碩 士 學 位 論 文

인공지능 기반 스포츠 훈련 지도 보조 시스템

AI-Based Coaching Assistant System for Sports Practice

高麗大學校 大學院

電子情報工 學科

任 洗 旼

2018 年 12 月 日

吳 亨 哲 敎 授 指 導

碩 士 學 位 論 文

인공지능 기반 스포츠 훈련 지도 보조 시스템

AI-Based Coaching Assistant System for Sports Pracitce

이 論文을 工學 碩士學位 論文으로 提出함.

2018 年 12 月 日

高 麗 大 學 校 大 學 院

電 子 情 報 工 學 科

任 洗 昊 (印)

任洗旼의 工學 碩士學位論文 審査를 完了함.

2018 年 12 月 日

委員長 오 형 철 (인)

委 員 박 주 영 (인)

委 員 안 인 경 (인)

목 차

그림 목차

수식 목차

표 목차

국문 요약

인공지능 기반 스포츠 훈련 지도 보조 시스템

임 세 민
지도교수: 오 형 철
전자정보공학 전공
대학원 전자정보공학과
고 려 대 학 교

최근 인공지능기술의 발달에 따라 많은 수의 센서(sensor)를 통합한 웨어러블 디바이스(wearable device)와 기계 학습 기술을 접목하여 건강관리 응용 분야에 사용되는 인공지능 시스템이 주목 받고 있다. 이러한 시스템은 건강관리를 위해 사람의 행동을 인식하거나, 다양한 운동 자세를 인식하여 교정 및 훈련을 위한 장치로 사용될 수 있다. 그러나 이러한 시스템은 계산량이 많은 기계 학습 기술의 사용으로 인하여 감당하기 어려운 부담을 초래한다. 이러한 부담을 덜기 위해 탁구연습에 사용될 저렴한 휴대용 장치를 위한 효율적인 추론 시스템을 모색하였다.

딥 러닝(deep learning) 기술의 LSTM RNN (Long-Short Term Memory Recurrent Neural Network)을 사용하되 프루닝(pruning) 기법을 사용하여 신경망을 간소화하였고, 소규모 응용인 점을 감안하여 딥 러닝 기술의 대안이 될 수 있는 해결방안을 모색하여 코사인 유사도 기반 추론 시스템이 좋은 대안이 될 수 있음을 보였다.

운동자 5명의 5가지 탁구 자세를 추론하였고, Python 언어 기반 Tensorflow 딥 러닝 프레임워크(framework)를 사용하여 LSTM RNN 추론 시스템을 설계하였으며, 코사인 유사도 기반 추론 시스템 또한 Python 언어로 설계하였다. 96%로 LSTM RNN이 가장 높은 정확도를 나타냈고, 코사인 유사도 기반 추론 시스템 역

시 93.3%의 높은 정확도를 나타내었다. 추가로 FPGA(Field Programmable Gate Array) 기반 코사인 유사도 계산기도 설계하였으며, 베릴로그 언어 기반 Quartus Prime 18.0 환경에서 Intel Cyclone FPGA (5CSEMA5F31C6)를 타깃(target)으로 구현하였다. 이 계산기는 FPGA의 소량 부분만을 사용하며, 176MHz의 최대동작주파수를 가진다. 이러한 동작 주파수는 제공되는 행렬 데이터를 약 0.2ms로 처리할 수 있는 성능이다.

주제어 : 웨어러블 디바이스, 딥 러닝, LSTM, 코사인 유사도, 하드웨어 계산기

Abstract

LSTM Recurrent Neural Networks based Assistive System for Sports Practice

Se-Min Lim
Advised by Prof. Hyeong-Cheol Oh
Major in Electronic Information Eng.
Dept of Electronic & Information Eng.
Graduate School, Korea Univ.

Recently, artificial intelligence systems, draw attention to the field of health care. These system combine wearable devices that incorporate a large number of sensors and the machine learning technology. Such a system can be used as a device for calibrating human activity posture and practicing exercise skill by recognizing human behavior for health care or recognizing various exercise postures. However, such a system brings an unmanageable burden due to the use of machine learning techniques. To alleviate this burden, we seek an efficient inference system for low-cost portable devices to be used for table tennis practice.

In the thesis, the LSTM RNN(Long-Short Term Memory Recurrent Neural Network) of deep learning technology is adopted and simplified by using the pruning technique. In consideration of the fact that it is a small-scale application, we prove that cosine similarity based inference system can be a good alternative.

We classified five types of table tennis posture of five players, designed

LSTM RNN inference system using Python language based Tensorflow deep learning framework, and designed cosine similarity based inference system in Python language. LSTM RNN showed accuracy of 96% which is the highest accuracy of overall inference system, and the cosine similarity based inference system also showed high accuracy of 93.3%. In addition, FPGA(Field Programmable Gate Array) based cosine similarity calculator is designed and implemented with Intel Cyclone FPGA (5CSEMA5F31C6) in Quartus Prime 18.0 environment based on verilog language. The calculator uses only a small fraction of the FPGA and has a maximum operating frequency of 176 MHz. This operating frequency is a performance capable of processing the provided matrix data in about 0.2 ms.

제 1장 서 론

1.1 개요

최근 삶의 편의성과 효율성 향상을 목적으로 다양한 응용 분야에서의 인공지능 기술 활용이 주목되고 있다. 그 중 사람의 건강관리 응용 분야에 인공지능 기술이 핵심적인 역할을 하고 있다는 것이 입증되고 있으며, 이에 따라 사람의 행동 패턴을 학습하여 인식하고 추론함으로써 교정 및 훈련 역할을 수행하는 시스템의 필요성이 증가하고 있다.

전술한 시스템은 대표적으로 영상으로 행동을 인식 및 추론하는 방식과 웨어러블 디바이스(wearable device)를 통해 수집된 센서(sensor) 값을 통해 인식하여 추론하는 방식으로 나눌 수 있다. 영상기반 시스템도 높은 추론 능력을 보여주지만, 웨어러블 디바이스 기반 시스템 역시 매우 높은 추론 능력을 보이면서 세밀한 자세 변동까지 인식할 수 있는 장점을 가지고 있다.

행동 인식 및 추론 시스템은 고정된 곳에서 사용되는 시스템보다는 휴대용 장치에 사용될 시, 장소 및 시간에 구애 받지 않고 이용할 수 있다. 이에 따라 휴대용 장치에서의 시스템 구현이 필수적이다.

인공지능기술은 학습과정과 추론 과정으로 나뉘며, 전술한 시스템은 추론과정을 수행하는 시스템이다. 학습 과정은 매우 많은 계산량을 필요로 하기 때문에 휴대용 장치에서의 구현이 매우 어렵지만, 추론 과정 역시 인공지능 신경망의 부담스러운 가중치의 양 때문에 휴대용 추론 시스템 완성은 도전적인 과제이다.

1.2 연구동향 및 연구내용

제 1장 1절에서 전술하였듯이, 휴대용 장치에서 사람의 행동 및 자세 인식 시스템의 구현이 매우 중요한 연구 분야로 자리 잡았으며, 이러한 사람의 움직임에 관

한 정보를 분석하여 행동을 인식하는 행동인식기술의 적용이 활발하게 연구되고 있다[1,2,3,4,5,6,7].

[8,9]은 하나의 센서를 기반으로 하고, [10]은 여러 개의 센서를 기반으로 CNN (Convolutional Neural Network)을 이용하여 사람 행동 인식 시스템을 구현하고 있다. 그러나 CNN의 지역(local) 연결성의 제한은 신경망이 시계열 데이터의 특정 지점 사이에서 알 수 없는 지속 기간의 지연을 효과적으로 처리하지 못하게 한다. 따라서 활동의 시간적 역학을 보다 효과적으로 처리하기 위해 [3]에서 LSTM(Long Short-Term Memory)[11]을 이용한 RNN(Recurrent Neural Network)이 제안되었다. [3]에서는 LSTM RNN이 걷기나 뛰기 같은 사람 행동 인식에서 CNN이나 random forest, support vector machine 등 다른 기계 학습 기술들보다 더 나은 결과를 보임을 증명하였다.

본 논문은 최근 급성장하는 휴대용 장치에서의 행동 및 자세 인식 시스템을 설계 및 구현하되, 비용 절감 및 성능을 향상시킬 수 있는 방법에 대한 연구이다. 사람의 행동 인식 중 특정 운동 분야인 탁구를 선택하여 좀 더 세밀한 변화를 인식할 수 있는 시스템을 구현하였으며, 전술한 연구 동향에 동기를 얻고 각 운동자의 탁구 기술에 따라 자세의 지속과 시간이 다를 수 있는 점을 고려하여 딥 러닝 (deep learning) 기술 중 이미 [3]에서 시계열 데이터 배열의 효과적인 처리에 따른 높은 추론 능력이 증명된 LSTM RNN을 선택하였다. 또한, 수집한 탁구 자세 데이터처럼 소규모 응용 분야에서 최적화된 추론 성능을 이루기 위해 2 Stacked LSTM RNN 추론 시스템을 구현하였다.

[12]에서는 딥 러닝의 추론 과정만을 휴대용 장치에 구현하였지만, 비용 면에서 상당히 제한적인 모습을 보이고 있다. 구현한 2 Stacked LSTM RNN 추론 시스템 역시 방대한 계산량이 필요하기 때문에 휴대용 장치에서의 구현에 있어서 부담스러운 문제를 야기한다. 이와 같은 문제를 해결하기 위해 본 논문에서는 다양한 대처 방안들을 모색하였고, 기존의 LSTM RNN 기반 추론 시스템을 프루닝 (pruning) 기법을 통해 간소화하는 방안과 코사인 유사도를 이용한 추론 시스템을 또 다른 방안으로 선택하였다.

LSTM RNN 기반 추론 시스템을 간소화하면서 계산량을 크게 줄였지만, 간소화하면서 발생한 희소 행렬을 효과적으로 처리해야하는 새로운 문제에 직면하게 된다. 또한, 코사인 유사도 기반 추론 시스템이 LSTM RNN에 비해 정확도 면에서 많은 차이가 없고 높은 추론 성능을 보였다. 이에 따라 코사인 유사도 기반 추론 시스템을 최종적인 저비용 방안으로 채택하였고, 시스템의 비용 면에서 가장 큰 부분을 차지하는 유사도 계산 부분을 휴대용 장치인 FPGA(Field Programmable Gate Array)에 구현하였으며, 전체 FPGA에서 매우 작은 부분만을 사용하고 있는 모습을 보여 높은 가능성을 가진 저비용 방안이라는 것을 보였다.

