석사학위논문

시퀀스-투-시퀀스 모델을 적용한 웨어러블 입력 인터페이스 오류 교정 모델

제주대학교대학원

컴퓨터공학과

한 승 의

2021년 12월

시퀀스-투-시퀀스 모델을 적용한 웨어러블 입력 인터페이스 오류 교정 모델

지도교수 곽 호 영

한 승 의

이 논문을 컴퓨터공학 석사학위 논문으로 제출함

2021 년 12 월

한승의의 공학 석사학위 논문을 인준함

심사위원장 이 상 준 ㊞

위 원 김 수 준 ㊞

위 원 곽 호 영 ㊞

제주대학교 대학원

2021 년 12 월

Sequence-to-sequence model based error correction model for wearable input interfaces

Seung-Eui Han
(Supervised by professor Ho-Young Kwak)

A thesis submitted in partial fulfillment of the requirement for the degree of Master of Science in Computer Engineering

2021 . 12 .

This thesis has been examined and approved.

Thesis director, Seung Joon Lee
Thesis director, Sookyun Kim
Thesis director, Ho-Young Kwak
December 2021

Department of Computer Engineering
GRADUATE SCHOOL
JEJU NATIONAL UNIVERSITY

목 차

그 림 목 차

표 목 차

국 문 초 록

시퀀스-투-시퀀스 모델을 적용한 웨어러블 입력 인터페이스 오류 교정 모델

컴퓨터공학과 한 승 의
지 도 교 수 곽 호 영

최근 머신러닝 기술에 대한 연구 범위가 공학·기술의 연구 분야를 넘어 일상의 다양한 서비스 분야까지 그 범위가 넓어지고 있다. 이런 연구의 흐름은 인간의 언어를 컴퓨터가 처리할 수 있도록 하는 기술인 자연어 처리(Natural Language Processing)에서도 찾아볼 수 있다. 이러한 자연어 처리의 적용 분야 중 하나인 철자 오류 교정 알고리즘은 다양한 검색 엔진이나 문서 작성 프로그램에 도입되어 사용자의 검색어 철자 오류 교정, 문서 내 문법 오류 교정 등의 기능을 지원하고 있다.

본 논문에서는 웨어러블 입력 인터페이스를 사용하여 문장을 입력할 때 오류에 의해 잘못 입력된 철자를 교정하기 위해 시퀀스-투-시퀀스(Sequence-to-Sequence)를 적용한 입력 오류 교정 알고리즘을 제안하였다. 이를 구현하기 위해 실제 사용자가 웨어러블 입력 인터페이스를 사용하여 입력한 문자열과 기기에서 측정된 데이터를 수집하였다. 수집한 데이터의 분석을 통해 입력 문자열의 철자 오류가 발생하는 형태가 4가지의 오류 패턴을 보인다는 것을 확인할 수 있었다. 이를 통해 사용자가 문자를 입력할 때 측정되는 기기 측정 데이터는 입력 오류를 교정하는 데 중요한 요소임을 확인하였다. 따라서 본 논문에서 제안하는 시퀀스-투-시퀀스를 적용한 입력 오류 교정 모델의 입력 시퀀스로 입력된 문자열과 기기 측정값을 결합한 형태의 데이터를 사용하였다.

제안한 입력 오류 교정 모델의 성능 평가를 위해 성능 평가 지표인 정확도를 재정의 하였으며, 이를 토대로 입력 시퀀스가 기기 측정값을 포함한 모델과 그렇지

않은 모델에서의 정확도를 측정하였다. 성능 평가 결과, 기기 측정값을 포함한 모델에서 입력 오류 교정 모델의 정확도가 매우 향상되는 것을 할 수 있었다. 또한 실제 피실험자를 대상으로 제안된 모델이 적용된 웨어러블 입력 인터페이스를 사용하여 피실험자 군집별 입력 데이터를 측정·분석하였다. 이를 통해 피실험자 군집별 오류 패턴의 형태를 알 수 있었고, 오류 패턴에 따라 개선된 교정률을 확인하였다.

주제어 : 시퀀스-투-시퀀스, 철자 오류 교정, 웨어러블 입력 인터페이스

Abstract

Sequence-to-sequence model based error correction model for wearable input interfaces

Han, Seung-Eui
Depart of Computer Engineering
Graduate School

Supervised by Professor Kwak, Ho-Young

Areas of engineering and technology to the areas of daily services. Such research trends could also be found in Natural Language Processing which is a technology that enables a computer to process human languages. Spelling Error Correction, one of the areas of Natural Language Processing, is introduced in various search engines and word processing programs to support keyword spelling error correction or grammatical error correction in documents.

This paper proposes a sequence-to-sequence-based input error correction model to correct mistyped letters when entering words using wearable input interfaces. The inputted data and sensed data were collected using a wearable input interface as input sequence data for error correction model. Analyzing the collected data, the types of mistyped letters showed four different error patterns. This result proved that data sensed by the device is an important factor in correcting errors. Therefore, a combined form of inputted data and sensed data were used as the input sequence of the input error correction model.

For performance evaluation on the proposed error correction model, the accuracy, which is a performance evaluation index, was redefined. The accuracy of models was measured based on the redefined index where input sequences of the models differed by the presence of the sensed data. After the evaluation, the model with sensed data showed that the accuracy of the error correction model improved exceedingly. Also, input data classified by the test subjects were measured and analyzed using a wearable input interface with the model proposed for test subjects. From these results, I could find the types of error patterns by different test subjects, and corrections were made according to the error patterns.

Keyword : Sequence to Sequence, Spelling Error Correction, Wearable Input Interface

Ⅰ. 서 론

1. 연구 배경 및 목적

최근 머신러닝 기술은 공학·기술과 같은 연구 분야에 적용되는 것을 넘어 일상의 서비스에서도 익숙하게 접할 수 있을 정도로 그 범위가 넓어지고 있다. 그중에서도 머신러닝 기반의 자연어 처리(Natural Language Processing) 기술은 음성 인식이나 기계 번역, 챗봇, 추천 시스템 등 다양한 연구·산업 분야에 적용되고 있다.

자연어 처리란 인간의 언어인 자연어(Natural language)를 컴퓨터가 이해하고 처리할 수 있도록 하는 연구 및 응용 분야를 의미한다[1]. 기존 자연어 처리 연구의 통계적 언어모델(Statistical Language Model)은 통계 데이터에 없는 문장은 0%의 확률을 가지는 '데이터 희소성(Data Sparsity)'의 한계가 존재하였으나, 머신러닝 기술의 발전으로 신경망 기반의 언어모델(Neural Networks Language Model)의 등장으로 이러한 한계점을 보완하면서 빠르고 정확하게 대용량 문맥 정보의 처리가 가능하게 되었다[2].

이에 이러한 신경망 언어 모델을 철자 오류 교정에 활용하는 연구가 활발히 진행되고 있으며 이는 대체로 기존의 통계적 언어모델보다 더 나은 교정 효율을 보이고 있다[3,4]. 이는 실제 서비스에서도 활용되고 있으며, '네이버'에서는 딥러닝 기반의 검색어 교정 시스템인 'AIQSpell' 서비스를, '구글 독스(Google Docs)'는 기계 번역 기반 문법 교정 서비스를 제공하고 있다. 더 나아가 '네이버'에서는 스마트폰 키패드에서 사용자의 입력 패턴과 같이 오타를 유발하는 다양한 변인을 학습한 모델링을 탑재함으로써 오타 보정이 가능한 '스마트 보드' 애플리케이션을 서비스하는 등 딥러닝 기술을 활용한 다양한 철자 오류 교정 서비스가 제안되고 있다[5,6].

이에 본 논문에서는 웨어러블 입력 인터페이스를 이용하여 문장을 입력했을 때

잘못 입력된 철자를 교정할 수 있는 시퀀스-투-시퀀스(Seqence-to-Sequence)를 적용한 오류 교정 모델을 구현하고자 한다.

2 연구 내용

웨어러블 입력 인터페이스의 오류 교정 모델의 구현을 위해 먼저 웨어러블 기기의 구조와 기기에서 발생할 수 있는 오류의 패턴을 분석한다. 그리고 사용자의 오류 패턴을 반영하여 오류를 교정할 수 있도록 학습 데이터의 구조와 전처리 과정을 제안하고, 이를 시퀀스-투-시퀀스에 적용한 오류 교정 모델의 구조를 제시한다. 마지막으로 제안한 입력 오류 교정 모델의 성능을 평가하기 위해 실험을 통해 실제 사용자가 웨어러블 기기를 사용하여 입력한 데이터를 수집하고, 이를 바탕으로 교정의 정확도를 측정한다.

3 논문의 구성

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구로써 시퀀스 분석을 위한 신경망 모델의 일종인 RNN과 LSTM을 소개하고, 신경망 언어 모델에 대해 살펴본다. 3장에서는 본론으로 들어가 본 연구의 적용 대상인 웨어러블 입력 인터페이스에 대해 설명하며, 4장에서는 입력 오류 교정 모델을 위한 데이터 전처리 및 모델의 구조에 대해 설명한다. 이를 바탕으로 5장에서는 실험을 통해 제안한 오류 교정 모델의 성능을 평가하고, 마지막 6장에서 본 논문의 결론을 제시한다.

Ⅱ. 관련 연구

1 시퀀스 데이터 분석을 위한 신경망 모델

시퀀스 데이터(Sequence Data)란 데이터를 구성하는 요소들이 순서(Sequence)를 가지고 나열되어있는 것을 말한다. 시퀀스 데이터는 데이터 간 특정 순서를 가지고 있으므로 각각의 데이터가 독립되어 있지 않다. 즉, 과거의 데이터가 미래의 데이터에 영향을 미친다는 특징을 가지고 있다. 이러한 시퀀스 데이터를 기존의 독립 데이터를 학습하는데 사용되는 피드 포워드 신경망(Feed Forward Neural Network; FFNN)이나 컨볼루션 신경망(Convolution Neural Network; CNN)을 사용하여 학습 시킬 경우, 길이가 길어지거나, 가변적인 길이로 주어졌을 때 한계가 발생한다. 이에 시퀀스 분석을 위한 신경망 모델로 RNN(Recurrent Neural Networks; 순환 신경망)과 그의 변형 모델 LSTM(Long-Short Term Memory; 장단기 기억 신경망), GRU(Gated Recurrent Unit; 게이트 순환 유닛) 등의 모델들이 사용되고 있다.

1) RNN(Recurrent Neural Networks)

RNN[7]은 과거의 데이터를 기억하여, 이를 현재의 데이터에 반영되도록 함으로써 순서대로 나열되어있는 시퀀스 데이터를 학습시키는데 적합하다. RNN은 과거의 데이터를 기억하기 위해 특별한 구조를 갖는다. RNN의 은닉층은 입력된 데이터를 바탕으로 출력값을 계산한 뒤, 이를 출력층으로만 보내는 것이 아니라, 다시 다음 시점의 입력 데이터와 함께 은닉층으로 전달한다. RNN은 이러한 순환 구조를 바탕으로 과거 시점의 데이터를 계속해서 다음 데이터로 전달하게 된다. RNN의 구조는 그림 1과 같다.

