”}일 때, 총 음소의 개수는 31개이며 이 중 “ㅂ”은 2번 등장하고, “ㅏ”는 8회 등장한다. 따라서 $p_{ㅂ}$=2/31, $p_{ㅏ}$=8/31이다. 그렇다면 단어 내에서 어떤 음소 α가 변형될 확률 m_α은 다음과 같이 표현된다.

$$m_\alpha = \frac{\sum_{\beta(\neq\alpha)} F_{\alpha,\beta}}{p_\alpha \sum_{\beta} \sum_{\gamma(\neq\beta)} F_{\beta,\gamma}} \cdot \frac{1}{L} \qquad (7)$$

분모의 $\Sigma_\beta \Sigma_{\gamma(\neq\beta)} F_{\beta,\gamma}$는 한 음소가 변형된 빈도를 의미하며 이 값에 음소 α의 출현 비율 p_α를 곱해줌으로써 출현 비율에 따라 가중치를 부여할 수 있다. 예를 들어 전체 음소 변형 횟수가 100회가 일어났는데, 단어 중 음소 α가 차지하는 비율이 0.03이라면, 음소 α가 변형된 횟수는 3회라고 예측할 수 있다. 분자의 $\Sigma_{\beta(\neq\alpha)} F_{\alpha,\beta}$는 실제로 음소 α가 변형된 횟수이다. 만약 예측되는 음소 변형 빈도보다 실제 변형이 일어난 횟수가 더 크다면 이 음소는 변형이 매우 쉽게 일어나는 음소라는 사실을 알 수 있다. 즉 이들의 비는 음소 α의 상대적인 변형 확률을 의미한다. 한편 L는 변형 확률을 계산하기 위한 단위 구간을 의미한다. 원래 PAM 행렬을 구성하는 방법[8]에서는 $L=100$이 사용되었는데 이는 길이 100당 변형 확률을 의미하는 것이다. 본 논문에서는 한 단어당 변형 확률을 구하고자 하므로, 문자열 집합 W내의 모든 문자열들의 길이의 평균값을 이용하였다. 한편 m_α가 확률을 의미함에도 불구하고 L과 주어진 데이터의 특성에 따라 1이 넘는 경우가 발생하기도 하는데, 이후 최종 유사도 점수 변환 과정에서 처리하도록 한다.

유사도 점수를 구하기에 앞서 두 음소 $\alpha,\beta \in \Sigma$ 가 서로 대응될 확률 $M_{\alpha,\beta}$를 구하고자 한다. 같은 음소가 대응될 확률은 음소 α가 변형되지 않을 확률과 같으므로 아래와 같이 정의된다.

$$M_{\alpha,\alpha} = 1 - m_\alpha \qquad (8)$$

서로 다른 음소 α, β에 대해서는 대응될 확률 $M_{\alpha,\beta}$가 다음과 같이 정의된다.

$$M_{\alpha,\beta} = \frac{F_{\alpha,\beta}}{\sum_{\gamma(\neq\alpha)} F_{\alpha,\gamma}} m_\alpha \qquad (9)$$

최종적으로 $M_{\alpha,\beta}$을 이용하여 α와 β간의 유사도 점수 $Score$는 다음과 같이 구할 수 있다.

$$Score_{\alpha,\beta} = \left\lfloor 10\log\frac{M_{\alpha,\beta}}{p_\beta} \right\rfloor \qquad (10)$$

상수 10을 곱하고 바닥 함수(Floor function)을 사용한 것은 일반적으로 문자열 정렬(Alignment)시에 점수 값의 데이터 형을 정수로 사용하기 때문이다. 상수 10은 시스템에서 요구하는 정밀도에 따라 적당한 값으로 대체될 수 있다.

수식 (10)에서 $\alpha \neq \beta$의 경우 $M_{\alpha,\beta}$를 $Score_{\alpha,\beta}$에 대입하면 $F_{\alpha,\beta}/(L \cdot p_\alpha \cdot p_\beta \cdot \Sigma_\gamma \Sigma_{\delta(\neq\gamma)} F_{\gamma,\delta})$꼴이 되므로

모든 α,β에 대하여 $Score_{\alpha,\beta} = Score_{\beta,\alpha}$임을 쉽게 확인할 수 있다.