1.3 본 논문의 구성

본 논문의 구성은 다음과 같다. 제 2장에서 LSTM RNN과 코사인 유사도 및 프루닝에 대한 이론적 지식을 소개하고, 제 3장에서는 탁구 자세 측정을 위해 사용된 데이터 수집 장치에 대해 설명한다. 제 4장에서는 제 2장에서 소개한 LSTM RNN의 이론적 지식을 기반으로 설계한 2 Stacked LSTM RNN에 대해 설명하고 제 5장에서는 설계한 프루닝 기법과 프루닝 기반 신경망 재설계에 대해 설명한다. 제 6장에서는 제 2장에서 소개한 코사인 유사도의 이론적 지식을 기반으로 설계한 추론 시스템을 설명하고, FPGA 기반 코사인 유사도 계산기에 대해 설명한다. 제 7장에서 설계한 추론 시스템들의 성능을 평가하고, 제 8장에서 결론을 내리며 본 논문을 마친다.

제 2장 배경

본 장에서는 RNN의 기본 지식과 LSTM RNN에 대해 설명하고 신경망을 간소화하는 방법 중 본 논문에서 채택한 프루닝 기법을 설명한다. 또한, 코사인 유사도의 기본 지식을 소개하고 설명하며 이 장을 마친다.

2.1 RNN(Recurrent Neural Network)

RNN은 숨겨진 마디(hidden node)가 방향을 가진 엣지(edge)로 연결되어 순환구조를 이루는 인공신경망의 한 종류이다. RNN의 순환구조는 동일한 계산을 한 배열의 모든 요소마다 적용하고, 출력 결과를 이전의 계산 결과에 영향을 받게 한다. 이러한 신경망은 음성, 문자 등 순차적으로 등장하는 데이터 처리에 매우 높은 추론 능력을 보인다. 또한, 시계열 데이터 길이에 관계없이 데이터를 입력 받고 출력할 수 있는 신경망 구조이기 때문에, 필요에 따라 다양하고 유연하게 구조를 만들 수 있다는 점이 가장 큰 장점이다.

[식 2-1]은 기본적인 RNN의 수식을 나타낸다.

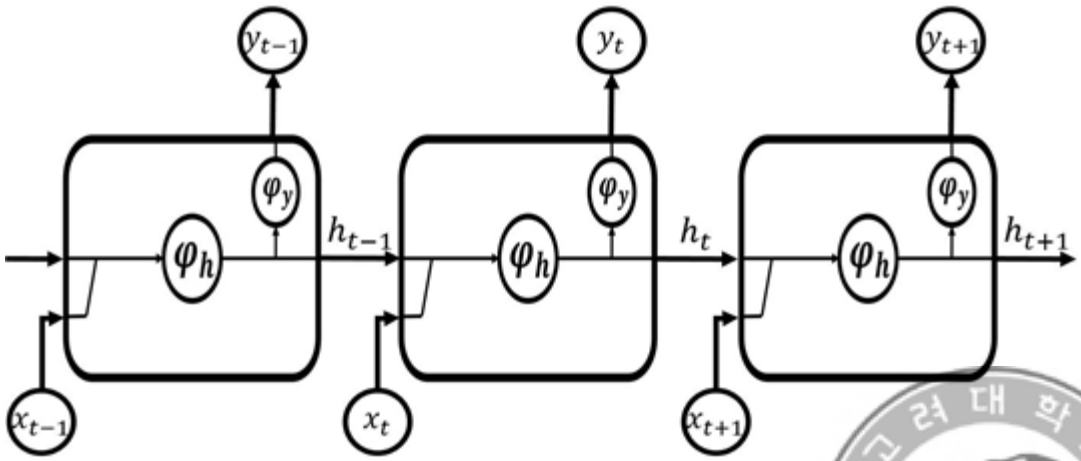
$$h_t^l = A(W_{xh}^l x_t^l + W_{hh}^l h_{t-1}^l + b_h^l)$$
$$y_t = A(W_{hy} h_t^l + b_y)$$

[식 2-1] RNN의 정의

위의 수식에서 t는 현재 시간을 의미하며, h_t, x_t, y_t는 각각 현재 숨겨진 상태(hidden state)와 현재 입력(input), 현재 출력(output)을 의미한다. 숨겨진 상태는 신경망의 메모리와 같은 역할을 수행하며, 과거 시간에서 계산된 정보를 담고 유지하여 현재 및 미래 시간의 숨겨진 상태와 최종 추론 결과 도출에 사용된다. l은 l개의 층(layer)로 구성되어 있는 숨겨진 층중에서 해당하는 층을 나타낸다. l의 수가

클수록 숨겨진 층의 수가 많다는 뜻이며 신경망의 크기가 커짐을 의미한다. 대규모 데이터 사용 시 숨겨진 층의 수를 크게 구성하면 추론 성능 개선에 도움이 되지만, 소규모 데이터 사용 시 과도하게 숨겨진 층을 늘리게 되면 과적합(overfitting)이 발생하여 추론 성능이 오히려 더 악화될 수 있기 때문에 숨겨진 층의 수는 사용되는 데이터에 맞게 구성해야한다. A는 활성화 함수를 의미하며 비선형 함수가 사용되는데, 이는 선형 함수가 사용될 경우 함수가 숨겨진 층의 수에 맞게 여러 번 곱셈 연산이 수행되어도 하나의 함수로 나타낼 수 있으므로 층이 쌓이는 효과를 얻지 못하기 때문이다. 활성화 함수로는 ReLu(Rectified Linear Unit) 함수와 tanh 함수 및 시그모이드(sigmoid) 함수가 자주 사용된다. W는 신경망 내 연결들의 가중치로 구성된 매개변수 행렬을 의미한다. 예를 들어 W_{xh}는 입력 층과 숨겨진 층 사이의 가중치 행렬을 의미한다.

[그림 2-1]은 기본적인 RNN의 구조를 나타낸다. 과거 상태 $t-1$에서 현재 t 및 미래 상태 $t+1$까지 연결되는 모습을 볼 수 있으며, φ_h, φ_y는 각각 숨겨진 상태와 출력에서 사용되는 활성화 함수를 의미한다.



[그림 2-1] RNN의 구조

2.2 LSTM(Long Short-Term Memory) RNN

제 2장 1절에서 설명한 RNN는 두 가지의 연속적인 과정을 거쳐 학습이 진행된다. 먼저 순전파 과정(forward propagation)을 통해 입력이 들어가면 가중치와 해당 상태의 숨겨진 상태 및 출력 값 등 신경망 내의 모든 매개변수가 생성된다. 이후 순전파 과정에서 발생한 오차를 신경망 전체의 매개변수에 미분연산과 곱셈 및 덧셈 연산이 포함된 함수 계산을 반복하는 역전파 과정(back propagation)을 통해 매개변수들을 갱신하게 된다.

그러나 RNN는 관련 정보와 그 정보를 사용하는 지점 사이 거리가 멀어질 경우, 연속적인 미분 연산이 포함된 역전파 과정이 진행될 때 기울기(gradient)가 점차 줄어 이전 정보에 대한 기억 능력이 크게 감소함에 따라 과거의 중요 정보를 잊게 됨으로써 학습 능력이 크게 저하되는 단점이 있다. 이러한 문제를 극복하기 위해 RNN에 LSTM을 추가하여 RNN의 숨겨진 상태에 셀(cell) 상태를 추가하고 각 게이트(gate) 함수를 더하여 이전 정보와 현재 정보에 대한 기억을 조절하여 학습한다. [식 2-2]는 LSTM를 구성하고 있는 게이트들과 셀 및 숨겨진 상태를 나타낸다.

$$
\begin{aligned}
f_t^l &= \varphi_f(W_{xf}^l, x_t^l, W_{hf}^l, h_{t-1}^l, b_f^l) \\
i_t^l &= \varphi_i(W_{xi}^l, x_t^l, W_{hi}^l, h_{t-1}^l, b_i^l) \\
o_t^l &= \varphi_o(W_{xo}^l, x_t^l, W_{ho}^l, h_{t-1}^l, b_o^l) \\
g_t^l &= \varphi_g(W_{xg}^l, x_t^l, W_{hg}^l, h_{t-1}^l, b_g^l) \\
c_t^l &= f_t^l \otimes c_{t-1}^l + g_t^l \otimes i_t^l \\
h_t^l &= o_t^l \otimes A(c_t^l)
\end{aligned}
$$

[식 2-2] LSTM RNN 셀 내의 각 게이트 함수와 셀 상태 및 숨겨진 상태

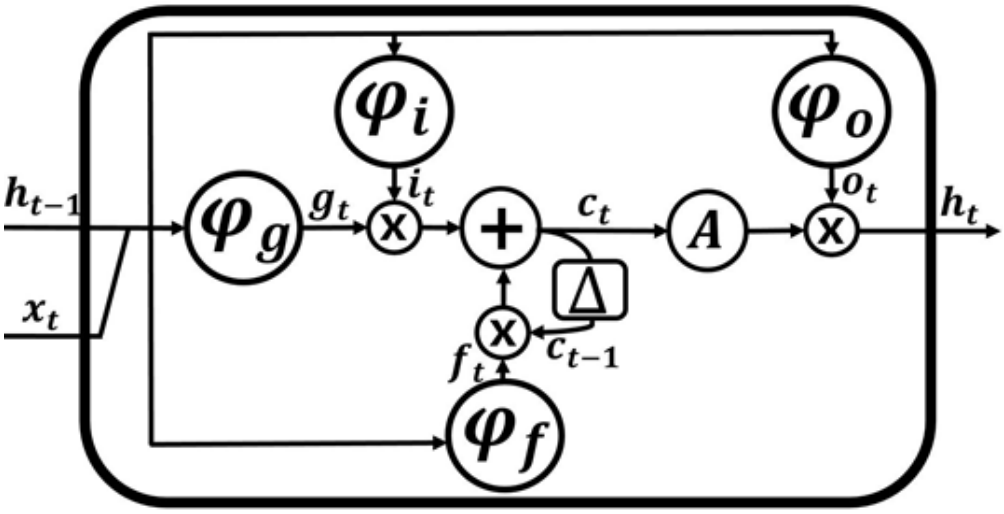
[식 2-1]의 표기법과 동일하며, ⊗는 요소 별 곱셈 연산을 의미한다. $f_t^l, i_t^l, o_t^l, g_t^l$는 각각 망각 게이트(forget gate), 입력 게이트(input gate), 출력 게이트(output gate), 입력 조절 게이트(input modulation gate)를 의미하며, 게이트 내부에서 이루어지는 계산은 [식 2-3]과 같이 정의된다.

$$\varphi_k(W_{xk}, x, W_{hk}, h, b_k) = A(W_{xk}x + W_{hk}h + b_k)$$

[식 2-3] 각 게이트 내의 계산

[식 2-3]에서 $k = f, i, o, g$이며, A는 활성화 함수를 의미한다.