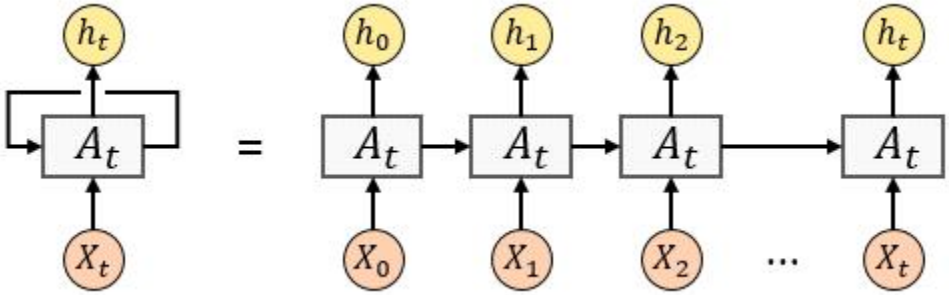


그림 1. RNN 구조()

그림 1에서 좌측은 RNN의 구조를 단적으로 표현한 것이며, 이를 시간 순서(time step)대로 펼치면 그림 1의 우측과 같이 표현할 수 있다. 각 타임 스텝에서 X_t와 같이 입력이 주어지면 은닉층 A_t에서는 X_t를 바탕으로 은닉 상태(hidden state) h_t를 출력한다. h_t는 현재 타임 스텝의 최종 출력을 계산하기 위해 출력층으로 전달되며, 동시에 다음 타임 스텝의 은닉층 A_{t+1}으로 전달되어 다음 타임 스텝의 입력 X_{t+1}에 영향을 미치게 된다. 이때 해당 구조에서 활성화 함수를 통해 결과 h_t를 출력하는 각각의 은닉층 노드를 셀(cell)이라고 부를 수 있다.

RNN에서 은닉 상태 값 h_t와 이를 바탕으로 출력층에서 계산되는 최종 출력 y_t는 아래의 식 1과 같이 정의된다. 현재 타임 스텝 t에서의 은닉 상태 값을 h_t, 입력 데이터를 x_t, 각각의 가중치를 W_x, W_h, 바이어스를 b로 두었을 때, h_t는 x_t와 이전 타임 스텝에서 전달된 h_{t-1}에 각각의 가중치를 곱한 값에 비선형 활성화 함수인 sigmoid 또는 tanh 연산을 거쳐 계산된다. y_t는 현재 타임 스텝에서 계산된 h_t에 출력 가중치 W_y를 곱한 값을 출력층의 활성화 함수 f의 연산을 거쳐 계산된다.

$$h_t = \tanh(W_x x_t + W_h h_{t-1} + b) \tag{1.1}$$

$$y_t = f(W_y h_t + b) \tag{1.2}$$

RNN은 타임 스텝이 길어질수록, 즉 기억해야 할 정보가 많아질수록 과거의 데이

터가 점점 흐려져 마지막 노드까지 전달되지 못하는 장기의존성(Long-term Dependency) 문제가 발생하기 때문에 긴 시퀀스를 학습하기에 적합하지 않다.

2) LSTM(Long-Short Term Memory)

학습시켜야 하는 시퀀스 데이터의 길이가 길어질 때, 초반의 데이터가 후반까지 잘 전달되지 않는다면 모델의 성능에 부정적인 영향을 미친다. LSTM[8]은 그림 2와 같이 RNN의 구조를 변형시켜 장기 의존성 문제를 해결함으로써 긴 시퀀스에 적합하도록 설계되었다. LSTM은 RNN의 은닉층에 forget(망각), input(입력), output(출력) 게이트를 추가하여 불필요한 지난 데이터는 삭제시키고, 기억해야 하는 현재 데이터의 양을 조절할 수 있는 구조를 갖는다.

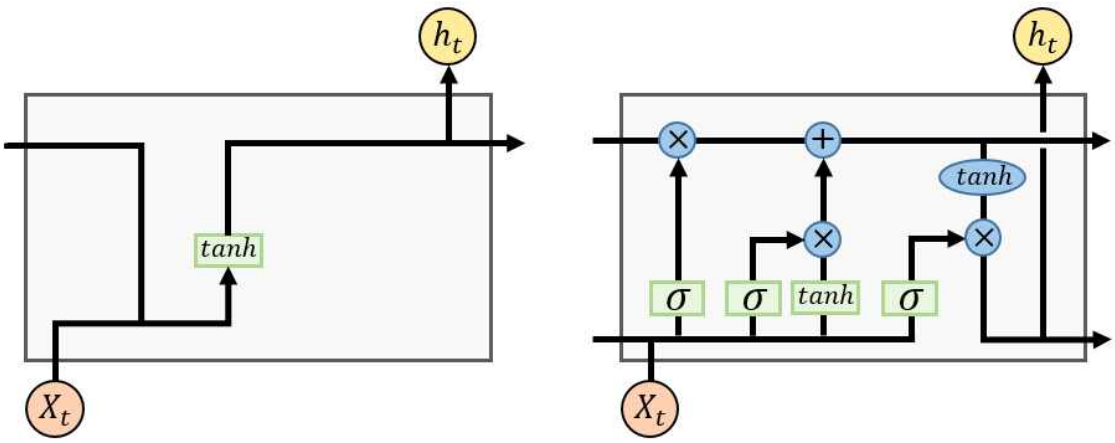


그림 2. RNN과 LSTM의 메모리셀 구조

각각의 게이트 역할은 다음과 같다. 먼저 forget gate는 현재 타임 스텝의 입력 데이터를 기반으로 이전 타임 스텝에서 전달된 데이터를 얼마나 잊을지 결정한다. input gate는 현재 타임 스텝에서 들어온 입력 데이터를 얼마나 기억할지 결정한다. 마지막으로 output gate는 현재 시점에서 input gate와 forget gate를 기반으로 계산된 총 데이터를 얼마나 은닉층의 출력으로 내보낼지 결정한다. 이러한 LSTM의 구조를 RNN과 비교하여 나타낸 그림은 그림 2와 같다. 은닉층에 추가된 forget, input, output gate는 sigmoid(σ) 연산을 통해 0~1 사이의 출력 f_t, i_t, o_t

를 가지며, 출력 결과가 0에 가까워질수록 각 게이트에 입력된 데이터가 최종 출력에 미치는 영향이 낮고, 1에 가까울수록 출력에 미치는 영향이 높다고 할 수 있다. LSTM은 f_t와 i_t를 기반으로 장기 기억에 해당하는 셀 상태(Cell State, C_t)를 계산하며, o_t를 기반으로 단기 기억에 해당하는 은닉 상태 h_t를 계산하여 다음 셀로 전달한다. 이를 수식으로 나타내면 아래의 식 2와 같다.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (2.1)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2.2)$$

$$\widetilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (2.3)$$

$$C_t = f_t * C_{t-1} + i * \widetilde{C}_t \quad (2.4)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (2.5)$$

$$h_t = o_t * \tanh(C_t) \quad (2.6)$$

forget gate에서는 이전 타임 스텝의 은닉 상태 h_{t-1}와 현재 타임 스텝의 입력 데이터 x_t에 가중치를 곱한 값을 입력으로 받아 sigmoid 연산을 거쳐 데이터의 삭제 정도를 출력 f_t로 갖는다. input gate에서는 h_{t-1}과 x_t의 가중치를 곱한 값을 sigmoid 연산을 거쳐 현재 데이터의 중요도 i_t를 출력으로 갖는다. 이는 현재 시점의 입력 데이터 흐름인 C'_t와 곱해진다.

현재 타임 스텝의 셀 상태 C'_t는 식 2.4와 같이 앞서 계산한 f_t에 이전 타임 스텝의 셀 상태 C_{t-1}을 곱한 값인 $f_t * C'_t$와, i_t에 현재 시점의 입력 데이터 흐름인 $\widetilde{C}_t$를 곱한 $i_t * \widetilde{C}_t$를 더해 계산된다. output gate에서는 h_t와 x_t에 각각의 가중치를 곱한 값을 sigmoid 연산을 거쳐 현재 시점의 C_t의 어느 부분을 출력할지를 결정하는 o_t를 계산한다, 최종적으로 현재 시점의 은닉층 출력값인 h_t는 o_t와 활성화 함수를 통과한 C_t를 곱한 값으로 결정된다.

3) GRU(Gate Recurrent Unit)

GRU[9]는 LSTM의 구조를 단순화시킨 변형 모델의 하나로, 그 구조는 그림 3과 같다. GRU는 LSTM과 같이 게이트를 사용하여 과거의 정보 중 불필요한 정보를 제거하되 2개의 게이트만 사용하도록 변형된 구조를 갖는다. GRU는 LSTM의 forget gate와 input gate를 통합한 update gate와 reset gate로 2개의 게이트를 사용한다. reset gate는 현재 시점에 과거의 정보를 얼마나 반영할지 결정하며, update gate는 과거의 정보를 얼마나 기억하고, 현재 시점의 입력을 보존할지 결정한다.

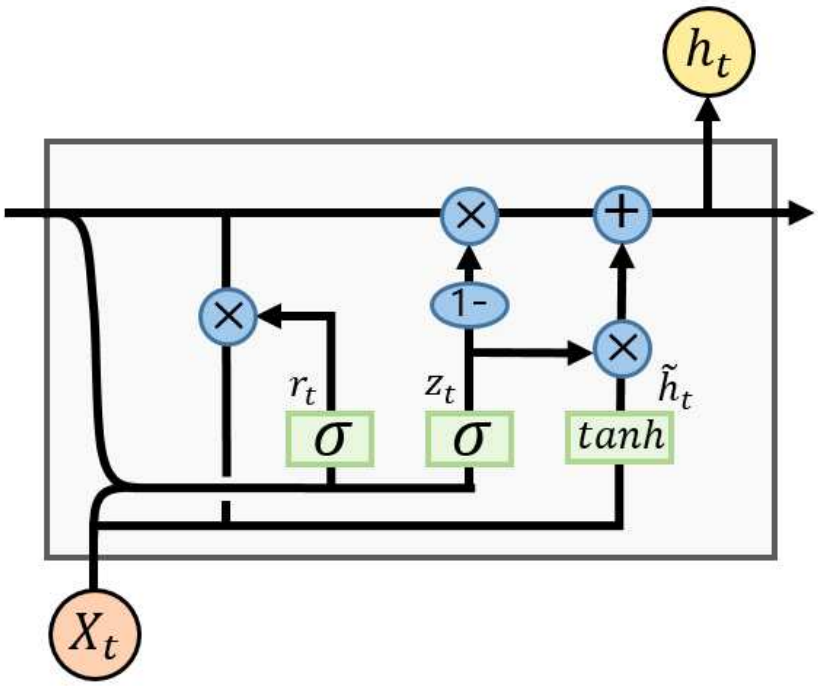


그림 3. GRU 구조

GRU에서는 reset gate의 출력값 r_t와 update gate의 출력값 z_t를 기반으로 현재 타임 스텝의 은닉 상태 h_t를 계산한다. 이때 GRU의 각 셀은 게이트를 2개로 통합시키면서 기존의 LSTM에서 장기 기억에 해당하는 셀 상태 C_t를 h_t로 흡수시켜 계산하는 방식을 취한다. 이를 수식으로 나타내면 아래의 식 3과 같다.

$$z_t = \sigma(W_z \bullet [h_{t-1}, x_t]) \tag{3.1}$$

$$r_t = \sigma(W_r \bullet [h_{t-1}, x_t]) \tag{3.2}$$

$$\tilde{h}_t = \tanh(W \bullet [r_{t-1} * h_{t-1}, x_t]) \tag{3.3}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \tag{3.4}$$

GRU는 LSTM보다 단순한 구조를 가지므로 학습 시 필요한 파라미터의 수가 적다는 장점이 있으나, 성능 측면에서 바라보았을 때, LSTM과 거의 같은 성능을 보인다[10].

