키-시퀀스 예측을 통한 가변형 소프트 키보드: 안드로이드 플랫폼 적용 사례 연구

홍택규° 고봉석 김기응
한국과학기술원 전산학과
taekgyu@kaist.ac.kr, bsgoh@ai.kaist.ac.kr, kekim@cs.kaist.ac.kr

Adaptive Soft Keyboard via Key-Sequence Prediction: A Development Case Study on Android Platform

Teakgyu Hong° Bongseok Goh Kee-Eung Kim
Department of Computer Science, KAIST

요 약

소프트 키보드(soft keyboard)는 소프트웨어로 제어가 가능하다는 점과 물리적인 키보드를 사용할 때보다 넓은 화면을 사용할 수 있다는 장점으로 인해 최근 대부분의 스마트폰에서 이용되고 있다. 이러한 소프트 키보드의 장점에도 불구하고 스마트폰 자체의 작은 화면 크기로 인해 사용자들은 많은 오타를 낸다. 이 문제를 해결하기 위해 Microsoft Research의 Gunawardana et al. [1]은 키-시퀀스(key-sequence) 예측을 통한 가변형 소프트 키보드를 제안하였다. 이는 사용자가 누른 키보드 위치와 현재까지 입력한 문자들을 바탕으로 다음에 입력할 문자들에 대한 확률 모델을 만들고, 이를 바탕으로 키보드를 구성하는 각 키(key)의 감지 영역을 변환하는 방법이다. 본 논문은 안드로이드 플랫폼(android platform)에서 키-시퀀스 예측을 통한 가변형 소프트 키보드를 적용한 사례와 한국어 키보드로도 적용한 사례를 보인다.

1. 서 론

소프트 키보드(soft keyboard)는 다양한 전자기기의 화면 위에 올라오는 키보드로써 온-스크린 키보드(on-screen keyboard) 또는 소프트웨어 키보드(software keyboard)라고도 불린다. 소프트 키보드는 소프트웨어로 제어할 수 있기 때문에 여러 나라의 언어를 지원할 수 있고, 이를 사용했을 때 물리적인 키보드가 들어갈 자리까지 화면으로 이용할 수 있기 때문에 넓은 화면을 사용할 수 있다는 장점이 있다. 그러나 스마트폰 자체의 작은 화면 크기로 인해 많은 사용자들이 오타를 빈번하게 낸다는 문제점이 있다.

Microsoft Research의 Gunawardana et al. [1]은 이러한 문제점을 해결하기 위해 키-시퀀스 예측을 통한 가변형 소프트 키보드를 제안하였다. 키-시퀀스 예측을 통한 가변형 소프트 키보드는 각 키의 감지 영역이 사용자의 키보드 터치 위치와 현재까지 입력된 문자들을 바탕으로 변하는 키보드이다. 가변형 소프트 키보드를 사용하는 경우 기존의 소프트 키보드보다 평균적으로 오타율이 약 11% 감소한 것으로 나타났다.

소프트 키보드의 구현을 위한 플랫폼(platform)으로는 크게 iOS와 안드로이드가 있다. iOS는 프로그램을 개발하고 그것을 사용하기 위해 개발자 등록 과정을 거쳐야만 한다. 반면 안드로이드는 그러한 과정 없이 누구나 프로그램을 개발 할 수 있고 그것을 사용할 수도 있다. 본 논문은 개발의 용이성을 위하여 키-시퀀스 예측을 통한 가변형 소프트 키보드를 안드로이드 플랫폼에 적용한다. 또한 Gunawardana et al. [1]은 영어 키보드에 대해서만 다루었지만, 본 논문에서는 영어 및 한국어 키보드로 적용한 사례를 다룬다. 그리고 그 과정 속에서 발생한 문제점과 해결방안을 제시한다.

본 논문의 구성은 다음과 같다. 2장에서는 가변형 키보드의 키 감지 영역 결정 방법을 설명하고 3장에서는 이 방법을 안드로이드 플랫폼에 적용한 사례를 설명한다. 마지막으로 4장에서 결론을 내리고 향후 연구방향을 제시한다.