[식 2-2]에서 망각 게이트는 이전 상태의 정보를 얼마나 보존할지를 결정하는 역할을 수행한다. 즉, 추론에 큰 영향을 미치지 않는 과거 정보를 제거하는 게이트이며, 예를 들어 활성화 함수로 시그모이드 함수를 사용한다고 가정하면, 출력 범위가 0부터 1 사이이기 때문에 게이트가 0에 가까운 값을 출력할수록 제거되는 이전 상태의 정보가 많음을 뜻하고, 1에 가까울수록 많은 양의 이전 정보를 보존함을 뜻한다. 입력 게이트 및 입력 조절 게이트는 함께 구성되어 현재 셀 상태를 갱신하는데 사용되는데, 기억해야하는 현재 정보의 양과 방향을 조절하며, 추론 시 필수적으로 기억되어야할 현재 정보를 보존하고 큰 영향을 미치지 않는 정보를 제거하는 역할을 수행한다. 최종적으로 현재 셀 상태는 내부 회귀를 통한 이전 셀 상태와 망각과 입력 및 입력 조절 게이트에 의해 갱신되며, 현재 숨겨진 상태는 활성화 함수를 거친 셀 상태와 출력 게이트에 의해 갱신된다.



[그림 2-2] LSTM RNN 셀의 구조

[그림 2-2]는 LSTM RNN 셀의 구조를 나타낸다. △는 이전 셀 상태를 저장하기 위한 메모리 요소를 의미하며, 나머지 요소들은 [식 2-2]와 [식 2-3]의 표기법과 동일하다. 이전 숨겨진 상태와 현재 입력을 통해 셀 내부에서 [식 2-2]와 [식 2-3]을 이용하여 설명한 계산 방식들이 적용되는 순차적인 과정들을 통해 최종적으로 현재 숨겨진 상태가 출력되는 것을 확인할 수 있다.

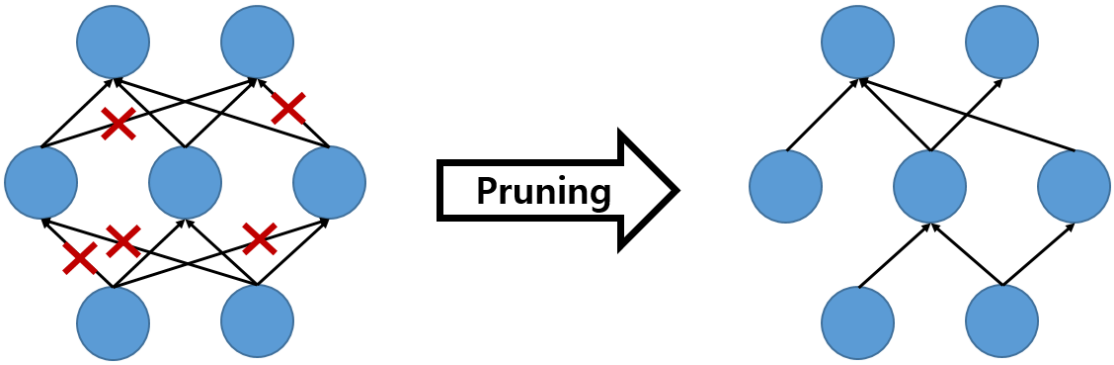
LSTM RNN은 상태들을 연결하는 경로를 설정함에 따라 방향성을 설정할 수 있는데, 대표적으로 단방향(unidirectional) LSTM RNN과 양방향(bidirectional) LSTM RNN[13,14]이 있다. 단방향 LSTM RNN은 가장 기본적인 신경망 구조이며 전 순서 및 현재 순서의 숨겨진 상태가 후 순서의 상태를 갱신하는데 사용되는 형태이다. 그러나 대부분의 시계열 데이터는 하나의 특정 배열 전체가 추론 시 전부 참조되어야한다. 즉, 전 순서 및 현재 순서 상태만을 참조하여 후 순서 상태를 갱신하는 형태는 국한적인 추론 성능을 야기할 수 있다. 따라서 기존에 전방 경로(forward path)만을 가지고 있던 단방향 LSTM RNN에 현재 순서 및 전 순서 상태를 갱신할 때 후 순서 상태를 참조하게 하는 후방 경로(backward path)를 추가하여 개선된 추론 성능을 가지게 하며, 이를 양방향 LSTM RNN이라 한다.

2.3 프루닝(Pruning)

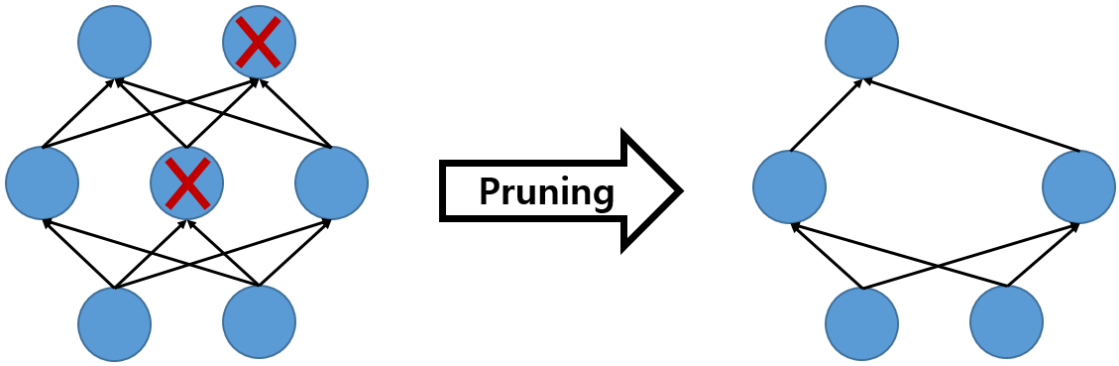
딥 러닝 기술은 학습 및 추론 과정에서 매우 많은 계산을 수행한다. LSTM RNN 역시 기존의 RNN에 LSTM 셀이 추가되면서 방대한 가중치들이 추가되어 많은 계산이 수행되어야한다. 하지만 추가된 많은 양의 가중치들이 추론에 모두 핵심적인 역할을 수행하지 않는다. 가중치는 추론 시 필수적으로 존재해야하는 정보에 큰 값을 곱하여 정보를 가중시키는 역할을 수행한다. 따라서 가중치가 작은 값은 추론에 큰 영향을 미치지 않는 정보라고 판단할 수 있으며, 작은 값을 나타내는 가중치까지 학습 및 추론에 이용하는 것은 불필요한 과정이다. 또한, 휴대용 장치에서의 추론 시스템 구현에서 신경망 크기 감소는 매우 필수적인 과제이다. 이에 따라 최근 추론 성능에 영향을 미치지 않으며 신경망을 더 작게 구성하고 학습 및 추론 속도를 개선하는 연구가 활발히 진행되고 있다. 본 논문은 이러한 연구들 중 LSTM RNN의 모든 가중치가 추론 시 필수적인 영향을 미치지 않는다는 것을 감안하고 간단하게 구현하여 신경망을 감소시킬 수 있는 방안인 프루닝 기법[15]을 선택하였다.

프루닝 기법은 대표적으로 가중치 프루닝와 신경 프루닝 기법으로 나눌 수 있다. 가중치 프루닝 기법은 신경들을 연결하고 있는 가중치들을 제거하는 기법이며, 신경 프루닝 기법은 신경들을 직접 제거하며 연결되어 있는 가중치 또한 모두 제거하는 기법이다. 전술한 두 기법 모두 임계값을 설정하여 이하의 가중치 및 신경을 제거하는 방식으로 진행된다. 이러한 프루닝 기법은 추론 성능에 큰 영향을 미치지 않으며 신경망의 크기를 감소하고 추론 속도를 개선하는데 도움을 줄 뿐만 아니라 데이터와 신경망의 크기 사이에서 발생할 수 있는 과적합 방지에 사용될 수 있다.

[그림 2-3]은 가중치 프루닝 기법과 신경 프루닝 기법을 도식화하여 나타낸 것이다. 임계값 이하의 값을 갖는 가중치와 신경이 선택적으로 제거되어 신경망이 간소화되는 것을 확인할 수 있다.