2. 신경망 언어 모델

언어 모델은 식 4와 같이 n개의 단어로 구성된 문장 W가 주어졌을 때, 각 단어들이 순서대로 연결되어 전체 문장을 구성할 확률을 계산한다. 이는 순서대로 나열된 단어들 $w_1, w_2, ..., w_n$이 주어졌을 때, 다음에 올 단어를 연속적으로 예측하여 전체 문장 W를 구성할 확률을 계산함으로써 임의의 문장을 생성하거나, 오타를 검출하는 등의 자연어 처리 분야에서 널리 사용된다.

$$\begin{aligned} P(W) &= P(w_1)P(w_2|w_1)P(w_3|w_1,w_2)...P(w_n|w_1,w_2,...,w_{n-1}) \\ &= \prod_{i=1}^{n} P(w_i|w_1,w_2,...,w_{i-1}) \end{aligned} \quad (4)$$

언어 모델은 통계적 언어모델(Statistic Language Model)과 신경망 언어모델(Neural Networks Language Model)로 나뉜다. 기존의 자연어 처리에서 사용되던 통계적 언어모델은 수집된 데이터 내 해당 단어의 출현 빈도수를 기반으로 확률을 계산하는 언어 모델로, 특정 단어나 문장의 등장 빈도수가 적을 때 예측 효율이 떨어지는 데이터 희소성의 문제가 존재한다. 신경망 언어 모델은 딥러닝 알고리즘을 기반으로 확률을 예측하는 언어모델로, 단어 간 유사도를 반영하여 학습함으로써 기존의 통계적 언어모델이 가지고 있던 데이터 희소성 문제를 완화시켜 최근 자연어 처리 분야에서 활발히 연구되고 있다[10].

1) NNLM(Nerual Networks Language Model)

NNLM[11]은 가장 기본적인 신경망 구조인 FFNN(Feed Forward Neural Network) 기반의 언어 모델로, 그 구조는 그림 4와 같다. 입력값으로 학습시키고자 하는 문장을 정해진 N개의 단어로 분할하고, one-hot vector로 변환한 값을 사용한다. 이때 one-hot vector란 학습시키고자 하는 데이터 내 단어들에 고유한

인덱스를 부여하고, 단어 인덱스에 해당하는 요소는 1, 나머지의 요소는 0으로 하여 각 단어를 표현한 벡터를 말한다. NNLM에서는 각 단어의 one-hot-vector에 학습된 가중치를 곱하여 벡터의 차원을 밀집시킴과 동시에 각 단어의 특징과 유사도를 반영할 수 있도록 하는 '워드 임베딩' 과정을 거쳐 embedding vector를 계산하고, 이를 은닉층의 입력으로 사용한다. 이후 NNLM에서는 은닉층과 출력층을 거쳐 예측된 데이터를 cross-entropy 손실함수를 사용하여 가중치 행렬들을 재학습시킨다.

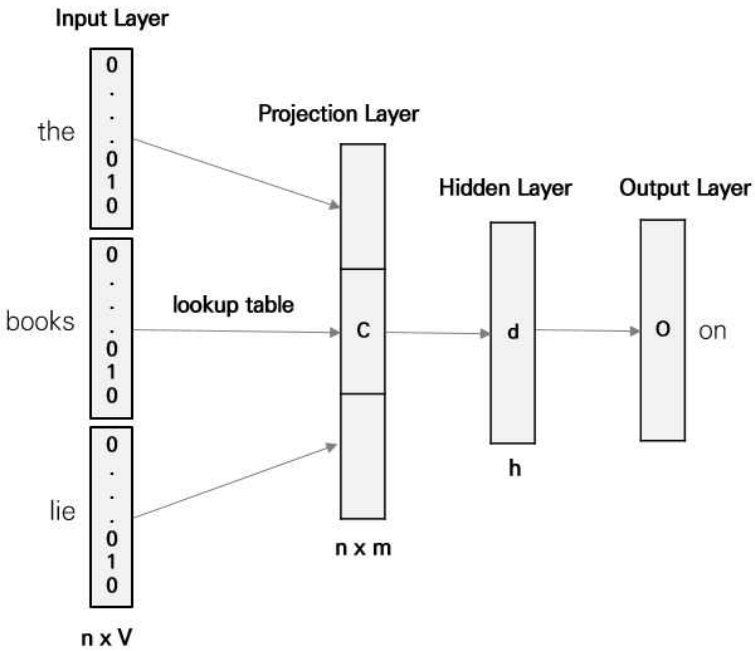


그림 4. NNLM 구조

이와 같은 NNLM의 구조로 인해 신경망 언어모델은 기존의 통계적 언어모델과 달리 단어 간 유사도를 학습함으로써 보다 나은 성능을 보인다. 그러나 NNLM은 입력의 형태가 고정된 FFNN 신경망을 기반으로 하고 있으므로 입력 단어의 개수가 N개로 고정되어 있어 가변적 길이의 문장이나 길이가 긴 문장을 학습하는 데 한계가 존재한다. 이후 시퀀스 데이터를 학습하기 위한 순환 신경망 구조가 확립되면서, 입력과 출력의 형태가 자유로운 RNN과 그 변형인 LSTM, GRU 기반의 언어모델이 연구되고 있다[12].

2) RNNLM(Recurent Neural Networks Language Model)

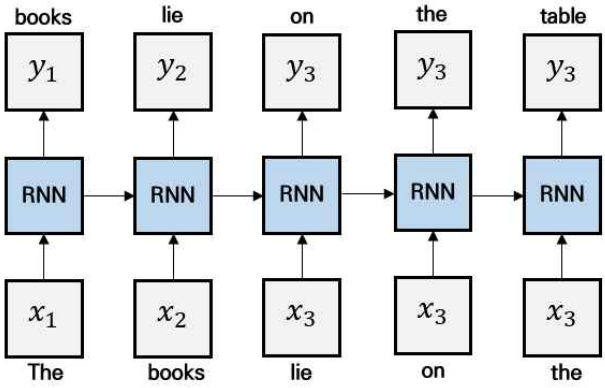


그림 5. RNNLM의 문장 예측 과정

RNNLM[13]은 RNN 또는 그의 변형인 LSTM이나 GRU를 사용하여 기본적으로 다음 단어에 대한 예측을 위하여 이전 시점의 출력을 현재 시점의 입력으로 전달되도록 한다. 즉, 그림 5와 같이 “The books lie on the table”과 같은 문장이 주어졌을 때, 가장 먼저 ‘The’를 입력 데이터로 ‘books’를 예측하며, 이 ‘books’는 다시 다음 시점의 입력이 되어 ‘lie’를 예측하는 과정을 연속적으로 진행함으로써 전체 문장의 확률을 계산한다.

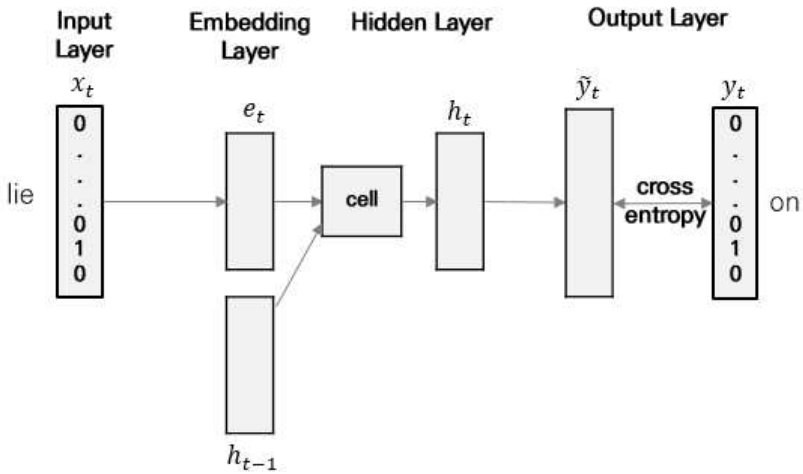


그림 6. RNNLM 구조

이러한 RNNLM의 구조는 그림 6과 같이 나타낼 수 있다. 학습을 주어진 문장 내 단어들을 one-hot vector로 변환시킨 후, 이를 각각 타임 스텝의 입력(x_t)으로 사용한다. 이때 입력된 one-hot vector는 임베딩 과정을 거쳐 임베딩 벡터(e_t)로 변환되며, 이는 RNN의 은닉층에 입력되어 이전 타임 스텝의 데이터와 함께 현재 타임 스텝의 은닉 상태(h_t)를 계산하는데 사용된다. RNNLM은 출력층의 활성화 함수인 softmax 함수를 거쳐 출력된 y_t에 손실함수인 cross entropy를 사용하여 오차를 계산한 뒤 가중치 행렬을 재학습시킨다.

3) char-level RNNLM

기존의 RNNLM의 단어 단위(word-level)의 입출력을 글자 단위(char-level)로 고려하기 위해 제안된 언어 모델[14,15]로, char-level RNNLM의 입출력 구조를 기존의 word-level RNNLM과 비교하여 나타내면 그림 7과 같다. word-level RNNLM은 주어진 문장을 각각의 단어로 분할하고, 이를 입력값으로 하여 다음에 올 단어를 출력하는 반면, char-level RNNLM은 주어진 문장을 각각의 글자로 분할하고, 이를 입력값으로 하여 다음에 올 글자를 출력한다.

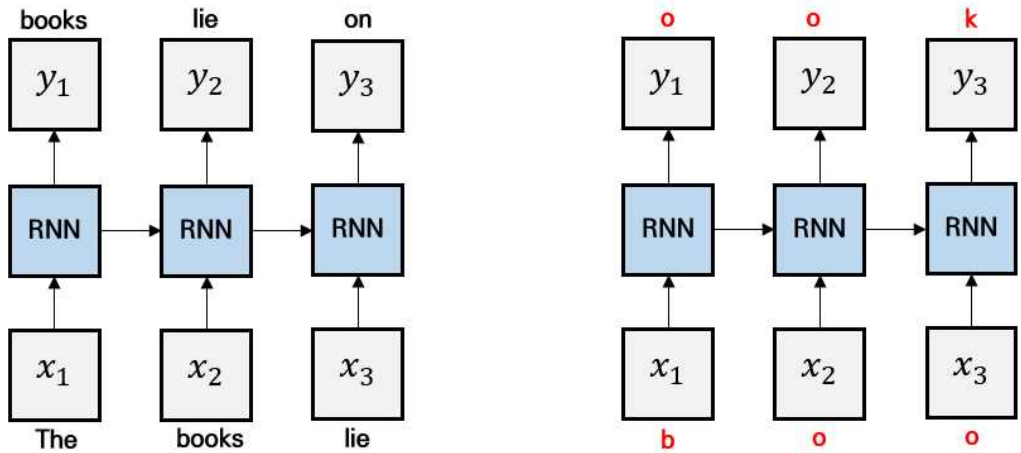


그림 7. word-level RNNLM과 char-level RNNLM 구조

이 과정에서 word-level RNNLM은 훈련 데이터 내의 모든 단어를 one-hot vector로 나타낼 경우, 성능에 한계가 발생하므로 일반적으로 임베딩 과정을 거친 embedding vector를 은닉층으로 전달한다. 그러나 char-level RNNLM의 경우, 훈련 데이터가 영어라는 가정하에 one-hot vector로 변환하면 각 글자를 대·소문자를 구분한 52개의 알파벳만을 특징으로 가진 벡터로 표현할 수 있으므로, 임베딩 과정을 생략할 수 있게 된다.

일반적으로 한 언어에서 문자의 개수가 단어의 개수보다 훨씬 적기 때문에 문자 단위로 언어 모델을 정의할 경우 단어 단위로 정의했을 때보다 적은 개수의 특징으로 RNN을 구성할 수 있는 특징이 있다.