앞서 언급했듯이 변형 단어들은 유전자 염기 서열과는 달리 문자열의 길이가 충분히 길지 않고 구성하는 문자(음소)의 종류가 다양하기 때문에 m_α를 구하는 과정에서 설정한 L의 값과 데이터의 특성에 따라 로그를 취하기 전의 값 $M_{\alpha,\beta}$이 양의 실수가 아닌 경우가 발생한다. $M_{\alpha,\beta}$이 양의 실수가 아닌 경우에는 두 음소 간의 유사도 점수가 극히 낮다고 판단해도 무방하나 실용적인 측면에서 음의 무한대의 값을 부여하는 것은 예외처리에 대한 부담을 가중시키므로, 이 경우 $M_{\alpha,\beta} > 0$인 다른 α,β들이 가지는 $Score_{\alpha,\beta}$값들 중 최솟값, 즉 $\min_{\alpha,\beta;(M_{\alpha,\beta}>0)} Score_{\alpha,\beta}$ 을 사용하도록 한다.

한편 갭(Gap)문자의 경우에는 원래 문자열에는 존재하지 않지만 정렬 과정에서 발생하는 특수한 경우이므로 p_α, p_β에 해당하는 부분을 구할 수가 없다. 또한 주어진 단어 집합을 구성하는 문자가 아닌 알려지지 않은 문자의 경우에도 앞에서 주어진 식에 의한 학습이 불가능하다. 따라서 갭(Gap) 점수와 알려지지 않은 문자와의 불일치(Mismatch) 점수는 유사도 점수들의 평균값으로 한다.

$$Score_{\alpha,\#} = Score_{\#,\alpha} = \frac{\sum_{\beta\in\Sigma}\sum_{\gamma\in\Sigma} Score_{\beta,\gamma}}{|\Sigma|\times|\Sigma|} \qquad (11)$$

단, "#"은 갭(Gap) 또는 알려지지 않은 문자이다.

Ⅳ. 실험 및 결과

1. 실험 데이터

실험을 위한 데이터는 다음과 같은 방법을 이용하여 직접 수집하였다. 참여자는 20대 남녀 10명이었으며, 각각의 사용자가 변형 욕설을 입력하면 이를 시스템이 무작위로 음소를 변형하고, 그 결과 단어에 대하여 다른 참여자들이 욕설 여부를 평가한 결과를 취합하는 방식으로 진행되었다. 평가 결과 참여자 중 7명 이상이 욕설이라고 판정을 내린 단어가 총 11,871 단어였으며, 특성은 [표 4]와 같다. 수집 방법의 특성 상 하나의 단어로부터 여러 개의 변형 단어가 파생되기 때문에 본 논문에서 제시한 가정에 부합되는 데이터임을 확인할 수 있다. 또한 "이발", "씨알", "지발" 등과 같이 욕설에서 변형되는 과정에서 원래의 의미를 상실한 단어 1,881개 역시 성능 평가에 활용하였다.

표 4. 수집된 데이터의 특성. 일부 클래스에 변형 형태가 집중적으로 출현하고 있음을 알 수 있다.

항목	값
단어 개수	11,871
음소 가짓수	131
클래스 개수	216
클래스 당 최대 단어 개수	3,362
클래스 당 평균 단어 개수	54.96
단어 평균 길이 (음소 단위)	7.77

2. 실험 방법

수집된 욕설 단어 11,871개를 이용하여 10-fold cross validation 기법에 따라 데이터를 10등분하여 9부분은 학습에, 나머지 1부분을 평가에 이용하는 실험을 교차적으로 시행하였다.

표 5. 유사도 점수 계산 결과의 일부. 유사한 음소 간에는 유사하지 않은 음소에 비해 더 높은 유사도 점수가 부여된다.