2. 가변형 키보드의 키 감지 영역 결정 방법

키-시퀀스 예측을 통한 가변형 소프트 키보드를 구현할 때, 각 키의 감지 영역은 언어 모델(language model)과 터치 모델(touch model)에 의해서 결정된다[2]. 언어 모델은 현재까지 입력된 문자들을 바탕으로 다음에 입력될 문자들의 확률 모델이고, 가장 널리 쓰이고 있는 언어 모델로는 N-gram 언어 모델이 있다[3]. N-gram 언어 모델은 입력될 문자들의 확률을 최근 입력된 $N-1$개의 문자들을 기반으로 구한다. 터치 모델은 어떤 문자를 터치하려고 할 때 각 픽셀이 터치될 확률 모델이고, 대표적인 터치 모델로는 픽셀의 좌표값 x와 y를 변수로 가지는 이변수 정규 분포(bivariate normal distribution)가 있다[2].

위의 두 모델을 사용하여 현재까지 입력된 문자들과 사용자가 현재 터치한 픽셀의 좌표를 바탕으로 다음에 입력될 문자들에 대한 확률을 구할 수 있다. 사용자가 입력할 수 있는 키 k의 집합을 K라 하자. 예를 들어 영어의 경우 $K = \{a,...,z\}$ 이다. 현재 터치한 픽셀의 좌표가 $l(=(x,y))$이고 최근에 입력된 $N-1$개의 문자들을 $h(=k_{i-N+1},\cdots,k_{i-1})$라고 하자. 키 k가 입력될 확률값 $p(k)$는 언어 모델의 확률값 $p_L(k|h)$와 터치 모델의 확률값 $p_T(l|k)$의 곱에 비례하는 형태로 나타낸다[1].

$$p(k) \propto p_L(k|h)p_T(l|k)$$

이 확률값을 이용하여 키보드에서 알파벳 키 k에 대한 감지 영역 $\Gamma_{key}(h)$는 다음과 같이 정의한다.

$$\Gamma_k(h) = \{l \mid p_L(k|h)p_T(l|k) > p_L(k'|h)p_T(l|k'),\ \forall k' \neq k\}$$

이는 입력된 문자들 h가 주어졌을 때, 한 알파벳 키 k에 속할 확률이 가장 높은 픽셀들의 집합을 의미한다. 위의 과정을 통해 모든 키에 대해 감지 영역을 구하면 키-시퀀스 예측을 통한 가변형 소프트 키보드를 구현할 수 있다.

3. 가변형 키보드 적용 사례

안드로이드 플랫폼에 가변형 키보드를 적용한 사례에 대한 설명은 다음과 같이 진행된다. 우선 언어 모델과 터치 모델의 구현과정을 설명하고 영어 및 한국어 키보드로 적용한 사례를 설명한다. 그 후 적용하는 과정 중 겪었던 문제점과 해결방안을 제시한다. 개발 대상 기기는 삼성 갤럭시(Galaxy) S2이고 해당 기기에 기본으로 설치되어있는 키보드인 삼성 키패드(Samsung keypad)의 이미지를 이용하여 가변형 키보드를 구현하였다.

3.1. 언어 모델 구현

기존 가변형 키보드에서는 보간 Kneser-Ney 스무딩(interpolated Kneser-Ney smoothing)을 이용하는 8-grams 언어 모델을 사용하였다[1, 4]. 하지만 본 논문에서는 모델의 간결성을 위해 라플라스 스무딩(Laplace smoothing)을 이용하고, 실험환경의 제약으로 인하여 3-grams 언어 모델을 사용한다.

언어 모델의 확률값 $p_L(k|h)$는 학습 데이터(training data)에 최대우도추정(maximum likelihood estimation)을 통해 다음과 같이 학습된다.

$$p_L(k|h) = p(k_i|k_{i-2},k_{i-1}) = \frac{c(k_{i-2},k_{i-1},k_i)+1}{\sum_{k_i'} c(k_{i-2},k_{i-1},k_i') + |K|}$$

여기서 $c(\cdot)$는 트레이닝 데이터에서 문자열의 빈도수이다.

3.2. 터치 모델 구현

기존 가변형 키보드에서는 최대사후확률추정(maximum a posterior estimation)으로 학습시킨 full-covariance 이변수 정규 분포로 터치 모델을 사용하였다[1]. 하지만 본 논문에서는 모델의 간결성을 위해 각 키의 중심점을 평균으로 하는 비상관 이변수 표준 정규 분포(uncorrelated bivariate standard normal distribution)를 사용한다. 터치한 픽셀의 좌표 l과 키 k에 대한 중심점의 좌표가 (k_x,k_y)라고 할 때, 확률밀도함수(probability density function)는 다음과 같다.