Ⅲ. 웨어러블 입력 인터페이스 오류 패턴 분석

1. 웨어러블 입력 인터페이스 선행 연구

최근 스마트폰 시장이 성숙기에 진입함에 따라 다양한 형태의 웨어러블 디바이스가 새로운 스마트 기기의 시장으로 주목받고 있다[16]. 또한 증강 현실 및 가상 현실 기술이 발전함에 따라 사용자와 시스템 간 동적인 상호작용을 위하여 기존의 물리적 키보드나 마우스를 벗어나 다양한 형태의 입력 인터페이스가 연구되고 있다. 이들은 사용자의 움직임을 감지하기 위한 센서 기반의 입력기기뿐만 아니라 각 기기의 구조에 맞는 입력 방식 또는 더 나아가 개인화 알고리즘을 제안함으로써 사용자와 시스템 간의 공간적 제약을 벗어나 자유로운 상호작용을 지원한다.

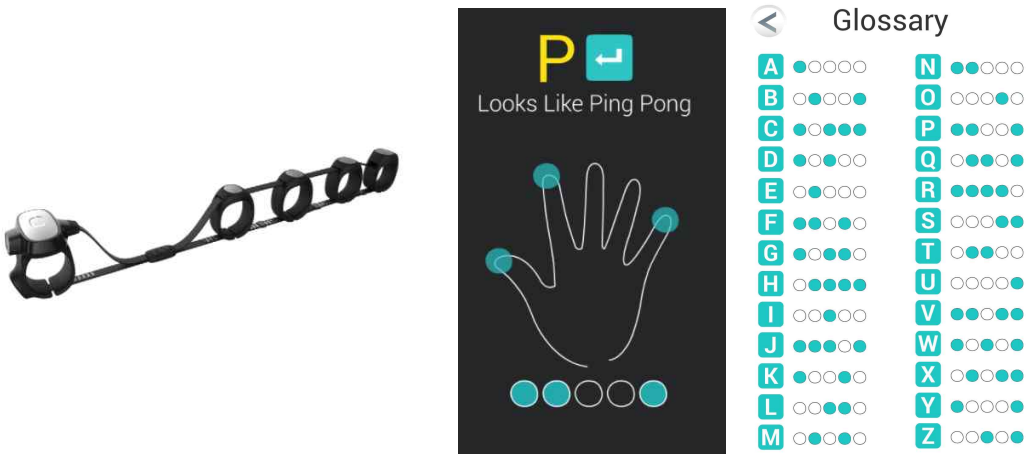


그림 8. 가속도 센서 기반 웨어러블 입력 인터페이스 (tap strap)

미국의 Tap System사는 그림 8과 같이 가속도 센서를 기반으로 한 손에 착용하는 손가락 스트랩 형태의 입력 인터페이스인 ‘tap strap’을 개발하여 출시하였다

[17]. 이는 다섯 손가락에 끼울 수 있는 링 구조를 하나로 엮은 스트랩 형태의 웨어러블 입력 인터페이스이다. 각각의 손가락 링에는 3축 가속도계가 장착되어 있으며, 엄지 링에는 6축 IMU 센서 및 광학 마우스 칩이 장착되어 있다. 각각의 센서는 손가락의 두들김에서 발생하는 가속도 및 각속도를 측정하여 이를 키 입력으로 사용한다. 또한 tap strap을 착용한 다섯 손가락의 두들김으로 모든 문자를 입력할 수 있도록 각 손가락의 두들김을 조합한 입력 방식을 제안하였으며, 자체적인 모바일 애플리케이션을 함께 제공함으로써 기기와 입력 방식의 개인화를 지원한다.

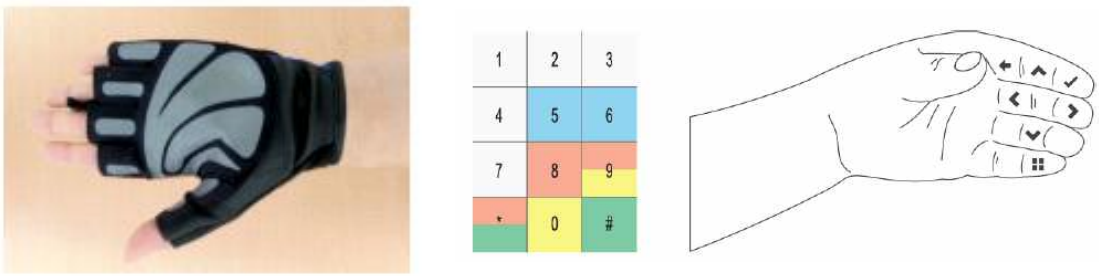


그림 9. 근전도 센서 기반 웨어러블 입력 인터페이스

손가락 움직임을 측정하여 입력 인터페이스를 구현하는 다른 방법으로 근전도 센서를 이용한 웨어러블 입력 인터페이스가 제안 및 연구되었다. 근전도 신호 측정을 통한 웨어러블 디바이스 입력 인터페이스 개발[18]에서는 그림 9와 같이 한 손에 착용하는 장갑 형태의 웨어러블 입력 인터페이스를 제안 및 개발하였다. 해당 기기의 손바닥 면에는 근전도 센서가 부착되어 있으며, 이를 착용한 손의 근육 변화를 측정하여 키 입력으로 사용한다. 해당 기기는 문자 입력 방식으로써 손마디 키패드[19]의 입력 방식을 차용한다. 이는 엄지를 제외한 손가락의 12마디에 0~9, *, #의 문자를 매핑하고, 엄지로 각 마디를 눌러 각 문자를 입력할 수 있도록 하는 방식이다. 이때 해당 연구에서는 손마디 키패드 입력 방식을 적용하여 제안한 입력 인터페이스에 대한 실제 사용자에 대하여 입력 실험 진행하였을 때, 피실험자별 키 입력 시 발생하는 근전도 신호가 각기 다름을 인지하였으며, 이에 입력에 대한 정확도를 높이고자 사용자별 입력 패턴을 분석하여 이를 바탕으로 입력 키의 위치를 조정하여 사용하는 개인화 방식을 사용한다.

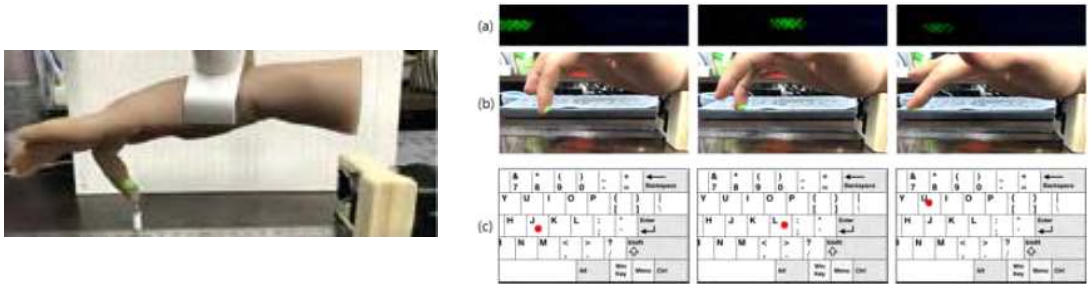


그림 10. 구조광 패턴 기반 웨어러블 입력 인터페이스

그림 10과 같이 손가락의 자체적인 움직임 측정을 벗어나 카메라를 통해 손가락의 움직임을 인식하는 형태의 입력 인터페이스 또한 개발 및 연구되었다. Vuylsteke 구조광 패턴을 이용한 손목 착용형 웨어러블 가상 입력 장치[20]에서는 적외선 레이저 패턴 프로젝트 및 적외선 카메라 기반의 손목 착용형 웨어러블 입력 인터페이스를 제안 및 개발하였다. 해당 기기는 장치에서 투사된 적외선 패턴이 구부러진 손가락에 반사되어 적외선 카메라에 입력되는 정보를 분석하여 손가락의 움직임을 인식하며, 이를 키 입력으로 사용한다. 해당 기기는 문자 입력 방식으로써 일반적인 키보드 배열인 QWERTY 자판을 차용하여 가상 키보드를 구현하였다. 이때 해당 연구에서 진행된 입력 실험에서는 한 손가락 입력의 동작만을 확인하였다.

2. 웨어러블 입력 인터페이스

본 논문에서는 전도성 섬유와 플렉스 센서를 기반으로 손마디 터치를 인식하여 키패드로 사용하는 장갑형 웨어러블 입력 인터페이스를 제안한다. 또한 문자 입력 시 발생하는 오류의 패턴을 분석을 통한 입력 오류 교정 모델을 제안함으로써 기기의 구조적 한계로 보완되지 않는 입력 오류를 보정하여 정확도를 높이고자 한다.

제안하는 손마디 터치 인식 웨어러블 입력 인터페이스의 구조는 그림 11과 같다. 해당 기기는 한 손에 착용하는 장갑 형태의 웨어러블 기기로, 엄지손가락을 사용하여 나머지 손가락의 열두 마디를 터치하는 방식의 키패드로 설계되었다. 손바닥 면에는 엄지를 제외한 손가락 마디 위치마다 전도성 섬유가 부착되어 있으며. 손등 면에는 휘어짐 정도를 측정할 수 있는 플렉스 센서가 손가락을 따라 부착되어 있다. 각각은 손등 중앙에 위치한 컨트롤러에 의해 제어되며, 전도성 섬유는 각 마디의 터치 입력을 인식하고, 플렉서블 센서는 터치 입력 시 각 손가락의 구부러짐 정도를 측정한다.

엄지로 손마디를 터치하는 과정은 키패드의 키를 누르는 과정과 동일하게 동작한다. 제안된 기기의 손마디 부위를 터치하면 해당 위치의 전도성 섬유가 터치를 인식하고, 동시에 네 손가락의 구부러짐이 플렉스 센서를 통해 측정된다. 기기에서는 최종적으로 두 가지 인식 과정을 결합하여 어떤 손마디가 터치되었는지 판단한다. 이는 어느 하나의 인식 과정에 전적으로 의존하지 않도록 하여 웨어러블 입력 기기에서의 입력 손실을 최소화한다.