음소 1	음소 2	유사도 점수
ㄱ	ㄱ	24
	ㄱ	5
	ㄱ	2
	ㄴ	-30
ㅅ	ㅅ	36
	ㅅ	42
	ㅅ	32
	ㅏ	-79
ㅣ	ㅣ	23
	ㅣ	-5
	ㅣ	1
	ㅟ	-7
	ㅘ	-23
	ㅏ	-28
	ㅅ	-131

제안 기법을 통하여 유사도 점수를 계산한 결과가 [표 5]에 나타나있다. 지면상의 이유로 유사도 점수의 특성을 보여주는 일부의 예만 수록하였다. [표 5]에 나

타나듯이 유사한 형태의 음소 간에는 부여되는 유사도 점수가 서로 상이한 형태 간에 비하여 큰 것을 알 수 있다. 특히 모음의 경우 보다 명확하게 드러나는데, "ㅣ"의 경우에는 "I(대문자 I)"나 "|(Vertical Bar)" 등으로는 쉽게 변형되므로 두 음소가 대응될 때 비교적 높은 점수를 갖지만 "ㅘ"나 "ㅏ"로는 변형이 잘 일어나지 않아 해당 음소 쌍에 대하여는 상대적으로 낮은 유사도 점수를 가지게 된다. 특히 모음 "ㅣ"로부터 자음 "ㅅ"으로는 변형이 일어나는 빈도가 극히 드물기 때문에 매우 낮은 유사도 점수를 가진다.

3. 성능 평가 및 검토

주어진 질의 단어 q에 대하여 욕설 단어 집합 A내의 각 단어와 유사도 점수 행렬 S를 사용하여 전역 정렬을 수행했을 때 얻을 수 있는 가장 높은 점수를 $W(q;A,S)$라 정의 한다.

$$W(q;A,S) = \max_{a \in A} AlignScore(q,a;S) \quad (12)$$

$AlignScore(q,a;S)$는 수식 (1)에 따라 계산하며, 따라서 $W(q;A,S)$는 집합 A내의 단어 중 q와 가장 유사한 단어와의 정렬 점수를 의미한다. 이 때 어떤 집합 X 내에서 $W(q;A,S)$ 값이 주어진 매개변수 θ보다 큰 단어들의 집합을 $R(X;A,\theta,S)$라 하자.

$$R(X;A,\theta,S) = \{x \in X : W(x;A,S) > \theta\} \quad (13)$$

성능 평가 척도로는 민감도(Sensitivity)와 특이도(Specificity)를 이용한다. 욕설 단어(TP+FN)를 욕설과 유사하다고 판정(TP)하는 비율이 민감도이며, 욕설이 아닌 단어(TN+FP)를 욕설과 유사하지 않다(TN)고 판정하는 비율이 특이도이다. 실험 데이터 중 학습 데이터의 1/100에 해당하는 단어들을 무작위로 추출하여 집합 A라 하고, 평가 데이터 단어 집합이 B, 비욕설 단어 집합을 C라 하면, 민감도와 특이도는 다음과 같이 표현된다.

$$Sensitivity = \frac{TP}{TP+FN} = \frac{|R(B;A,\theta,S)|}{|B|} \quad (14)$$

$$Specificity = \frac{TN}{TN+FP} = \frac{|C| - |R(C;A,\theta,S)|}{|C|} \quad (15)$$

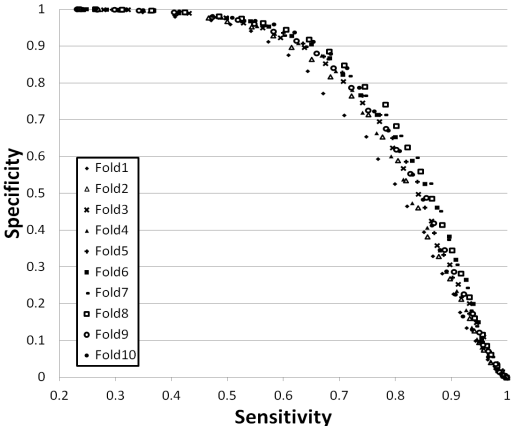

그림 6. 실험 결과를 나타낸 그래프. 민감도와 특이도가 교차하는 지점은 각각 74.7%, 74.6%이다.