(a) 가중치 프루닝



(b) 신경 프루닝

[그림 2-3] 프루닝 기법

2.4 코사인 유사도 (Cosine Similarity)

코사인 유사도[16]는 내적공간에 존재하는 두 벡터 간의 유사한 정도를 나타내며, 벡터들이 이루는 각도의 코사인 값을 이용한다. [식 2-4]는 코사인 유사도의 정의를 나타내는 수식이며, k와 k_G는 각각 벡터를 의미하고, $\bullet$는 벡터 간 내적을 의미한다. 이러한 유사도 값은 벡터의 크기에 영향을 받지 않으며, 오직 두 벡터 간 방향의 유사도를 판단하는 목적으로만 사용된다. 1부터 -1까지의 값을 가지며, 1인 경우는 두 벡터의 방향이 완전히 같은 경우를 의미하고, -1인 경우는 180°로 방향이 완전히 다른 경우를 의미한다. 코사인 유사도는 다차원에 적용이 가능하여 문서 비교와 시계열 데이터의 배열 비교 및 이미지 비교 등 다양한 응용 분야에서 자주 이용되는 기법이다.

$$\text{Cos\_Sim}(k,k_G) = \frac{k \bullet k_G}{|k||k_G|}$$

[식 2-4] 코사인 유사도

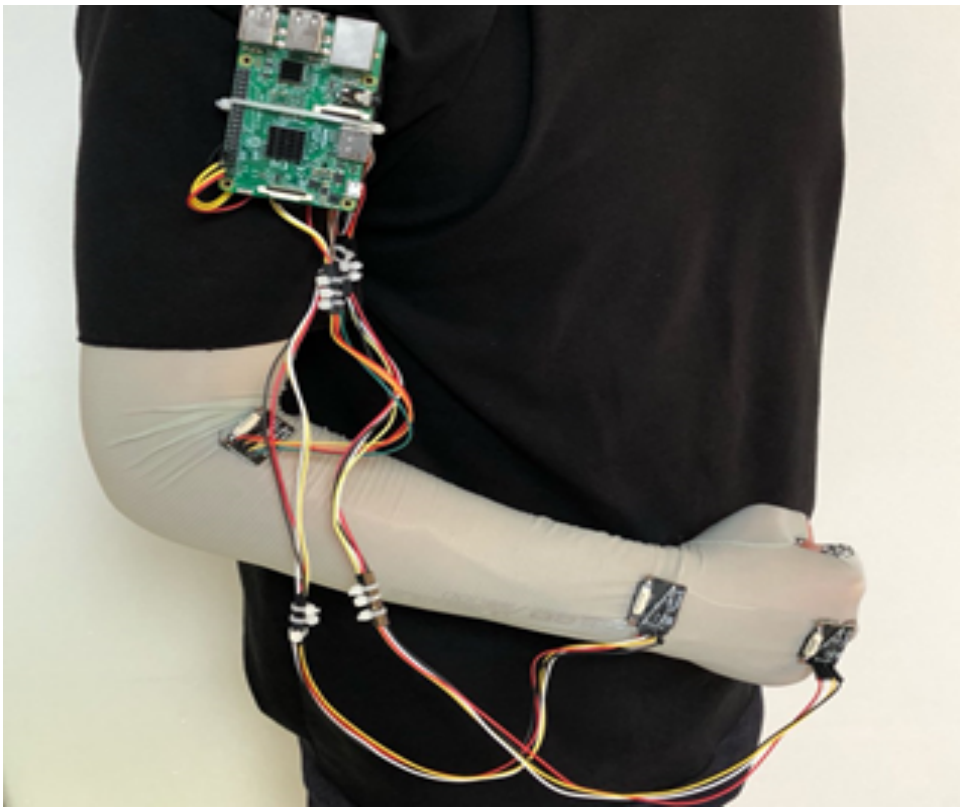
제 3장 데이터 수집

본 장에서는 추론 시스템의 기반이 되는 데이터를 수집하기 위해 설계한 데이터 수집 장치를 설명하고, 장치를 통해 수집한 데이터의 구성에 대해 설명한다. 본 장의 내용의 대부분은 공동연구의 결과인 참고문헌[4,5,6,7]에서 가져온 것이다.

3.1 데이터 수집 장치

탁구 자세를 측정하고 데이터를 수집하여 추론 시스템을 설계 및 구현하기 위해 데이터 수집 장치를 제작하였다. 장치는 임베디드(embedded) 컴퓨터인 라즈베리파이 보드 3 모델 B에 IMU(Inertial Measurement Unit) 센서 3개를 접속하여 구현하였다. IMU 센서는 MPU-9150을 사용하였고, 3축 가속도(accelerometer)와 3축 자이로(gyro) 및 3축 지자기(geomagnetic) 센서로 구성되어있다. 라즈베리파이 보드와 센서 사이의 통신은 I2C 통신을 이용하였다. 그러나 라즈베리파이 보드는 최대 2개의 디바이스와 I2C 통신만을 허용하고 있어, 3개의 센서를 접속하기 위해 I2C 통신 접속 멀티플렉서(multiplexer)인 TCA9548A를 추가로 사용하였다. 전술한 멀티플렉서는 최대 8개까지 I2C 연결 디바이스를 제어할 수 있으며, 라즈베리파이 보드와 연결되어 있는 각각의 센서에 할당되어 있는 주소로 접근하여 0.001초 단위로 3개의 센서 연결을 접속 및 해제하면서 센서가 모두 연결되어 측정되는 것과 동일한 효과를 나타낸다.

라즈베리파이 환경에서 측정한 데이터를 연속적으로 저장하기 위해 데이터베이스 구축하였으며, 측정 및 환경 구축의 편의성을 위해 데이터베이스를 라즈베리파이 환경에서 서버를 생성하여 저장함으로써, 어느 PC에서도 서버에 접근하여 저장된 데이터베이스를 다운로드할 수 있도록 구현하였다. [그림 3-1]는 본 논문에서 설계 및 구현한 데이터 수집 장치를 보인다.



[그림 3-1] 데이터 수집 장치

3.2 데이터 구성

5명의 운동자를 구성하였으며 각 운동자마다 탁구의 기본적인 5가지 자세를 수행하여 데이터를 측정하였다. [그림 3-2]는 탁구의 기본 5가지 자세인 포핸드 스트로크(forehand stroke), 백핸드 드라이브(backhand drive), 백핸드 쇼트(backhand short), 포핸드 커트(forehand cut), 포핸드 드라이브(forehand drive)를 수행한 모습을 나타낸다.

[그림 3-2] 탁구의 5가지 동작: 포핸드 스트로크, 백핸드 드라이브, 백핸드 쇼트, 포핸드 커트, 포핸드 드라이브

센서의 기능 중 지자기 센서는 자성이 존재하는 환경에서는 측정에 매우 제한적인 양상을 나타내는 이유로 가속도 및 자이로 센서 값만을 수집하였다. 센서 값 수집은 5Hz 단위로 측정하여 데이터베이스에 저장하였다. 각 운동자 및 자세마다 데이터의 배열이 다른데, 가장 긴 배열인 22개 기준으로 이하의 수를 나타내는 배열에는 zero-padding 하였다. 수집된 데이터 중 70%는 학습용, 30%는 테스트용으로 구성하였다. [식 3-1]은 학습용 및 테스트용 데이터의 내부 구성을 나타낸다.

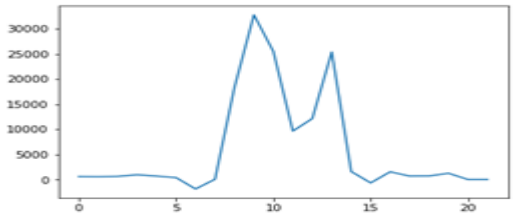
$$5(\text{운동자}) \times 5(\text{탁구자세}) \times 7(\text{반복수행}) \times 3(\text{축}) \times 3(\text{센서}) \times 2(\text{센서모듈}) = 3150$$
$$5(\text{운동자}) \times 5(\text{탁구자세}) \times 3(\text{반복수행}) \times 3(\text{축}) \times 3(\text{센서}) \times 2(\text{센서모듈}) = 1350$$

[식 3-1] 학습 및 테스트용 데이터 구성

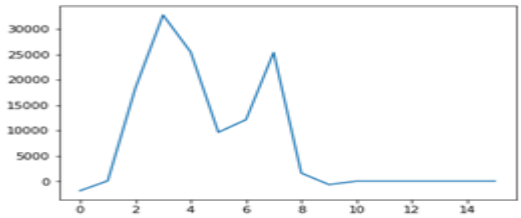
3.3 데이터 전처리(Preprocessing)

추론 시스템의 정확도 향상을 목적으로 데이터 수집 장치로 수집한 데이터를 전처리 과정을 거쳐 새로운 데이터 세트를 획득하였다. 데이터를 측정 및 수집 시 실제로 동작을 수행하여 얻은 데이터 이외에도 동작 수행 전이나 수행 후에 추론에 쓸모없는 데이터가 섞이게 된다. 획득한 데이터를 분석 후 추론에 필수적인 실제 동작이 데이터 배열의 극소 값에서 시작하여 극소 값을 끝나는 것을 확인하였다. 이에 따라 첫 극소 값이 발견되기 전과 마지막 극소 값이 발견된 후의 데이터를 제거하여 원본 데이터를 전처리하였다. 이러한 전처리 방법은 구현 과정이 매우 간단하며, 데이터의 순차적인 배열 형태나 값을 직접적으로 변환하는 것이 아니기 때문에 원본 데이터의 형태를 거의 그대로 보존한다.

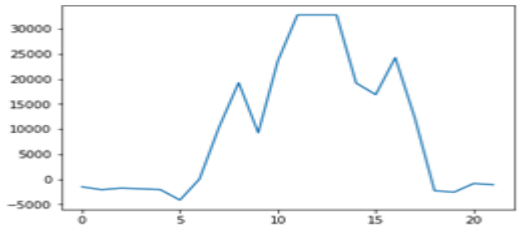
[그림 3-3]는 수집한 가속도 센서 값의 원본 데이터와 전처리한 데이터 배열을 그래프로 나타낸 것이다. 전술한 방식으로 데이터를 전처리하게 되면, 데이터 배열의 수가 감소하는데 전처리 데이터 배열 중 가장 긴 배열인 16개를 기준으로 이하의 배열 수를 가지는 전처리 데이터는 zero-padding하였다.