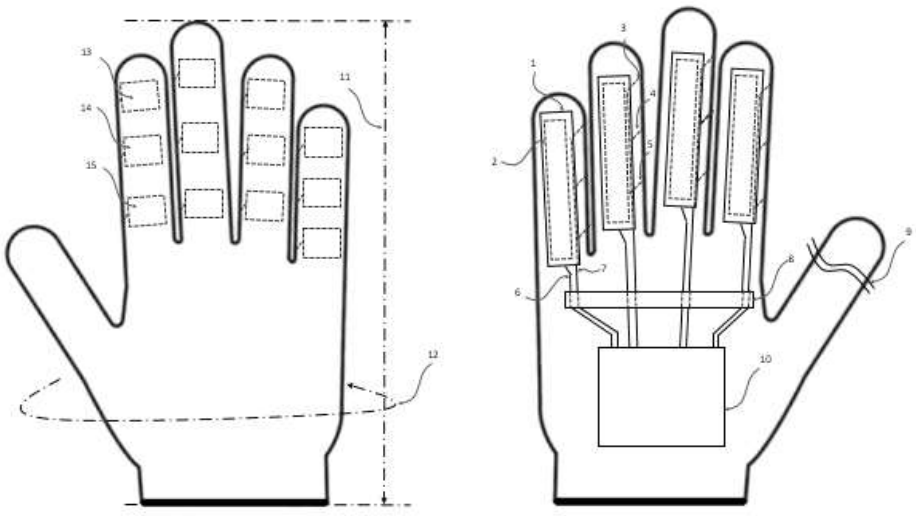


그림 11. 웨어러블 입력 인터페이스

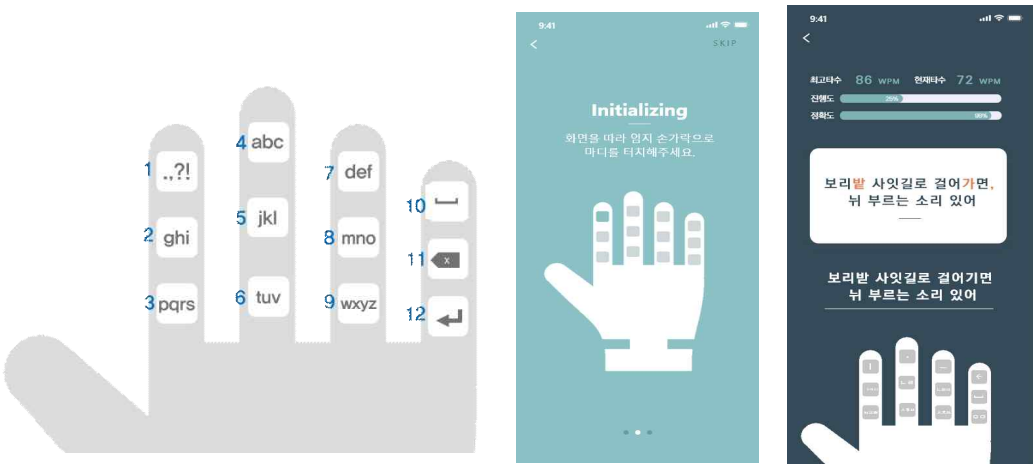


그림 12. 손마디 문자 입력 방식 키패드 구조 및 입력 애플리케이션

제안된 기기의 문자 입력 방식으로 3X4 휴대폰 자판 입력 방식[21]을 그림 12와 같이 구성하여 사용한다. 기기의 각 손마디에 알파벳과 특수문자는 3X4 키패드와 동일하게 배치하되, 다섯 번째 손가락인 소지의 커서 이동, 삭제, 개행과 관련된 특수 기호는 편의를 고려하여 재배치하였다. 이때 소지 위쪽 마디에 배치된 ‘ ␣ ’의 기능은 입력 커서 넘김을 의미한다. 이는 ‘aa’처럼 같은 마디의 글자를 연달아 입력할 때 ‘a’가 ‘b’로 바뀌지 않도록 입력 커서를 다음으로 넘겨준다. 소지 중간

마디에 배치된 문자는 입력 문자 삭제(Backspace)를 의미하고, 소지 아래쪽 마디에 배치된 문자는 줄 넘김(enter)을 의미한다.

이와 같이 장갑형 웨어러블 입력 기기의 손마디에 3X4 키패드 입력 방식을 적용함으로써 한 손 안에서 모든 문자 입력이 가능하며, 동시에 사용자가 쉽고 빠르게 기기에 적응할 수 있도록 한다. 그러나 3X4 키패드 입력 방식의 특성상, 입력해야 하는 문자 수에 비해 타수가 많아진다는 문제가 존재한다. 예를 들어 'ｓ'라는 문자를 입력하기 위해서는 'ｐ'-'ｑ'-'ｒ'-'ｓ'의 문자 입력 과정을 거쳐야 하며, 'hi'와 같이 동일한 마디에 배치된 문자를 연달아 입력하기 위해서는 ' ㄴ ' 기호를 포함해야 하므로 'hi'를 입력하기 위한 입력 과정은 'ｇ'-'h'- ㄴ -'ｇ'-'h'-'i'와 같다.

이처럼 한 개의 문자 입력을 위하여 같은 키를 연속적으로 터치해야 하는 3X4 키패드는 입력하고자 하는 문자열이 길어질수록 입력 오류가 증가한다. 따라서 제안된 기기로 많은 타수를 빠르게 입력할 경우, 기기의 손마디에 부착된 전도성 섬유나 손등에 부착된 플렉서블 센서값만으로는 입력의 정확도가 떨어진다. 이에 다음 절에서 기기를 사용하는 사용자가 실제 입력 과정에서 내는 오타 데이터를 수집하고, 이를 바탕으로 입력 오류의 패턴을 분석하고자 한다.

3. 웨어러블 입력 인터페이스의 오류 패턴 분석

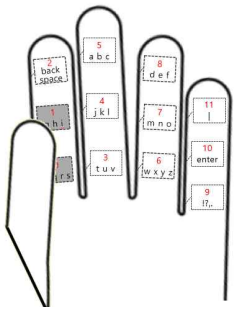
실제 기기의 사용과정에서 발생하는 오류 패턴에 대한 분석을 위하여 피실험자를 대상으로 제시된 문장을 따라 입력하도록 하였을 때 발생한 오타 데이터를 수집하였다. 이를 분석하였을 때, 입력 오류의 패턴은 같은 열 겹침 입력, 같은 행 겹침 입력, 입력 손실, 입력 초과의 네 가지로 분류되었다.

같은 열 겹침 입력 오류는 그림 13의 error type1과 같이 터치하고자 하는 손마디의 위 또는 아래 마디가 동시에 터치되어 입력되는 상황을 의미한다. 예를 들어, estab‘l’ished를 입력할 때 ‘l’ 아래의 ‘t’가 동시에 터치되어 estab‘jtk’ished로 잘못 입력된 경우가 여기에 해당된다. 이는 사용자가 마디와 마디 사이를 눌러 두 마디가 동시에 터치되거나, 손가락의 지나친 구부러짐으로 마디끼리 접촉되어 발생된다.

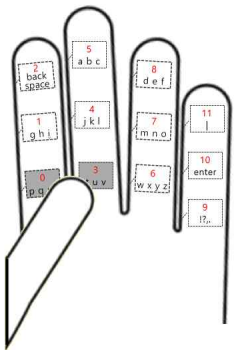
같은 행 겹침 입력 오류는 그림 13의 error type2와 같이 터치하고자 하는 손마디의 양 옆 마디가 동시에 터치되어 입력되는 상황을 의미한다. 예를 들어, o’p’pose를 입력할 때 ‘p’ 오른쪽의 ‘t’가 동시에 터치되어 o’pt’pose로 잘못 입력된 경우가 여기에 해당된다. 이는 사용자가 손가락과 손가락 사이를 눌러 두 마디가 동시에 터치되어 발생된다.

입력 손실 오류는 그림 13의 error type3과 같이 한 마디를 여러 번 누를 때, 누른 횟수보다 기기에서 인식된 터치 횟수가 더 적은 상황을 의미한다. 예를 들어 ‘f’or을 입력할 때 ‘f’에서 터치 횟수가 실제보다 적게 인식되어 ‘d’or로 잘못 입력된 경우가 여기에 해당된다. 이는 같은 마디를 빠른 속도로 연달아 터치할 때 전도성 섬유에서 엄지를 충분히 떨어뜨리지 않아 발생한다.

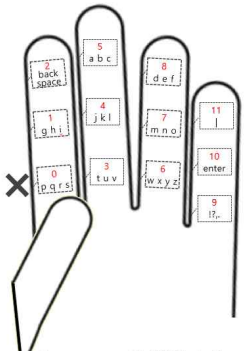
입력 초과 오류는 그림13의 error type4와 같이 한 마디를 여러 번 누를 때, 누른 횟수보다 기기에서 인식된 터치 횟수가 더 많은 상황을 의미한다. 예를 들어 faul‘t’를 입력할 때 ‘t’에서 터치 횟수가 실제보다 많이 인식되어 faul‘u’로 잘못 입력된 경우가 여기에 해당된다. 이는 같은 마디를 빠른 속도로 연달아 터치할 때 엄지의 떨림을 전도성 섬유에서 모두 터치로 인식하여 발생한다.



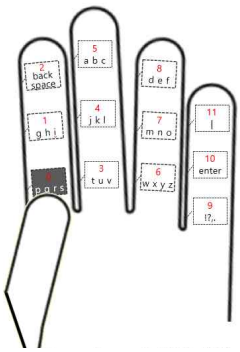
error type1. 같은 열 겹침 입력



error type2. 같은 행 겹침 입력



error type3. 입력 손실



error type4. 입력 초과

그림 13. 웨어러블 입력 인터페이스의 오류 패턴

제시된 네 가지 오류 패턴은 모두 특정 문자의 추가 또는 삭제로 인하여 발생한다. 오류 패턴 중 같은 행/열 겹침 입력이나 입력 초과 오류의 경우 다른 마디가 동시에 터치되어 불필요한 문자가 추가된 입력 오류이며, 입력 손실 오류는 마디를 터치하는 횟수가 부족함에 따라 필요한 문자가 삭제된 입력 오류이다.

이에 본 논문에서는 입력 오류가 발생한 단어를 하나의 입력 시퀀스로 하였을 때, 잘못 추가되거나 삭제된 문자를 바로잡아 교정된 단어가 출력 시퀀스로 나올 수 있도록 하고자 한다. 또한 주어진 입력 시퀀스에 각 문자를 입력할 때의 손가락 구부러짐 등의 데이터를 결합시켜 오류 교정 시 사용자의 입력 패턴을 같이 고려하는 입력 오류 교정 모델을 제안하고자 한다.

Ⅳ. 시퀀스-투-시퀀스를 적용한 오류 교정 모델

1. 시퀀스-투-시퀀스 모델

자연어 처리에서 LSTM 언어모델은 메모리셀로 LSTM을 사용함으로써 긴 문장을 학습하는 데 적합하다. 그러나 입력 시퀀스와 출력 시퀀스의 길이, 즉 입력 단어의 길이와 출력 단어의 길이가 같아야 한다. 이로 인해 입력 단어와 출력 단어의 길이가 다를 수 있는 본 논문의 오류 교정 모델에 적용하기에 한계가 있다. 이에 본 논문에서는 입력 단어의 길이와 출력 단어의 길이가 다른 구조에서 적용되고 있는 시퀀스-투-시퀀스에 LSTM 언어 모델을 적용하여 사용하고자 한다.

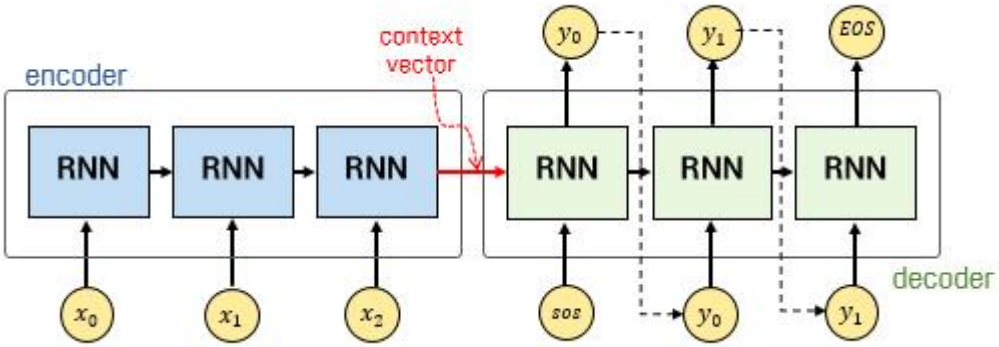


그림 14. 시퀀스-투-시퀀스 구조

시퀀스-투-시퀀스[22]는 입력 문장을 다른 나라의 언어로 번역하는 기계 번역과 같은 분야에 널리 사용되는 모델이다. 그 구조는 그림 14와 같이 각각 인코더(Encoder)와 디코더(Decoder)로 불리는 두 개의 순환 신경망이 연결된 구조이다. 인코더는 시퀀스 데이터를 입력으로 받아 고정된 크기의 문맥 벡터(context

vector)를 생성한다. 디코더는 문맥 벡터를 이전 메모리셀의 은닉 상태 값으로 하여, 미리 정해진 토큰(sos)을 첫 입력값으로 입력하였을 때 각각의 메모리셀의 출력을 결정한다. 결과적으로 시퀀스-투-시퀀스 모델은 학습된 입력 시퀀스와 출력 시퀀스가 문맥적인 의미가 동일하지만 출력의 길이는 다를 수 있게 된다.