$$f(x,y) = \frac{1}{2\pi} e^{-\frac{1}{2}\{(x-k_x)^2+(y-k_y)^2\}}$$

원래 확률밀도함수의 값은 확률이 아니지만 픽셀의 크기가 모두 동일하고 그 크기가 작기 때문에 이 값을 터치 모델의 확률값으로 근사화(approximation) 할 수 있다.

$$p_T(l|k) \propto f(x,y)$$

3.3. 가변형 영어 키보드 적용 사례

가변형 영어 키보드를 구현할 때 알파벳 대문자는 소문자로 변환하여 학습시킨다. 주어진 학습 데이터[1]로부터 미리 두 자리 및 세 자리의 가능한 모든 문자열의 집합 {aa, ab, ..., zy, zz, aaa, aab, ..., zzy, zzz}에 대한 언어 모델의 확률값을 스무딩을 이용해 미리 학습시킨다. 여기서 두 자리도 같이 고려하는 이유는 현재까지 입력된 문자열의 길이가 1인 경우를 다루기 위함이다. 현재까지 입력된 단어가 없는 한 자리 문자열의 경우에는 모든 문자들에게 동일한 확률값을 주어 언어 모델의 확률값을 정한다. 이후 사용자가 입력한 문자들과 키보드를 구성하는 모든 픽셀들에 대해서 $\Gamma_k(h)$를 구함으로써 각 키의 감지 영역을 구한다.

그림 1은 가변형 영어 키보드로 카카오톡(kakao talk) 대화창에서 메시지를 입력하는 화면이다.

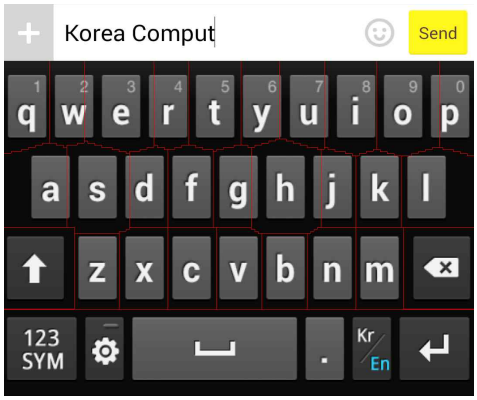

그림 1. 가변형 영어 키보드

3.4. 가변형 한국어 키보드 적용 사례

한국어는 '과학'과 같이 모아쓰기 형태로도 쓸 수 있고, 'ㄱㅘㅎㅏㄱ'과 같이 풀어쓰기 형태로도 쓸 수 있다[5]. 현대 한국어는 모아쓰기 형태로 쓰기 때문에 학습 데이터 또한 모아쓰기 형태로 주어져 있다. 언어 모

1) http://www.yorku.ca/mack/PhraseSets.zip

델이 키의 문자 단위로 학습이 되어야하기 때문에 학습 데이터를 풀어쓰기 형태로 변환하여야 한다. 또한 ‘ㅘ’, ‘ㅙ’, ‘ㅢ’ 등의 이중모음과 ‘ㄳ’, ‘ㄵ’, ‘ㅀ’ 등의 이중자음은 쿼티(qwerty) 키보드에서 한 번에 입력할 수 없기 때문에 각각의 구성요소로 분리한다. 예를 들면 ‘뭐가 없나요’를 풀어쓰기 형태로 나타내면 ‘ㅁㅝㄱㅏ ㅇㅓㅄㄴㅏㅇㅛ’이고, 여기서 이중 자음과 이중 모음을 각각의 구성요소로 분리하면 ‘ㅁㅜㅓㄱㅏ ㅇㅓㅂㅅㄴㅏㅇㅗ’이다. 이러한 풀어쓰기 형태로 변환된 학습 데이터를 통해 언어 모델을 학습시킨다.

‘ㅃ’, ‘ㅉ’, ‘ㄸ’, ‘ㄲ’, ‘ㅒ’, ‘ㅖ’는 쉬프트(shift) 키를 눌러야만 입력할 수 있다. 그렇기 때문에 쉬프트 키가 눌리지 않은 경우와 눌린 경우에 대해 각각의 가변형 키보드를 구현한다. 즉, 한국어에 대해서는 두 개의 키보드가 존재하고 쉬프트 키에 의해 키보드가 변환된다.

안드로이드에서 제공하는 키보드 인터페이스(interface)에서는 한국어를 입력할 때 모아쓰기 형태로 작성해주지 않는다. 따라서 모아쓰기를 해주는 한글 오토마타가 필요하다. 한글 오토마타는 Choe [6]를 참고하여 구현한다.