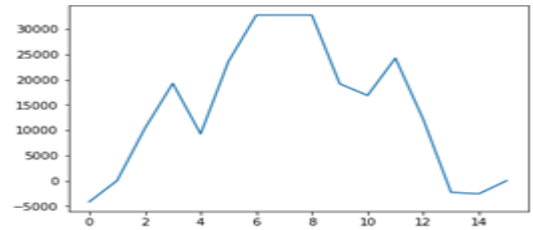
(a) 자세 1(Original)



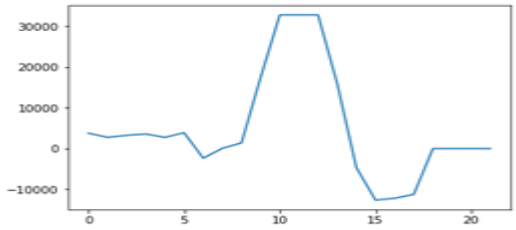
(b) 자세 1(Preprocessed)



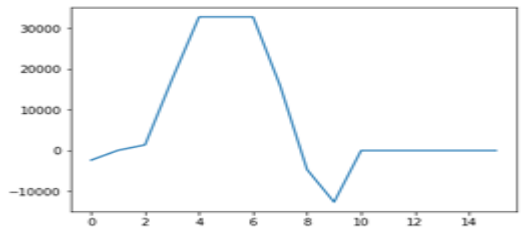
(c) 자세 2(Original)



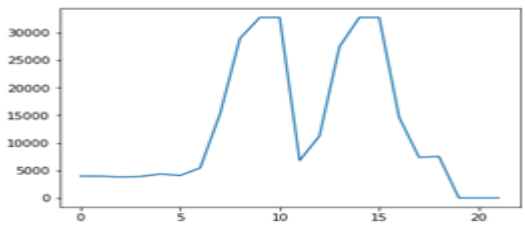
(d) 자세 2(Preprocessed)



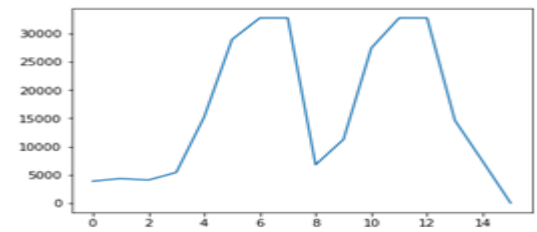
(e) 자세 3(Original)



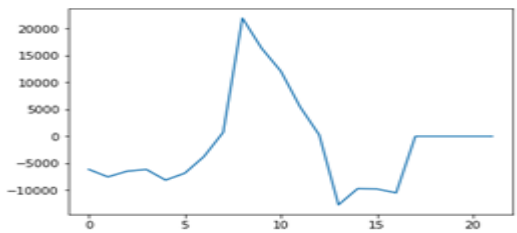
(f) 자세 3(Preprocessed)



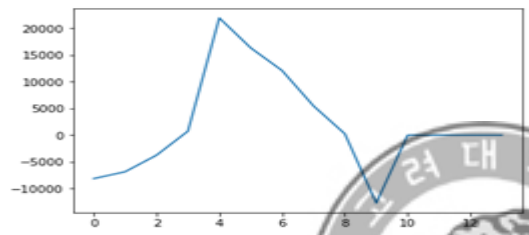
(g) 자세 4(Original)



(h) 자세 4(Preprocessed)



(i) 자세 5(Original)



(j) 자세 5(Preprocessed)

[그림 3-3] 데이터 전처리

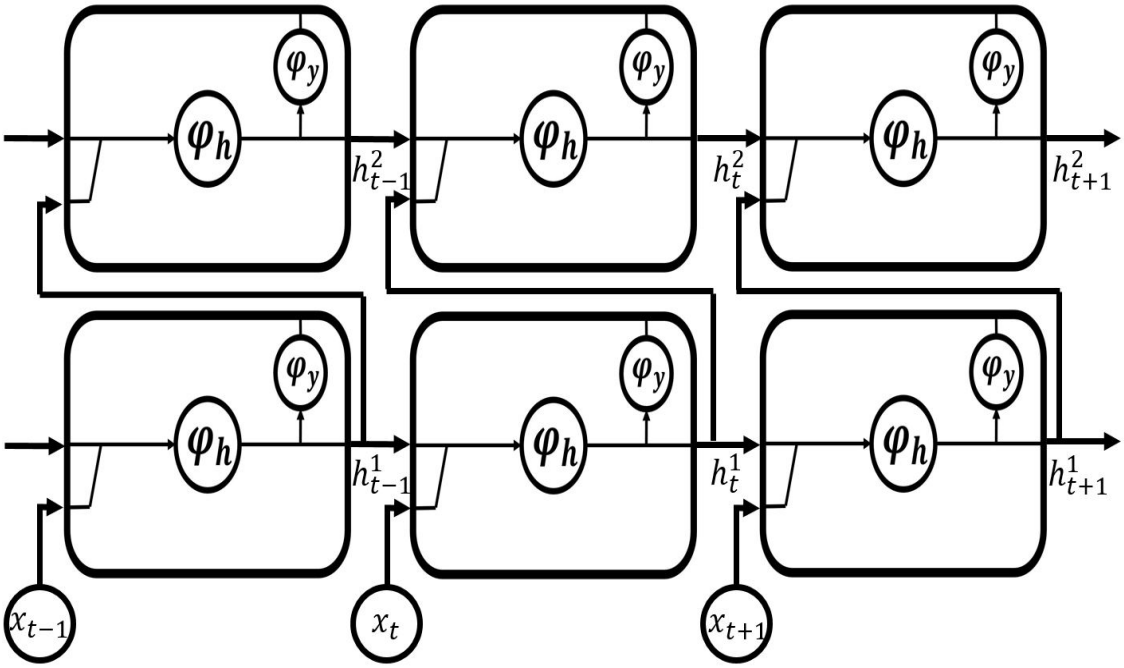
제 4장 2 Stacked LSTM RNN

본 장에서는 제 2장의 내용을 기반으로 하여, 수집한 데이터에 최적의 능력을 갖도록 설계한 단방향 2 Stacked RNN과 단방향 및 양방향 2 Stacked LSTM RNN을 설명한다. 본 장의 내용의 대부분은 공동연구의 결과인 참고문헌[4,6]에서 가져온 것이다.

4.1 단방향(Unidirectional) 2 Stacked RNN

제 2장 1절의 내용을 기반으로 숨겨진 층을 2개로 구성하여 단방향 2 Stacked RNN을 구현하였다. 숨겨진 층을 1개로 구현하였을 시 너무 낮은 추론 능력을 보였고, 3개로 구현하였을 시 데이터의 양에 비해 신경망의 크기가 너무 커서 과적합 현상이 심하게 발생하는 것을 확인하여, 최종적으로 2개의 숨겨진 층으로 이루어진 신경망을 구현하였으며 본 논문에서 수집한 데이터에 매우 적합한 모습을 보였다. 단방향 2 Stacked RNN는 본 논문에서 수집한 데이터에 이외에도 다른 소규모 응용 분야에서도 충분히 활용될 수 있다.

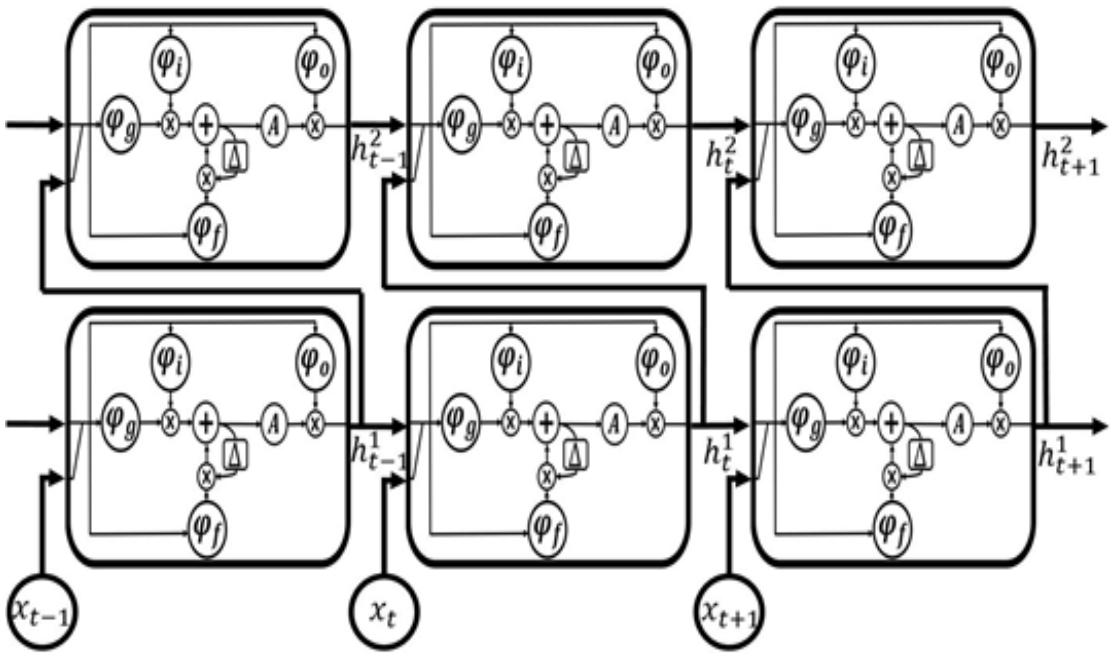
[그림 4-1]은 설계한 2 Stacked RNN의 구조를 나타낸다. 표기법은 [그림 2-1]과 동일하며, h_t^1, h_t^2는 각각 첫 번째 숨겨진 층의 현재 숨겨진 상태와 두 번째 숨겨진 층의 현재 숨겨진 상태를 의미한다. 활성화 함수로는 ReLu(Rectified Linear unit) 함수를 사용하였다. 첫 번째 숨겨 층의 각 상태가 두 번째 숨겨진 층의 해당 상태의 입력으로 들어가는 모습을 확인할 수 있다. 단방향으로만 구성한 신경망이기 때문에 전방 경로만을 수행하는 모습 또한 확인할 수 있다. 상태는 수집한 데이터에서 한 자세를 나타내는 배열의 수인 22개 혹은 16개로 구성하였다.



[그림 4-1] 단방향 2 Stacked RNN의 구조

4.2 단방향 2 Stacked LSTM RNN

제 2장 2절의 내용을 기반으로 제 4장 1절에서 설명한 2 Stacked RNN 환경을 동일하게 적용하여 단방향 2 Stacked LSTM RNN를 구현하였다. RNN 셀을 LSTM RNN 셀로 변경하였으며, 숨겨진 층 및 다른 조건들은 동일하다. [그림 4-2]는 단뱡향 2 Stacked LSTM RNN의 구조를 나타내며, [그림 4-1]과 비교했을 시, 셀이 [그림 2-2]의 LSTM 셀로 변경되고 나머지 조건은 동일한 것을 확인할 수 있다.