따라서 본 논문에서는 기기의 입력 오류가 발생한 단어를 하나의 시퀀스 입력으로 하여 교정된 단어가 출력 시퀀스로 나올 수 있도록 LSTM 언어 모델을 적용한 시퀀스-투-시퀀스 기반의 입력 오류 교정 모델을 제안한다.

2. 시퀀스-투-시퀀스를 적용한 오류 교정 모델 데이터 구조

본 논문에서는 문장 입력 실험을 통한 피실험자의 실제 입력 데이터를 수집하여 시퀀스-투-시퀀스 모델에 적용할 학습 데이터로 사용한다. 입력 실험에는 COCA(Corpus of Contemporary American English)[23]에서 수집한 약 6,000개 단어를 정답으로 사용하였다. 입력 실험은 피실험자에게 각 문장을 타겟으로 제시하고 이를 따라 입력하도록 하는 과정으로 진행되었으며, 이 과정에서 사용자가 실제 입력한 문자열을 수집하였다. 문자열 이외에 기기에서 측정되는 사용자 데이터로는 그림 15와 같이 각 손마디를 터치할 때 인식된 마디를 숫자 형식으로 수집하였으며, 동시에 각 손가락의 구부러짐 정도와 문자 입력에 소요되는 시간을 계산하여 수집하였다. 수집된 입력 데이터의 형식을 요약하여 나타내면 아래의 표 1과 같다.

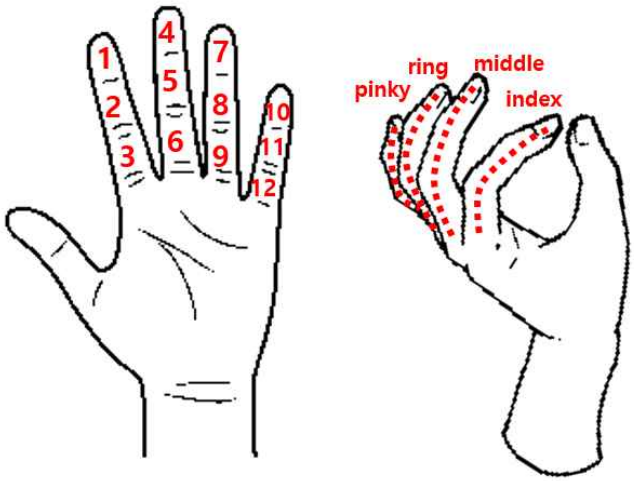


그림 15. 문자 입력 시 측정되는 문자열 이외의 사용자 입력 데이터

표 1. 수집 데이터 형식

<table>
<tr><th colspan="2">수집 데이터 형식</th></tr>
<tr><td>제시한 타겟 데이터</td><td>COCA에서 수집한 6,012개 단어로 구성된 타겟 데이터</td></tr>
<tr><td>수집된 사용자 데이터</td><td>하나의 문자 입력 시 수집되는 사용자 입력 데이터
<table><tr><th>데이터 특징</th><th>설명</th></tr><tr><td>signal</td><td>각 손마디에 부여된 고유 번호</td></tr><tr><td>index</td><td>검지의 구부러짐 정도</td></tr><tr><td>middle</td><td>중지의 구부러짐 정도</td></tr><tr><td>ring</td><td>약지의 구부러짐 정도</td></tr><tr><td>pinky</td><td>소지의 구부러짐 정도</td></tr><tr><td>alpha</td><td>입력된 실제 문자</td></tr><tr><td>time</td><td>문자 입력에 소요된 시간</td></tr></table></td></tr>
</table>

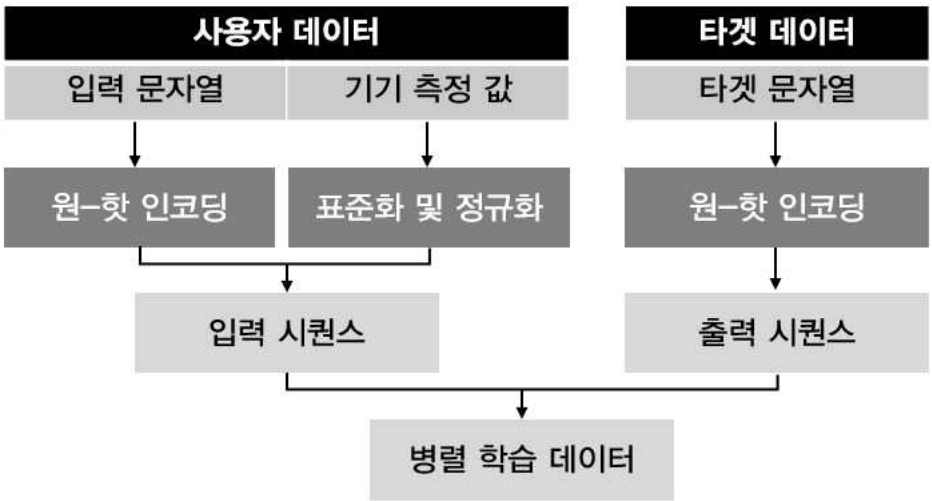


그림 16. 데이터 전처리 과정

시퀀스-투-시퀀스 모델을 학습시키기 위해서는 학습 데이터로 병렬 데이터가 필요하다. 즉, 인코더를 학습 시킬 입력 시퀀스와 디코더를 학습시킬 출력 시퀀스가 쌍을 이룬 형식의 병렬 데이터가 필요하다. 따라서 위에서 수집된 데이터를 사용자 데이터(user data)와 타겟 데이터(target data)를 쌍으로 하여 학습 데이터로 사용한다. 최종적으로 결정된 학습 데이터는 그림 16과 같이 전처리 과정을 통해 학습에 적합한 형태로 변형된다.

타겟 데이터는 문자열로만 구성되어 있으므로 원-핫 인코딩(one-hot-encoding)을 거쳐 벡터 형식으로 변환된다. 문자열과 기기 측정값이 섞여 있는 사용자 데이터의 경우, 각 데이터 형식에 맞는 전처리 과정을 거친 후 다시 결합된다. 사용자 데이터의 문자열은 타겟 데이터와 마찬가지로 원-핫 인코딩을 거친 벡터 형식으로 변환된다. 사용자 데이터의 기기 측정값은 문자열의 0과 1로 구성되어 있는 원-핫 벡터의 값과 큰 차이를 가지므로 이러한 차이를 줄이기 위하여 표준화 및 정규화 과정을 거친다. 데이터 특성에 따라 구분된 전처리 과정을 끝낸 사용자 데이터는 결합되어 하나의 입력 시퀀스를 구성한다. 마지막으로 시퀀스-투-시퀀스 모델 학습을 위하여 입력 시퀀스와 출력 시퀀스를 하나로 결합하여 병렬 형태의 학습 데이터를 구성한다. 최종적인 병렬 형태의 학습 데이터는 표 2 및 표 3과 같은 형식을 갖는다.

표 2. 입력 오류가 없는 학습 데이터 형식

학습 데이터 형식

제시 단어 **it**

입력 오류가 없는 데이터

	user data							target data
	signal	index	middle	ring	pinky	alpha	time	
학습 데이터	1	161	139	143	131	g	1.368	g
	1	158	139	143	131	h	0.260	h
	1	158	139	143	130	i	0.209	i
	3	158	117	141	130	t	1.209	t

표 3. 입력 오류가 있는 학습 데이터 형식

제시 단어 **fact**

입력 오류가 있는 데이터 (입력 횟수 초과 오류 발생)

	user data							target data
	signal	index	middle	ring	pinky	alpha	time	
학습 데이터	8	158	138	205	129	d	1.164	d
	8	158	139	191	129	e	0.319	e
	8	160	140	191	129	f	0.091	f
	5	158	317	152	128	a	0.572	a
	11	159	146	191	140	|	0.528	|
	5	161	274	148	129	a	0.489	a
	5	156	238	150	129	b	0.095	b
	5	159	257	150	128	c	0.002	c
	5	159	257	150	128	a	0.178	t
	3	156	113	144	129	t	1.806	

3. 시퀀스-투-시퀀스를 적용한 오류 교정 모델

본 논문의 입력 오류 교정 모델로써 시퀀스-투-시퀀스를 사용한다. 해당 모델은 입력 시퀀스로 표 2와 표 3의 사용자 데이터와 같이 입력 문자열과 기기 측정값을 결합한 형태의 사용자 데이터를 사용한다.

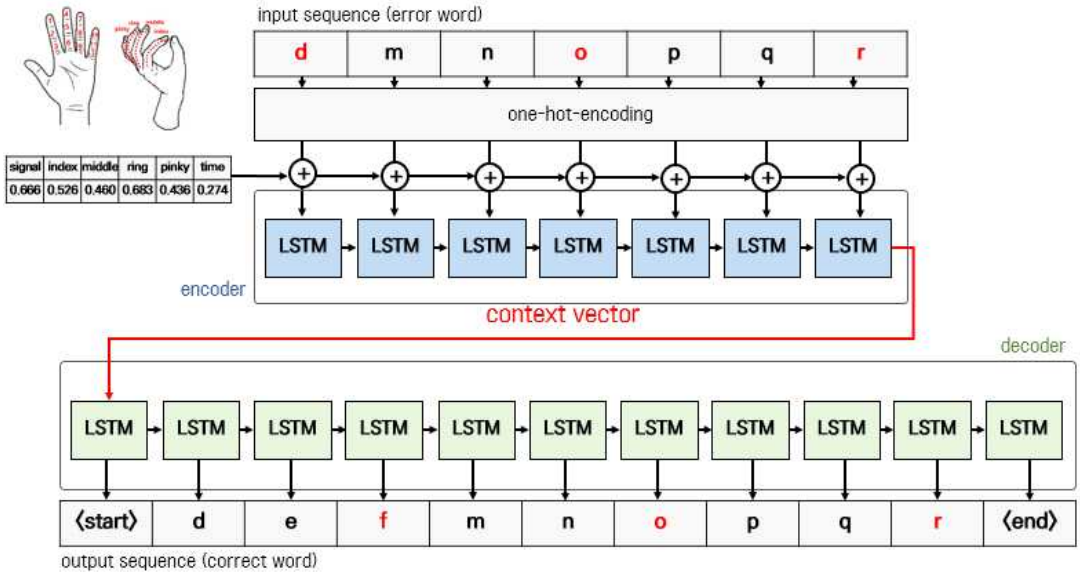


그림 17. 입력 시퀀스로 기기 측정값을 포함하는 시퀀스-투-시퀀스 모델 구조

입력 오류 교정 모델의 구조는 그림 17과 같다. 이는 'for'의 입력 과정에서 입력 손실이 발생한 'dor'의 교정 과정을 설명한다. 'dor'의 입력 문자열은 모델의 입력 시퀀스로 사용되기 전 원-핫 인코딩 과정을 거쳐 벡터화된 값을 갖는다. 이는 전처리된 기기 측정값이 결합되어 입력 시퀀스로 사용된다. 인코더에서는 해당 입력 시퀀스의 문맥 정보를 추출한 컨텍스트 벡터(context vector)를 디코더로 전달한다. 디코더의 가장 처음에 위치한 메모리셀은 컨텍스트 벡터와 시작 문자열인 <START>를 바탕으로 다음에 올 문자를 예측한다. 이러한 예측 결과는 다시 다음 메모리셀로 전달되며, 종료 문자열인 <END>가 출력될 때 까지 위의 과정을 계속

해서 반복한다. 따라서 입력 오류 교정 모델은 최종적으로 <START> 기호부터 <END> 기호를 포함한 교정된 문자열 즉, ‘for’에 대한 문자열을 최종 출력값으로 갖는다.