매개변수 θ를 조절하면서 실험한 결과는 [그림 6]과 같다. 그래프 상으로 민감도와 특이도의 교차지점은 각각 74.7%, 74.6%이다. 구체적인 수치(Fold 8)는 [표 6]에 나타나있다. θ=80에서 77.3%의 민감도와 69.6%의 특이도를 가짐을 확인할 수 있다. 단, 여기서의 θ는 모든 데이터에 대하여 일정한 것이 아니라 시스템에서 사용하는 데이터에 의존적이므로, 시스템의 목적에 맞게 실험적으로 적합한 값을 계산해야한다. 기존 방법과의 성능 비교는 [표 7]에 기술되어 있다. "기본 점수" 방법은 대응하는 두 음소가 일치하는 경우에는 +1점, 불일치하거나 갭(Gap)과 대응되는 경우에는 -1점을 부여하는 점수 부여 방식이고, "수동 설정"은 [6]에서 사용한 점수 부여 방식에 따라 점수를 부여하여 실험을 수행한 결과이다. 제안 기법의 성능은 실험 시 가장 우수한 결과(Fold 8)와 가장 저조한 결과(Fold 10)를 기술하였다. 기본 점수 방법이 84.1%의 민감도에서 불과 29.6%의 특이도를 보이는 반면 제안 기법은 해당 민감도 수치 구간에서 39.4%~42.6%의 특이도를 가지는 것을 확인하였다. 이는 수동으로 직접 음소 간 유사도를 설정한 결과(33.2%)보다 6.2%~9.4% 향상된 결과이다.

적당한 수준의 특이도를 유지하여 실험한 경우 기본 점수 방법은 74.9%의 특이도에서 56.7%의 민감도를 보인 반면, 제안 기법은 그보다 다소 높은 특이도(77.2%~80.4%)에서 67.1%~70.8%의 민감도를 가짐으로서

기본 점수 방법에 비하여 10.4%~14.1%, 수동 설정 기법(59.0%)에 비하여는 8.1%~11.8%의 민감도 성능 향상이 있음을 확인하였다.

표 6. 매개변수 θ에 따른 성능 평가 결과(Fold 8). θ=80일 때 73.7%, 특이도 78.7%의 성능을 보인다.

θ	Sensitivity	Specificity
20	0.972	0.062
40	0.993	0.201
60	0.865	0.426
80	0.773	0.696
100	0.638	0.896
120	0.498	0.978
140	0.368	0.997

표 7. 기존 방법과의 비교. 비슷한 민감도 또는 특이도를 기준으로 비교를 했을 때 제안기법이 우수하다.

기법	Sensitivity	Specificity	θ
기본 점수	0.841	0.296	-1
	0.567	0.749	1
수동 설정	0.856	0.332	0
	0.590	0.770	3
제안 기법 (Fold 8)	0.865	0.426	60
	0.708	0.804	110
제안 기법 (Fold 10)	0.851	0.394	90
	0.671	0.772	110

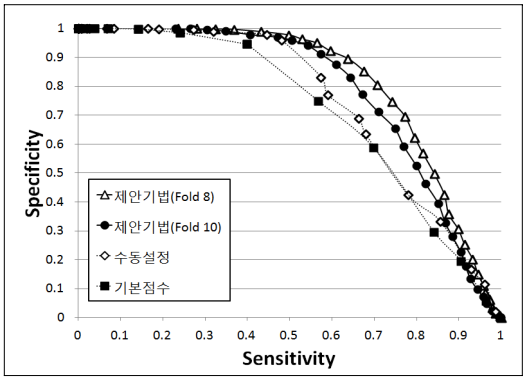

그림 7. 기존 방법과의 비교. 기본 점수는 Match=+1, Mismatch=Gap=-1으로 부여하고, 수동 설정은 [6]의 점수를 이용하였다. 제안 기법이 전체적으로 우수하다.

[그림 7]은 기존 방법과의 성능 비교 그래프이다. 가로축은 민감도이며, 세로축은 특이도이다. 실선으로 표시된 것이 제안 기법의 성능이며, 기본 점수 부여 기법이나 수동으로 음소간의 유사도를 설정하는 기법에 비하여 그래프의 곡선이 우측 상단에 위치해 있어 상대적으로 우수한 성능을 보임을 확인할 수 있다.

V. 결 론

본 논문에서는 주어진 단어 집합 내에서 자동적으로 유사 단어 및 음소 쌍을 찾아 유사도 점수를 계산하는 기법을 제안하였으며, 결론은 다음과 같다.