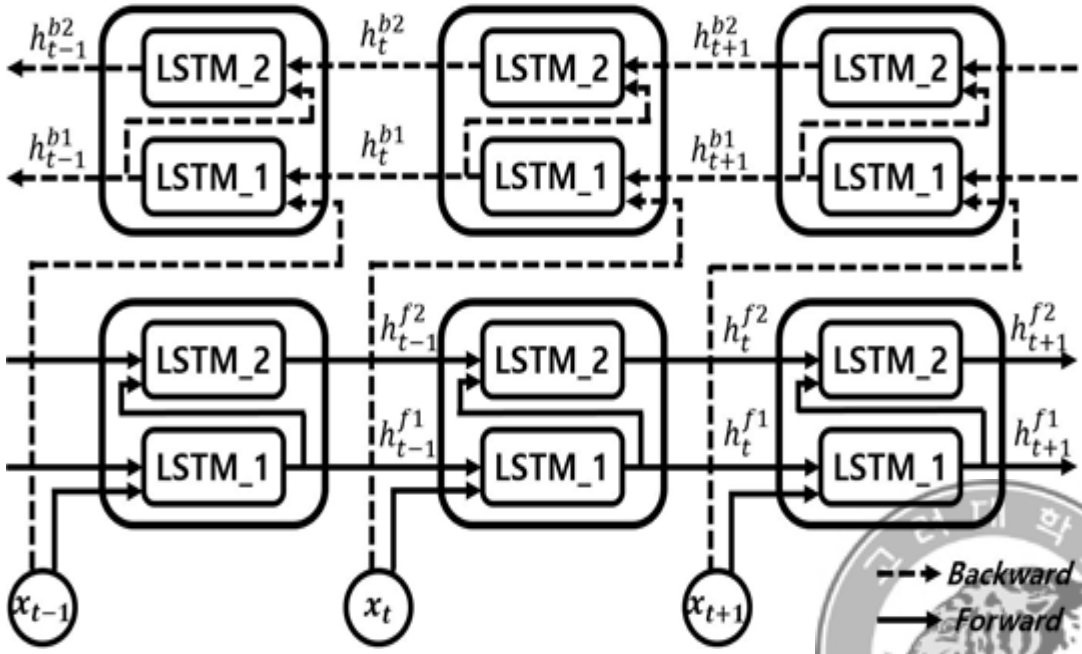


[그림 4-2] 단방향 2 Stacked LSTM RNN의 구조

4.3 양방향(Bidirectional) 2 Stacked LSTM RNN

제 2장 2절에서 설명한 바와 같이 시계열 형태의 데이터를 사용할 때, 현재 순서 데이터가 전 순서 데이터뿐만 아니라 후 순서 데이터에 영향을 받을 수 있으므로, 후방 경로를 추가함으로써 전방 경로만을 가지는 단방향 LSTM RNN보다 높은 추론 성능을 기대할 수 있다. 따라서 단방향 2 Stacked LSTM RNN에서 후방 경로를 추가하여 양방향 2 Stacked LSTM RNN을 설계하였다.

[그림 4-3]은 양방향 2 Stacked LSTM RNN의 구조를 나타내며, h_t^{f1}, h_t^{b1}는 각각 전방 경로 환경에서 첫 번째 숨겨진 층의 현재 숨겨진 상태와 후방 경로 환경에서 첫 번째 숨겨진 층의 현재 숨겨진 상태를 의미한다. 또한, 실선은 전방 경로를 나타내고, 점선은 후방 경로를 나타낸다. 추가된 후방 경로를 통해 현재 순서 데이터 x_t와 후 순서 상태인 $h_{t+1}^{b1}, h_{t+1}^{b2}$을 통해 현재 순서 상태인 h_t^{b1}, h_t^{b2}가 출력되는 모습을 확인할 수 있다.



[그림 4-3] 양방향 2 Stacked LSTM RNN의 구조

제 5장 임계값 설정 기반 신경망 프루닝

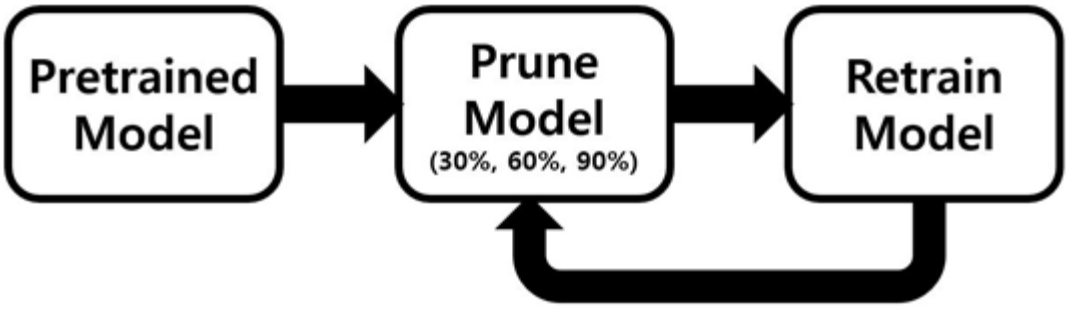
본 장에서는 제 2장 3절의 내용을 기반으로 설계한 신경망 프루닝 기법을 설명하고, 프루닝 기법 사용 후 신경망 재설계에 대해 설명하며 이 장을 마친다. 본 장의 내용의 대부분은 공동연구의 결과인 참고문헌[4,6]에서 가져온 것이다.

5.1 신경망 가중치 프루닝

제 2장 3절에서 두 가지 신경망 프루닝 기법을 소개하였다. 그 중 임계값 설정 기반 신경 프루닝 기법은 한 특정 신경에 연결되어 있는 가중치들 모두가 설정한 임계값 이하의 값을 가지면 해당 신경과 연결되어 있는 가중치를 모두 제거하는 기법으로써 가중치 프루닝 기법보다 신경망 축소 면에 있어서 효율적인 모습을 나타낸다. 하지만 특정 신경에 연결되어 있는 모든 가중치에 적용해야하는 조건 설정이 가중치 프루닝 기법보다 복잡하며, 조건에 부합하여 제거된 신경과 해당 신경의 가중치들 중 추론에 필수적인 가중치가 존재했을 수 있고, 이러한 가중치가 제거되면 추론 성능을 감소시킬 가능성이 있다. 또한, 소규모 응용 분야에서 활용되고 많은 신경들이 구성되어 있지 않은 비교적 작은 신경망에서 특정 신경 및 해당 가중치들이 모두 제거되는 것은 추론에 위험한 요소로 작용할 가능성이 있다. 따라서 본 논문에서는 조건 설정이 단조롭고, 특정 신경과 해당 전체 가중치가 제거되지 않아 추론 성능에 영향을 거의 미치지 않으면서 신경망의 크기 감소 및 추론 속도 개선 문제를 해결하는 임계값 설정 기반 가중치 프루닝 기법을 사용하였다.

5.2 프루닝 기반 신경망 재설계

[그림 5-1]은 임계값 설정 기반 가중치 프루닝 기법을 사용하여 신경망을 재설계하는 과정을 파이프라인 형식으로 나타낸 그림이다. 먼저 프루닝 기법을 사용하지 않고 LSTM RNN을 학습한 뒤 신경망 모델(pretrained model)을 생성한다. 다음으로 임계값을 설정하여 신경망 내 가중치를 지정한 양만큼 제거한다. 이 후 제거되고 남은 가중치만으로 다시 학습하여 최종 신경망 모델(retrain model)을 생성한다. 본 논문에서는 설계한 신경망의 추론 성능과 프루닝 기법 사이의 균형(trade-off)을 관측하고자 가중치의 30%와 60% 및 90%를 제거 후 각 단계의 신경망 성능을 검증하였다.



[그림 5-1] 프루닝 기반 신경망 재설계 과정

제 6장 코사인 유사도 기반 시스템

본 장에서는 제 2장 4절의 코사인 유사도에 대한 내용을 기반으로 하여 설계한 추론 시스템을 설명하고, 휴대용 장치에 구현한 코사인 유사도 계산기를 설명한다. 본 장의 내용의 대부분은 공동연구의 결과인 참고문헌[5,7]에서 가져온 것이다.

6.1 코사인 유사도 기반 추론 시스템

테스트용으로 선택된 30%의 데이터와 각 타법에 속하는 나머지 데이터와의 코사인 유사도를 계산하여 가장 유사도가 높은 동작으로 추론하였다. [그림 6-1]은 추론 정확도 계산 알고리즘을 보인 것이다. 크기가 18×22인 윈도우(window)들로 구성하고, 학습 데이터의 윈도우와 테스트 데이터의 윈도우를 비교하여 가장 높은 유사도 값을 가지는 특정 사람의 특정 타법을 결과로 출력하게 하여 정확도를 계산하였다. 윈도우 구성에서 18은 센서 3개 각각의 방향에서 가속도 및 자이로 데이터로 이루어진 총 특징(feature) 수를 의미하며, 22는 한 동작을 표현하는 데이터 배열의 길이를 나타낸다. 전처리 데이터 사용 시 22를 16으로 변환하여 윈도우를 구성한다.