Ⅴ. 실험 및 평가

1 실험 데이터 세트 및 환경 변수

모델 훈련 및 검증 데이터 세트로 COCA의 문장에서 특수 기호를 제거한 6,021개의 타겟 단어를 수집하고, 이를 사용자가 따라 입력하도록 하였을 때의 문자열 및 기기 측정값을 수집하였다.

타겟 단어는 최대 12개, 평균 7.2개의 길이를 가지며 오타가 없는 정답 단어만을 사용한다. 사용자 데이터는 입력 과정에서 발생하는 모든 글자를 포함하므로 최대 41개, 평균 23.4개의 길이를 가지며 입력 오류가 발생한 단어를 포함한다, 이때 수집된 6,021개의 사용자 데이터 중 입력 오류가 있는 데이터의 비율은 22.06%이며, 이 중 같은 행 겹침 입력에 대한 비율이 12.53%로 가장 높았으며, 같은 열 겹침 입력이 2.50%, 입력 손실이 1.94%, 입력 초과가 5.09%의 비율을 차지하였다.

이와 같이 수집된 타겟 데이터와 사용자 데이터는 전처리 과정을 거쳐 한 쌍의 병렬 데이터로 구성하였으며, 전체 데이터에서 8:2의 비율로 훈련 데이터와 검증 데이터를 나누어 사용하였다.

제안된 모델의 인코더 임베딩 차원은 35, 디코더 임베딩 차원은 32이며, 메모리 셀로 사용되는 LSTM의 은닉 차원은 256으로 설정하였다. 또한 훈련 파라미터로써 학습률은 0.001, 에포크(epoch)는 200으로 설정하였으며, 배치 사이즈는 각각 32, 64로 설정하여 성능을 비교하였다.

2 시퀀스-투-시퀀스를 적용한 오류 교정 모델 성능 평가

모델의 성능 평가 지표로 전체 정확도, 예측 정확도, 교정 정확도를 사용하고자 한다. 식 5.1의 전체 정확도(Acc_{total})는 검증 데이터인 사용자가 웨어러블 입력 인터페이스를 사용하여 입력한 전체 단어에 대해 단어를 정확히 예측하는 비율을 의미한다. 여기서 검증 데이터는 오류가 포함된 단어($word_{corr}$)와 오류를 포함하지 않은 단어($word_{err}$)를 말한다. 식 5.2는 예측 정확도(Acc_{corr})로 오류가 포함되지 않은 단어에 대해 정확히 예측하는 비율을 의미한다. 식 5.3은 교정 정확도(Acc_{err})를 나타내며 오류가 포함된 단어에 대해 정확히 예측하는 비율을 의미한다.

$$Acc_{total} = \frac{predict_{correct}}{word_{corr} + word_{err}} \tag{5.1}$$

$$Acc_{corr} = \frac{predict_{correct}}{word_{corr}} \tag{5.2}$$

$$Acc_{err} = \frac{predict_{correct}}{word_{err}} \tag{5.3}$$

이를 바탕으로 제안된 모델의 성능 평가를 위해 기기 측정값을 포함한 모델과 포함하지 않은 모델의 정확도를 측정하고자 한다.

표 4. 입력 오류 교정 모델의 정확도

입력 오류 교정 모델의 정확도						
	batch size=32			batch size=64		
	전체정확도	예측정확도	교정정확도	전체정확도	예측정확도	교정정확도
기기 측정값 미포함	26.37%	30.22%	12.50%	61.92%	69.60%	34.87%
기기 측정값 포함	77.66%	88.59%	64.80%	64.53%	69.13%	48.36%

표 4는 입력 오류 교정 모델의 성능 평가 결과로, 훈련 파라미터에서 batch size를 32로 설정했을 경우 기기 측정값을 포함하지 않은 모델의 전체 정확도는 26.37%, 예측 정확도는 30.22%, 교정 정확도는 12.05%였고, 기기 측정값을 포함한 모델의 전체 정확도는 77.66%, 예측 정확도는 88.59%였으며, 교정 정확도는 64.80%로 기기 측정값을 미포함한 경우보다 모든 정확도가 높은 것을 알 수 있었다. 또한 batch size를 64로 설정했을 경우 기기 측정값을 포함하지 않은 모델의 전체 정확도는 61.92%, 예측 정확도는 69.60%, 교정 정확도는 34.87%였고, 기기 측정값을 포함한 모델의 전체 정확도는 64.53%, 예측 정확도는 69.13%, 교정 정확도는 48.36%의 성능을 나타내었다. batch size 64에서 기기 측정값을 미포함한 경우 성능이 더 좋아지는 것을 확인할 수 있었다.

표 4의 성능 평가 결과를 살펴보면 전체 정확도가 80%를 넘지 못하며 batch size 32, 64에서 다른 성능 평가 결과를 나타내었다. 이에 그림 18과 같이 출력 시퀀스가 교정하고자 하는 단어 형태인 경우를 입력 오류 교정 모델에 적용하여 모델의 정확도를 측정해보고자 한다.

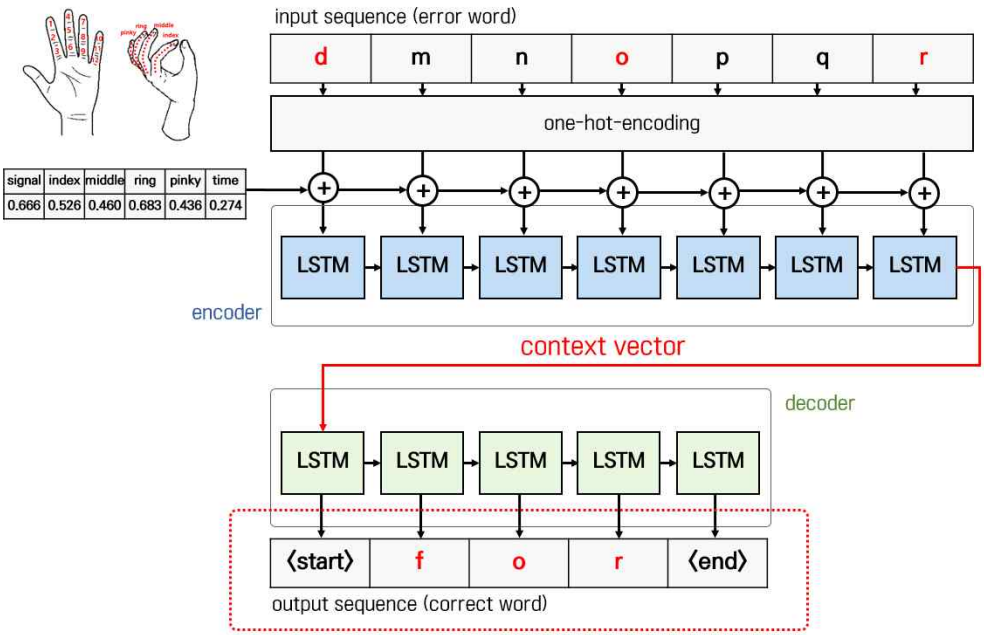


그림 18. 출력 시퀀스가 단어 형태인 입력 오류 교정 모델

출력 시퀀스가 교정하고자 하는 단어 형태인 경우를 입력 오류 교정 모델에 적용했을 때, 기기 측정값을 포함한 모델과 포함하지 않은 모델의 정확도는 표 5와 같이 측정되었다.

표 5. 출력 시퀀스가 단어 형태인 입력 오류 교정 모델의 정확도

출력 시퀀스가 단어 형태인 입력 오류 교정 모델의 정확도						
	batch size=32			batch size=64		
	전체정확도	예측정확도	교정정확도	전체정확도	예측정확도	교정정확도
기기 측정값 미포함	89.07%	95.79%	65.46%	82.96%	88.77%	62.5%
기기 측정값 포함	89.45%	94.76%	74.67%	81.57%	89.80%	58.12%

표 5를 살펴보면 훈련 파라미터에서 batch size를 32로 설정했을 때, 기기 측정값을 포함하지 않은 모델의 전체 정확도는 89.07%, 예측 정확도는 95.79%, 교정 정확도는 65.46%였다. 기기 측정값을 포함한 모델의 표 4의 정확도보다 매우 높은 것을 확인할 수 있었다. 또한 batch size를 64로 설정했을 경우에도 표 4보다 좋아지는 것을 알 수 있었다. 표 4와 표 5의 성능 평가 결과, 출력 시퀀스로 교정하고자 하는 단어 형태를 적용하였을 때 전체 성능이 향상되는 것을 알 수 있었으며 batch size를 32로 설정하였을 때 기기 측정값을 포함한 모델과 포함하지 않은 모델 모두 89% 이상의 전체 정확도를 나타낸 것을 알 수 있었다

3 웨어러블 입력 인터페이스에 적용한 오류 교정 모델의 성능 평가

본 논문에서 제안한 입력 오류 교정 모델의 교정률을 평가하기 위해 피실험자를 대상으로 출력 시퀀스가 단어 형태인 기기 측정값 포함 모델을 적용했을 때 오류 패턴별 교정 정확도와 모델 적용 전·후의 정확도를 측정하고자한다.

입력 데이터 수집을 위한 피실험자는 모두 3X4 키패드에 대한 경험이 있는 자를 대상으로 하였으며 웨어러블 기기 경험 여부와 평균타수에 따라 네 군집으로 나누어 입력 데이터를 수집하였다. 이때 웨어러블 기기 경험 여부가 있다는 것은 입력 데이터 수집 전 웨어러블 기기를 사용한 타자연습을 통해 기기와 입력 자판에 익숙해진 상태를 의미한다. 각 피실험자에서 수집한 입력 데이터 중 웨어러블 기기를 연습한 경험이 있는 숙련자 중 평균 타수 이상의 피실험자의 입력 데이터를 A 군집, 평균 타수 이하의 입력 데이터를 B 군집으로 정의한다. 또한 연습 경험이 없는 피실험자 중 평균 타수 이상인 피실험자의 입력 데이터를 C 군집, 평균 타수 이하인 피실험자의 입력 데이터를 D 군집으로 정의한다. 각 군집은 200~250개의 단어를 입력하였을 때 올바르게 입력된 데이터와 오류가 발생한 데이터를 모두 가지고 있으며, 이 중 오류 데이터에 대한 통계 자료와 그래프는 표 6과 그림 19와 같다.

표 6. 피실험자 군집별 오류 데이터 통계

	총 데이터 수	오류 데이터 수	같은 행 겹침	같은 열 겹침	입력 손실	입력 초과
A	256개	53개	32개	9개	9개	3개
B	228개	33개	2개	7개	12개	12개
C	219개	36개	7개	4개	4개	21개
D	231개	77개	26개	2개	2개	47개
합계	934개	199개	67개	22개	27개	83개

그림 19의 피실험자 군집별 오류 데이터 통계 그래프를 살펴보면 사용자마다 다양한 오류 패턴을 가지며 사용자가 웨어러블 입력 인터페이스를 사용하는데 있어 발생하는 특정 오류가 인터페이스 자체에 있다기보다 사용자들의 다양한 입력 특징에 있다는 것을 알 수 있다.