그림 2는 가변형 한국어 키보드로 카카오톡 대화창에서 메시지를 입력하는 화면이다.

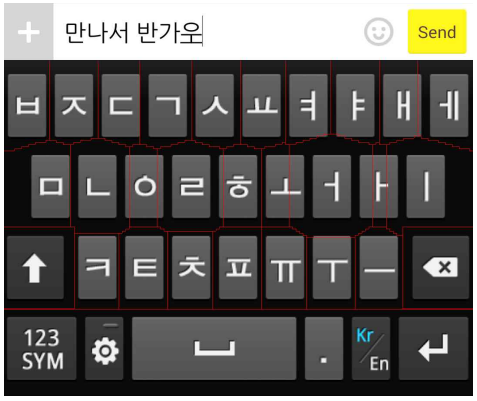

그림 2. 가변형 한국어 키보드

3.4. 문제점과 해결방안

안드로이드 플랫폼에 가변형 키보드를 적용할 때 주된 문제점은 키 감지 영역 $\Gamma_k(h)$가 바뀌는데 걸리는 시간이었다. 각 키의 감지 영역을 구할 때 키보드 화면을 구성하는 모든 픽셀을 고려하기 때문에 감지 영역이 바뀌는데 시간이 오래 걸렸다. 이를 극복하기 위해 픽셀들을 그룹화하고 그 그룹의 대표 픽셀이 속할 키를 해당 그룹에 속하는 픽셀들에게도 할당한다. 본 논문에서는 구현 시 픽셀 그룹을 3×3의 픽셀들로 정한다. 이때 대표 픽셀은 정 가운데의 픽셀로 정한다. 이렇게 했을 때 계산량이 9배 감소하므로 키의 감지 영역이 바뀌는 시간이 기존보다 빨라진다.

4. 결론 및 향후 연구

본 논문에서는 키-시퀀스 예측을 통한 가변형 소프트 키보드를 안드로이드 플랫폼에 적용한 사례와 그 과정 속에서 발생한 문제점과 해결방안을 제시하였다. 적용 결과 영어의 경우 키 영역 변환이 올바르게 이루어지나, 한국어의 경우는 그러하지 못하였다. 이는 한국어의 언어 모델을 학습시킬 때 학습 데이터의 문자들을 초성, 중성, 종성으로 분리하고 그 후 이중 자음과 이중 모음은 각각의 구성요소로 분리하기 때문이다. 3-grams 언어 모델을 통해 다음에 입력될 한글을 예측할 때 문자열 길이가 올바른 예측을 하기에는 짧다는 한계가 있었다.

본 논문에서 구현한 키보드의 성능을 높이기 위한 방안으로는 한국어의 언어 모델에 대해서만 N-grams의 계수를 높이는 방법이 있을 수 있다. 그리고 지금의 적용 사례에서는 학습 방식이 오프라인(off-line) 학습이지만 이를 온라인(on-line) 학습으로 구현한다면 지금보다 더 정확한 언어 모델을 만들 수 있을 것이다.

영어와 한국어에 대해서 동작하는 키-시퀀스 예측을 통한 가변형 소프트 키보드의 apk파일은 https://www.dropbox.com/s/yr9nbsfyi8kav2b/MyKeyboard.apk에서 다운받을 수 있다.

참 고 문 헌

[1] Gunawardana, Asela, Tim Paek, and Christopher Meek. “Usability guided key-target resizing for soft keyboards.“ Proceedings of the 15th international conference on Intelligent user interfaces. ACM, 2010.

[2] Goodman, Joshua, et al. “Language modeling for soft keyboards.“ Proceedings of the 7th international conference on Intelligent user interfaces. ACM, 2002.

[3] Goodman, Joshua T. “A bit of progress in language modeling.“ Computer Speech & Language 15.4 (2001): 403-434.

[4] Chen, Stanley F., and Joshua Goodman. “An empirical study of smoothing techniques for language modeling.“ Proceedings of the 34th annual meeting on Association for Computational Linguistics. Association for Computational Linguistics, 1996.

[5] 박병천 외, 「한글 글꼴 용어사전」, 세종대왕기념사업회, 2011

[6] Choe, K. M. A Study on the Hangul Formation, M. S. Diss. Thesis, Dept. of Comp. Sc., KAIS, 1978.