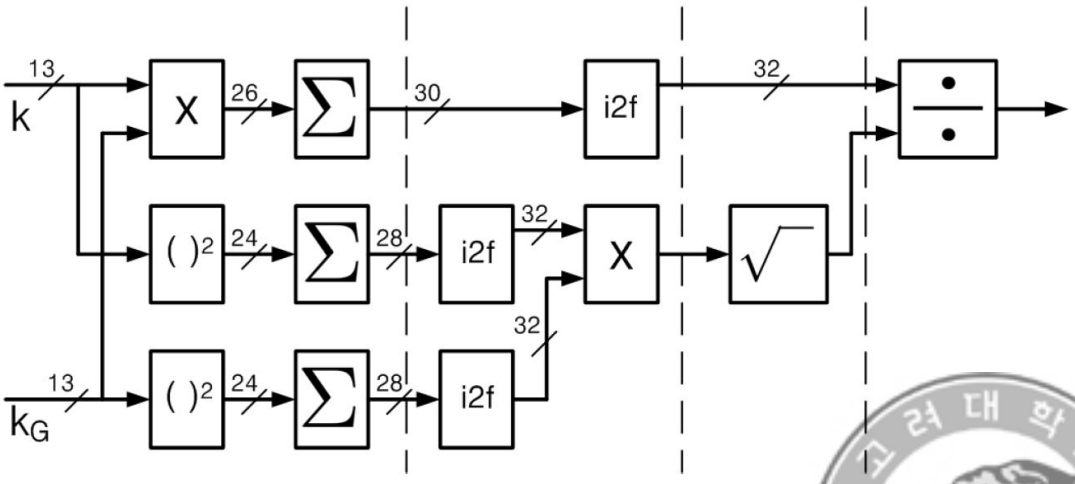
$$
\begin{aligned}
&For\ i\ in\ 5\ people's\ 5\ posture(25)\\
&\quad For\ x\ in\ total\ feature(18)\\
&\quad\quad For\ y\ in\ time\ series(22)\\
&\quad\quad\quad similarity[x] =\\
&\quad\quad\quad\quad calculate\_similarity(test[x,y], train[x,y])\\
&\quad temp_{result[i]} = argmax(similarity[x])\\
&\\
&result = argmax(temp_{result}[i])
\end{aligned}
$$

[그림 6-1] 코사인 유사도 기반 추론 정확도 계산 알고리즘

6.2 FPGA 기반 코사인 유사도 계산기

[그림 6-2]는 설계한 코사인 유사도 계산기의 블록도(block diagram)를 나타낸다. ∑는 축적기(accumulator)를 나타내며, $i2f$는 정수 부동소수점 변환기(integer-to-floating-point converter)를 나타낸다. 수집한 데이터는 13비트(bit) 부호형(signed) 정수로 나타내어진다. 따라서 하나의 13비트 부호형 곱셈기(multiplier)와 2개의 12비트 부호 없는(unsigned) 제곱기(squarer)[17]를 입력 단(stage)에서 사용하였다. 블록도의 간소화를 위해 제곱기의 입력으로 들어가기 전에 절대값 형태로 변환하는 과정은 나타내지 않았다.

설계한 계산기는 2 레벨(level) 파이프라이닝(pipelining) 방법으로 동작한다. [그림 6-2]의 점선은 하위 레벨(lower-level) 클록(clock)의 22 분할 주파수에서 동작하는 상위 레벨(higher-level) 파이프라이닝 레지스터(register)를 나타낸다. 전처리 과정을 거치고 얻은 데이터는 16개의 배열을 가지고 있지만, 계산기에서 사용하고 있는 곱셈기와 제곱기도 파이프라인 되어있기 때문에 하나의 상위 레벨 파이프라인 단을 동작시키기 위해 22 하위 레벨 클록을 사용하였다.



[그림 6-2] FPGA 기반 코사인 유사도 계산기 블록도

제7장 실험 및 평가

본 장에서는 제 4장과 제 5장 및 제 6장에서 설계한 추론 시스템들을 검증하고 설계의 성능 및 비용을 평가한다.

7.1 실험 방법 및 환경

제 4장과 제 5장 및 제 6장에서 설명한 추론 시스템은 Intel i7-7500U CPU와 NVIDIA GeForce 930MX GPU 환경에서 설계하였다. 2 Stacked RNN 및 LSTM RNN 기반 추론 시스템은 python 언어를 기반으로 설계하였으며, 학습 및 추론 과정은Tensorflow 딥 러닝 프레임워크(framework)[18]를 사용하였다.

수집한 데이터 중 70%는 학습용, 30%는 테스트용으로 사용하였다. 분류(classification)을 위해 데이터 셋(dataset)와 해당하는 라벨(label)을 함께 학습하는 지도 학습 기술(supervised learning)을 사용하였으며, 라벨은 one-hot 인코딩(encoding) 기법을 사용하여 표현하였다. 신경망 내의 가중치와 바이어스(bias)는 무작위(random)로 초기화하였고, 비용 함수 최소화를 목표로 학습된다. 비용 함수는 실제 라벨(ground truth label)과 예측된 출력 라벨(predicted output label) 간의 평균 크로스 엔트로피(mean cross entropy)이다. 비용 함수를 최소화하기 위해 최적화 알고리즘으로 Adam optimizer를 사용하였다.

데이터 입력 층과 첫 번째 숨겨진 층 사이에 임베딩(Embedding) 층을 추가로 삽입하고 입력 층에서 들어오는 데이터에 ReLu 함수를 거치게 하여 과적합을 방지하고 학습을 가속화하는데 도움을 줄 수 있도록 하였다. 또한, 배치(batch) 크기와 숨겨진 유닛(unit)의 수를 조절하면서 가장 높은 정확도를 출력할 수 있도록 설계하였다. 임베딩 층에서 ReLu 함수를 사용하며 과적합을 방지하였지만, 데이터의 크기와 신경망의 크기 사이에서 발생할 수 있는 과적합을 다시 한 번 방지하고자 L2 정규화(regularization)을 사용하였다.

코사인 유사도 기반 추론 시스템은 Python 언어 기반으로 설계하였으며, 정확도는 제 6장에서 설명한 알고리즘을 기반으로 출력하였다. FPGA 기반 코사인 유사도 계산기는 베릴로그(verilog) 언어를 기반으로 설계하였으며, Intel mega 함수[19]를 사용하고 Quartus Prime 18.0에서 Intel Cyclone FPGA(5CSEMA5F31C6[20]) 타깃 보드 환경에서 구현하였다.

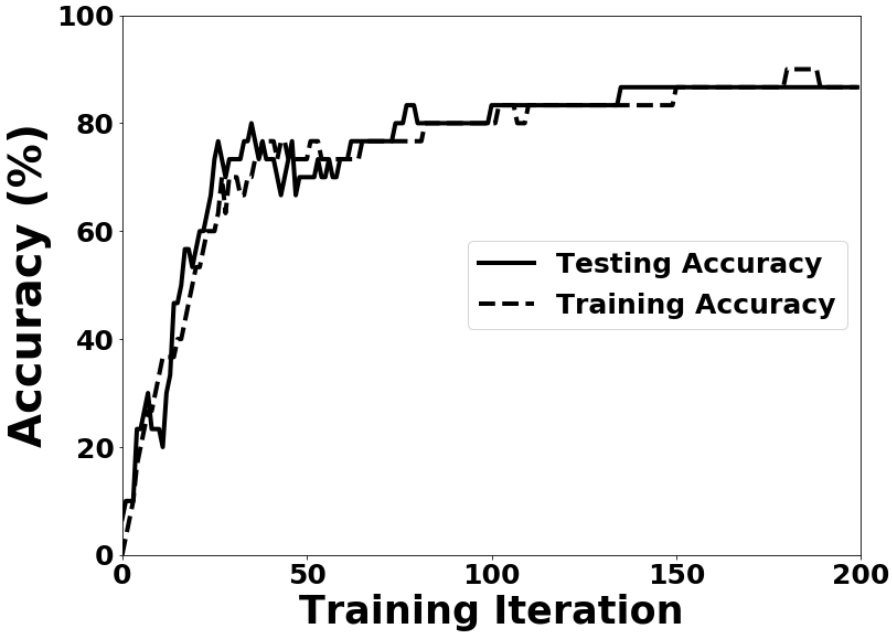
7.2 결과 및 분석

본 논문에서 설계한 추론 시스템들을 검증하게 위해 두 가지의 실험방법을 진행하였다. 2 Stacked LSTM RNN 기반 추론 시스템을 먼저 검증한 뒤, 도출한 결과를 기반으로 전체 추론 시스템들을 검증하였다. 본 절에서는 두 가지 검증에 대한 결과를 설명하고 분석한다.

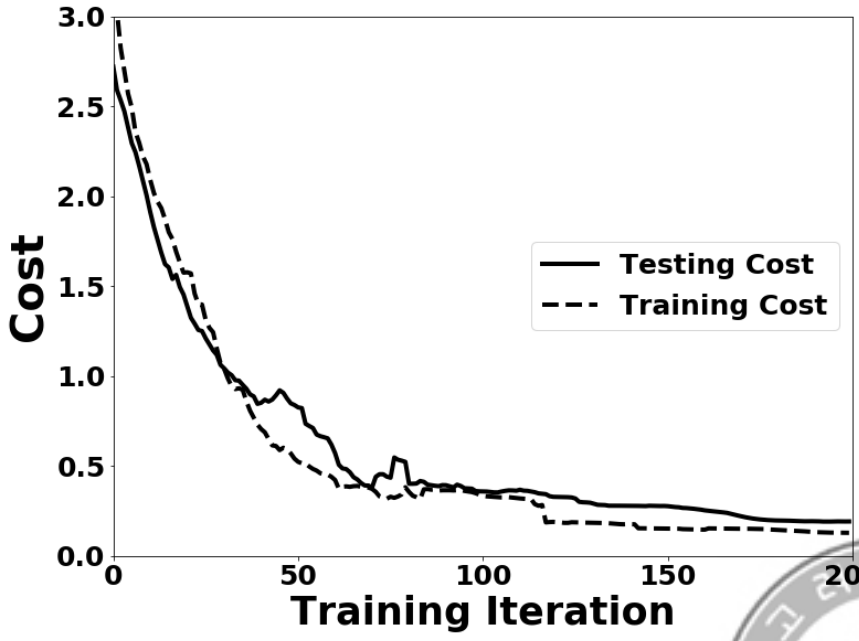
7.2.1 1차 검증

2 Stacked LSTM RNN를 검증하기 위해 전처리 과정을 거치지 않은 데이터를 학습 및 추론에 사용하였다. 가속도와 자이로 센서 값을 모두 사용하여 데이터 셋을 구성하였으며, 단방향 및 양방향 2 stacked LSTM RNN를 모두 실험하였다. 수집한 5명의 운동자 데이터 중 가장 숙련된 운동자와 가장 숙련되지 않은 운동자로만 구성한 2명의 데이터를 사용하여 학습하고 추론하였다.

[그림 7-1]은 단방향 2 Stacked LSTM RNN으로 실험하였을 때, 200번의 반복 학습 동안 기록된 학습 정확도와 테스트 정확도를 나타낸 그래프이며, [그림 7-2]는 반복 학습 동안 기록된 비용 함수의 값을 나타낸 그래프이다. 최종 반복 학습 위치에서 학습과 테스트의 정확도가 86.7%로 기록되어 전처리를 하지 않고 가장 분류하기 어려운 데이터를 기반으로 하였음에도 불구하고 높은 정확도를 나타냄을 확인할 수 있으며, 비용 함수 또한 낮은 값으로 수렴함을 확인할 수 있다.