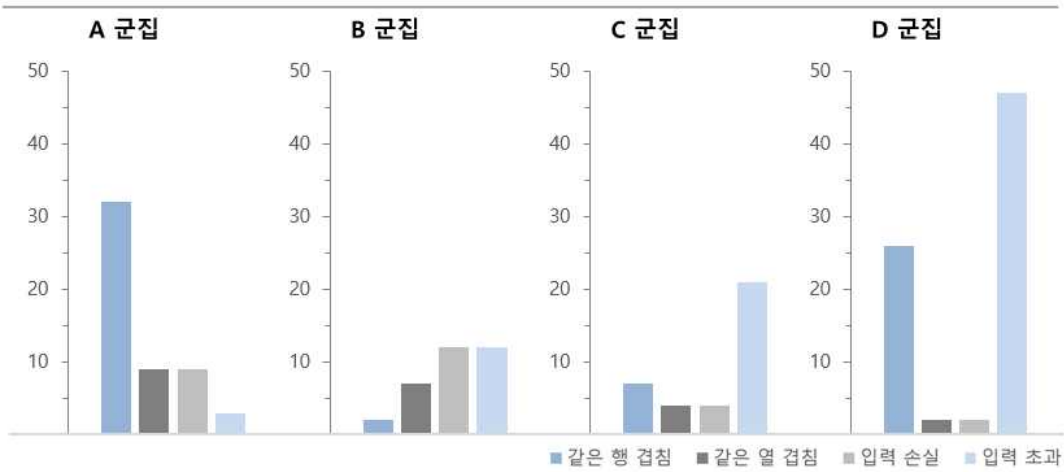


그림 19. 피실험자 군집별 오류 데이터 통계 그래프

표 7. 오류 패턴 별 교정 성공률

	오류 데이터 수	교정된 데이터	교정 정확도
같은 행 겹침 입력	67개	42개	62.69%
같은 열 겹침 입력	22개	20개	90.90%
입력 손실	27개	20개	74.07%
입력 초과	83개	56개	67.47%

모든 피실험자의 오류 데이터 199개에 대해 입력 오류 교정 모델을 적용하였을 때, 교정 성공률 즉 교정 정확도는 표 7과 같다. 이는 웨어러블 입력 인터페이스를 사용하는 사용자가 발생시킬 수 있는 4가지 오류 패턴에 대해 본 논문에서 제시한 입력 오류 교정 모델이 각각의 오류 패턴에서 어느 정도의 교정 성공률을 보이는지 나타낸다. 같은 행 겹침 입력 오류 데이터에 모델을 적용했을 때, 67개 데이터 중 42개 단어의 교정에 성공하여 62%의 교정 성공률을 보였으며, 입력 초과의 경우 교정 정확도가 60% 이상이었고, 입력 손실의 경우 74%, 같은 열 겹침 입력의 경우 90%로 같은 열 겹침 입력의 경우 교정 성공률이 가장 높게 나타나는 것을 알 수 있었다.

마지막으로 웨어러블 입력 인터페이스에 본 논문에서 제안한 오류 교정 모델을 적용하였을 때와 적용하지 않은 경우에 대한 정확도를 측정하여 성능 향상에 대해 비교·분석하고자 한다.

표 8. 입력 오류 교정 모델 적용 전/후 피실험자 군집별 입력 정확도

	교정 전 전체 정확도	교정 후 정확도		
		전체 정확도	예측 정확도	교정 정확도
A	(203/256) 79.30%	(242/256) 94.53%	(195/203) 96.06%	(47/53) 88.68%
B	(195/228) 85.53%	(208/229) 91.22%	(183/195) 93.85%	(25/33) 75.76%
C	(183/219) 83.56%	(209/219) 95.43%	(174/183) 95.08%	(35/36) 97.22%
D	(154/231) 66.67%	(177/231) 76.62%	(146/154) 94.80%	(31/77) 40.26%

표 8은 피실험자 군집별 오류 교정 모델을 적용하였을 경우 전체 정확도, 예측 정확도, 교정 정확도를 나타낸 것이다. 교정 전 정확도는 피실험자가 입력한 전체 250개 단어에서 사용자가 입력한 단어 중 오류를 발생한 단어의 비율을 의미한다. 각 군집별 오류 교정 모델을 적용하였을 때의 정확도를 살펴보면 A, B, C 군집에서 A 군집의 경우 B 군집보다 약 6%의 입력 오류를 더 발생시켰으나 오류 교정 모델 적용 후 성능 평가에서는 전체 정확도에서는 약 3%, 교정 정확도에서는 약 13% 더 높은 정확도를 보였다. 마찬가지로 C 군집 역시 B 군집보다 약 2%의 입력 오류를 더 발생시켰으나, 전체 정확도에서는 약 4%, 교정 정확도에서는 21% 더 높은 성능을 보였다.

따라서 교정 모델이 B 군집에 비해 A, C 군집에서 더 나은 교정 성능을 나타낸 것을 알 수 있었다. 이때 B 군집은 오류 패턴의 비율이 균일한데 비해, A, C군집은 각각 같은 행 겹침 입력, 입력 초과와 같이 특정한 오류 패턴이 매우 높게 발생하였음을 볼 수 있다. 이를 통해 교정 모델이 특정 오류 패턴 비율이 높은 실험자에서 더 나은 성능을 보임을 알 수 있었다.

D 군집의 경우 오류 교정 모델 적용 이후 정확도가 약 10% 향상되었으며, 예측 정확도는 94%로 다른 군집과 2%의 차이를 두고 비슷한 성능을 보였으나, 교정 정확도는 50% 이하로 크게 낮은 성능을 나타냈다. D 군집의 오류 패턴에서 가장 큰

비율을 차지하는 입력 초과 오류 데이터를 살펴보았을 때, 다른 군집에서 발생한 입력 초과 데이터에서는 평균 2.5개의 글자가 더 추가되어 발생했다면, D 군집의 경우는 평균 6개, 최대 12개의 글자가 더 추가되어 있음을 발견할 수 있었다. 따라서 D 군집에서 교정 모델이 낮은 성능을 보이는 것은 데이터의 길이가 길어질수록 예측의 정확도가 감소하는 기울기 소실로 인해 발생하는 문제로 판단된다.

결과적으로 본 연구에서 제안한 오류 교정 모델은 특정 오류 패턴의 비율이 높고, 오류로 인해 추가된 데이터가 일정 개수 이하인 사용자 데이터에서 가장 높은 전체 정확도를 가짐을 알 수 있었다.

Ⅵ. 결 론

본 논문에서는 웨어러블 입력 인터페이스를 사용하여 문장을 입력할 때 오류에 의해 잘못 입력된 철자를 교정하기 위해 시퀀스-투-시퀀스를 적용한 입력 오류 교정 모델을 제안하였다. 이를 구현하기 위해 실제 사용자가 웨어러블 입력 인터페이를 사용하여 입력한 문자열과 기기에서 측정된 데이터를 수집하였다. 수집한 데이터의 분석을 통해 입력 문자열의 철자 오류가 발생하는 형태가 4가지의 오류 패턴을 보인다는 것을 확인할 수 있었다. 이를 통해 사용자가 문자를 입력할 때 측정되는 기기 측정 데이터는 입력 오류를 교정하는 데 중요한 요소임을 확인하였다. 따라서 본 논문에서 제안하는 시퀀스-투-시퀀스를 적용한 입력 오류 교정 모델의 입력 시퀀스로 입력된 문자열과 기기 측정값을 결합한 형태의 데이터를 사용하였다.

제안한 입력 오류 교정 모델의 성능 평가를 위해 성능 평가 지표인 정확도를 재정의 하였으며, 이를 토대로 입력 시퀀스가 기기 측정값을 포함한 모델과 그렇지 않은 모델에서의 정확도를 측정하였다. 성능 평가 결과, 기기 측정값을 포함한 모델에서 입력 오류 교정 모델의 정확도가 매우 향상 되는 것을 할 수 있었다. 또한 실제 피실험자를 대상으로 제안된 모델이 적용된 웨어러블 입력 인터페이스를 사용하여 피실험자별 입력 데이터를 측정·분석하였다. 이를 통해 피실험자별 오류 패턴의 형태를 알 수 있었고, 오류 패턴에 따라 개선된 교정률을 확인하였다.

그러나 제안한 입력 오류 교정 모델에서 오류 패턴별 교정 정확도를 비교하였을 때, 교정 정확도가 가장 높은 오류 패턴은 같은 열 겹침 입력 오류 패턴으로 교정률이 90% 이상이었으며, 가장 낮은 교정 정확도를 갖는 같은 행 겹침 입력 오류 패턴의 교정률은 62%로 상이한 교정 결과를 보였다. 따라서 향후 연구에서는 입력된 전체 문장에서 문맥 오류를 검출 후 교정할 수 있는 알고리즘과 함께 오류 패턴별 교정 정확도를 높일 수 있는 특성화 된 모델 구현에 대한 연구가 필요하다.

참 고 문 헌

[1] Gobina G.chowdhury, “natural language processing” Annual Review of Information Science and Technology, Vol. 37, Issue 1p, pp. 51-89, 2003

[2] 김윤덕, “Word2Vec을 이용한 위키피디아 텍스트 데이터 분석 시스템 구현”, 숭실대학교 소프트웨어특성화대학원 석사학위논문, 2017

[3] Inchul Kang, Do-Gil Lee, “Context-based Spelling Error Correction using Sequence to Sequence with Attention” Proceedings of KIIT Conference, pp. 57-61, 2020

[4] JUNG-HUN LEE, MINHO KIM, AND HYUK-CHUL KWON, “Deep Learning-Based Context-Sensitive Spelling Typing Error Correction” IEEE Access, Vol. 8, 2020

[5] Google Cloud, "beyond spell check, how Google Docs is smart enough to correct grammar", https://cloud.google.com/blog/products/g-suite

[6] Naver, “Smart Board”, https://k.search.naver.com/

[7] Elman, Jeffrey L. "Finding structure in time." Cognitive science, Vol. 14, No. 2, pp. 179-211, 1990

[8] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation Vol. 9, No. 8, pp. 1735-1780, 1997

[9] Chung, Junyoung, et al. "Empirical evaluation of gated recurrent neural networks on sequence modeling." arXiv preprint arXiv:1412.3555, 2014

[10] Britz, D., Goldie, A., Luong, M. T., & Le, Q. “Massive exploration of neural machine translation architectures.” arXiv preprint arXiv:1703.03906, 2017

[11] Bengio, Yoshua, et al. "A neural probabilistic language model." The journal of machine learning research, Vol. 3, pp. 1137-1155, 2003

[12] Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781, 2013

[13] Mikolov, Tomas, et al. "Recurrent neural network based language model." Interspeech. Vol. 2, No. 3, 2010

[14] Kim, Yoon, et al. "Character-aware neural language models." Thirtieth AAAI conference on artificial intelligence, 2016

[15] Verwimp, Lyan, Joris Pelemans, and Patrick Wambacq. "Character-word LSTM language models." arXiv preprint arXiv:1704.02813, 2017

[16] NIA, “웨어러블 디바이스 기반의 창조경제 활성화 전략” IT & Future Strategy 보고서, 2014-제6호

[17] Tap System Inc, https://www.tapwithus.com/

[18] 김민지, “근전도 신호 측정을 통한 웨어러블 디바이스 입력 인터페이스 개발” 인제대학교 석사학위논문, 2015

[19] 정의태, “근전도 측정을 통한 한 손 동작 입력 방식의 제안” 디지털디자인학연구, 14.1, pp525~532, 2014

[20] 권혁신 외 2인, “Vuylsteke 구조광 패턴을 이용한 손목 착용형 웨어러블 가상 입력 장치” 대한전자공학회 학술대회, 2017.6, pp. 852-855, 2017

[21] ITU-T E161, ISO 9995-8 국제표준

[22] Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." Advances in neural information processing systems, 2014.

[23] Corpus of Contemporary American English, https://www.english-corpora.org/coca/