1. 생물정보학에서 사용되는 PAM 행렬을 응용하여 한글 문자열 상의 음소 간 유사도 점수를 계산하는 방법을 제안하였다.
2. 제안 기법은 자동적으로 음소 간의 유사도 점수를 계산해주므로 기존의 수동 점수 설정에 따른 과다한 작업량, 점수의 일관성에 관한 한계점을 극복할 수 있다.
3. 특이도 77.2~80.4% 수준에서 기본 점수 기법에 비해서는 10.4~14.1%, 수동 설정 기법[6]에 비해서는 8.1~11.8%의 민감도 향상을 실험적으로 확인하였다.

비록 제안 기법이 기존의 기법에 비하여 우수한 성능을 보이기는 했지만 PAM 행렬에서 근본적으로 가정하고 있는 두 문자 α, β의 대응확률이 각 문자의 출현 확률의 곱 $p_\alpha p_\beta$와 같다는 것을 그대로 사용하였다는 한계점이 있다. 이로 인하여 갭(Gap)문자가 학습에 이용되었음에도 불구하고 갭(Gap) 점수를 부여하는 과정에서는 단순히 다른 음소 간 유사도 점수의 평균 점수만을 이용할 수밖에 없었다. 따라서 문자의 출현 빈도와 대응 빈도 간의 관계에 대한 실험적 연구가 필요할 것으로 생각되며, 이에 기반을 둔 효과적인 갭(Gap)처리 모델을 위한 개선이 추가적으로 필요하다. 덧붙여, 유사 단어 쌍을 검출하는 단계에 있어서 군집화가 되는 유형이나 성능에 따라 유사도 계산을 위한 대상이 달라지므로 이에 대한 보다 심층적인 실험을 통한 관계 입증이 필요할 것으로 사료된다.

참 고 문 헌

[1] Gonzalo Navarro, "A Guided Tour to Approximate String Matching," ACM Computing Surveys, Vol.33, No.1, pp.31-88, 2001.

[2] 정보통신부, "*의미부류별 핵심어매칭기술을 이용한 한국어 및 영어 컨텐츠 유해등급 자동판정 시스템 개발*", 2003.

[3] 한국게임산업진흥원, "*게임언어 건전화 지침서 연구*", 2008.

[4] 노강호, 박근수, 조환규, 장소원, "음소의 분류 체계를 이용한 한글 편집 거리 알고리즘", 정보과학회논문지:시스템 및 이론, 제37권, 제6호, pp.319-367, 2010.

[5] 윤태진, 조환규, "반 전역 정렬을 이용한 온라인 게임 변형 욕설 필터링 시스템", 한국콘텐츠학회논문지, 제9권, 제12호, pp.113-120, 2009.

[6] 윤태진, 정우근, 조환규, "제한된 한글 입력환경을 위한 음소기반 근사 문자열 검색 시스템", 정보과학회논문지:소프트웨어 및 응용, 제37권, 제10호, pp.788-801, 2010.

[7] 안희국, 한욱표, 신승호, 양동일, 노희영, "스팸메일 필터링을 위한 한글 변칙어 인식 방법", 한국항행학회논문지, 제15권, 제2호, pp.287-297, 2011.

[8] J. Setubal and J, Meidanis, *"Introduction to Computational Molecular Biology,"* PWS Publishing Company, 1997.

[9] 송영길, 김학수, "다양한 스마트폰 키패드 환경에서 유사 단어 검색을 위한 수정된 편집 거리 계산 방법", 한국콘텐츠학회논문지, 제11권, 제12호, pp.12-18, 2011.

저 자 소 개

김 성 환(Sung-Hwan Kim) 준회원

- 2011년 : 부산대학교 정보컴퓨터공학부(공학사)
- 2011년 3월 ~ 현재 : 부산대학교 컴퓨터공학과 석사과정

<관심분야> : 한글언어처리, 정보검색, HCI

조 환 규(Hwan-Gue Cho) 정회원

- 1984년 : 서울대학교 계산통계학과(이학사)
- 1986년 : KAIST 대학원 전산학과(공학석사)
- 1990년 : KAIST 대학원 전산학과(공학박사)
- 1990년 3월 ~ 현재 : 부산대학교 컴퓨터공학과 교수

<관심분야> : 계산이론, 생물정보학