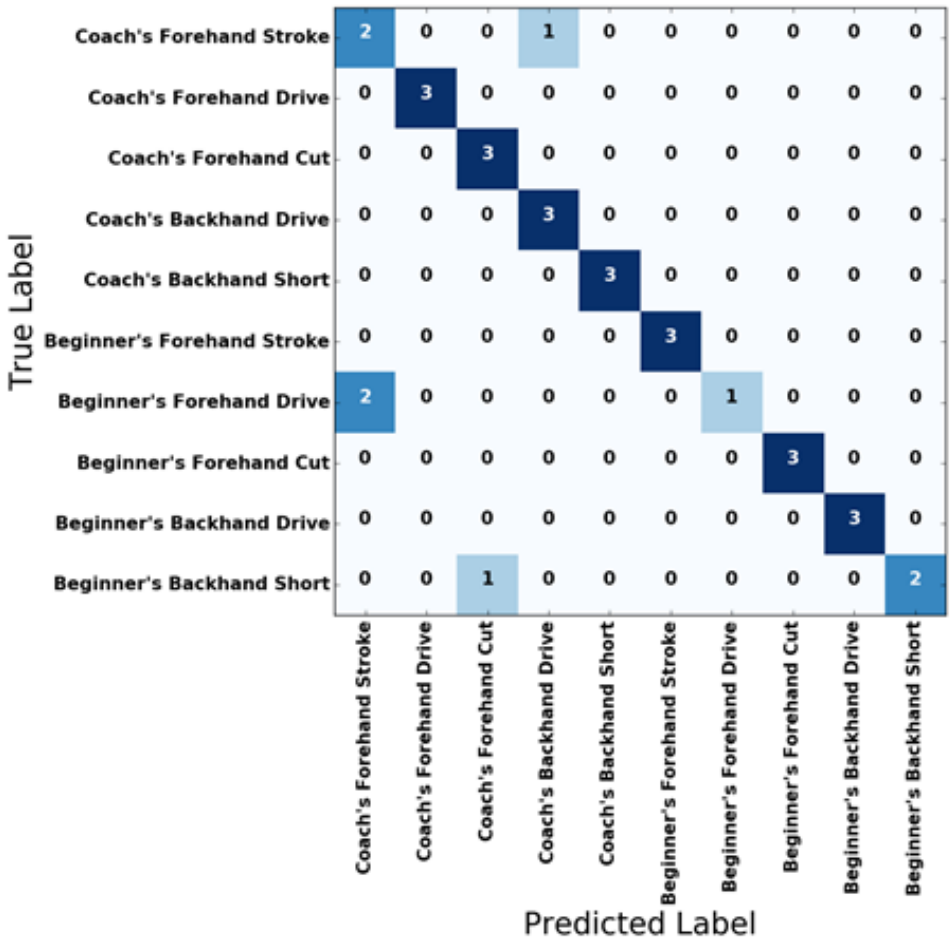


[그림 7-1] 단방향 2 Stacked LSTM RNN의 정확도



[그림 7-2] 단방향 2 Stacked LSTM RNN의 비용함수

[그림 7-3]은 단방향 2 Stacked LSTM RNN으로 학습 과정을 마치고, 테스트용 데이터를 이용하여 추론 과정을 진행하였을 시, 어느 라벨에서 추론에 성공 및 실패하였는지를 보이는 혼동행렬(confusion matrix)이다. 해당 라벨에서 추론이 모두 성공하였을 시 3이 기록되며, 모든 가로 방향의 합은 3이다.



[그림 7-3] 단방향 2 Stacked LSTM RNN의 혼동행렬

[식 7-1]은 설계한 추론 시스템의 성능을 검증하는 4가지 측정 지표를 나타낸다. t_p는 특정 클래스(class) 안에서 참인데 참이라고 맞춘 경우이고, t_n은 거짓인데 거짓이라고 맞춘 경우를 의미한다. 또한, f_p는 거짓인데 참이라고 하여 틀린 경우를 의미하고 f_n은 참인데 거짓이라고 예측하여 틀린 경우를 의미한다. T_p, T_n, F_p, F_n은 모든 클래스에서의 전술한 4가지 경우를 나타낸다. 전체 정확도(accuracy)는 모든 클래스에서 시스템이 예측한 결과 중 실제로 추론에 성공한 것을 수치화한 것을 의미한다. 평균 정밀도(average precision)는 각 클래스에서 추론 시스템이 참이라고 판단한 것 중에 실제로 정답이 참인 것들의 퍼센트를 도출 후 전체 클래스에서 평균을 계산한 것이며, 평균 재현율(average recall)은 실제 정답이 참인 것들 중 시스템이 얼마나 많이 참이라는 정답 예측에 성공하였는지를 퍼센트로 도출 후 전체 클래스에서 평균을 계산한 것이다. F1 점수(F1 score)는 정밀도와 재현율의 조화 평균을 의미한다.

$$Accuracy = \frac{T_p + T_n}{T_p + T_n + F_p + F_n}$$

$$Average\,\mathrm{P}recision = \frac{1}{C}\left(\sum_{l=1}^{C} \frac{t_p^l}{t_p^l + f_p^l}\right)$$

$$Average\,Recall = \frac{1}{C}\left(\sum_{l=1}^{C} \frac{t_p^l}{t_p^l + f_n^l}\right)$$

$$F1\,score = \frac{2 \times \mathrm{P}recision \times Recall}{\mathrm{P}recision + Recall}$$

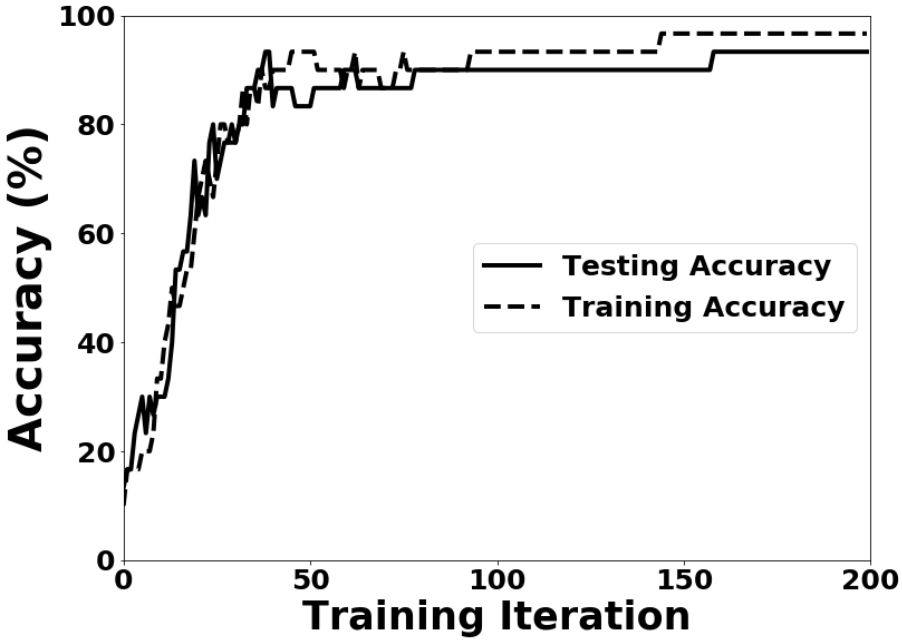
[식 7-1] 시스템 성능 검증 측정 지표

[표 7-1]은 [식 7-1]의 4가지 측정 지표를 사용하여 단방향 2 Stacked LSTM RNN의 추론 성능을 검증한 결과를 나타낸다. 정확도 포함 모든 지표 값들을 통해 2층의 구조로 설계한 LSTM RNN이 본 논문에서 사용하고 있는 데이터와 같은 소규모 응용 분야에서 적합하다는 것을 검증하였다.

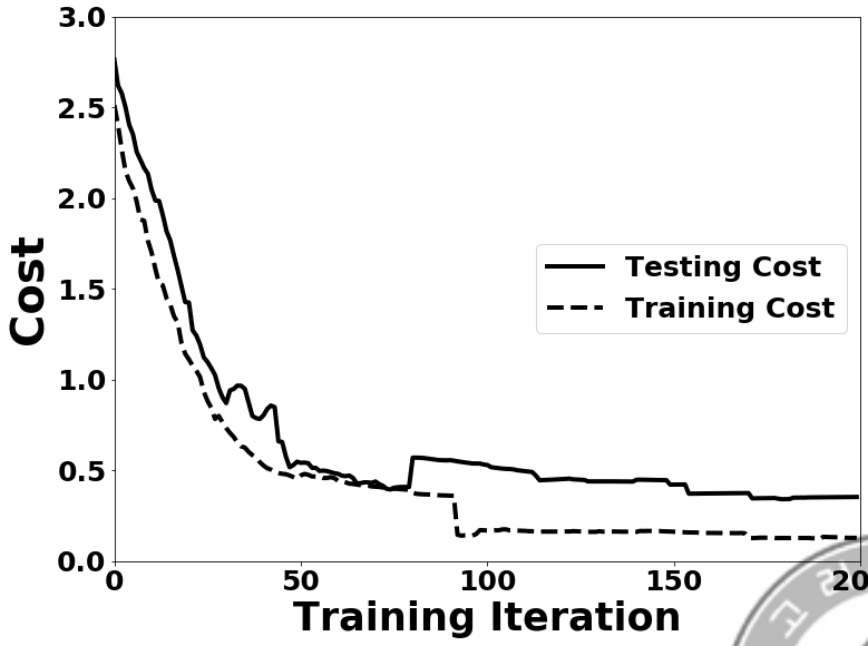
측정 지표	성능
전체 정확도	86.7%
평균 정밀도	87.5%
평균 재현율	86.7%
F1 점수	86.3%

[표 7-1] 단방향 2 Stacked LSTM RNN의 추론 성능

[그림 7-4]은 양방향 2 Stacked LSTM RNN으로 실험하였을 때, 200번의 학습 동안 기록된 학습 정확도와 테스트 정확도를 나타낸 그래프이며, [그림 7-5]는 반복 학습 동안 기록된 비용 함수 값을 나타낸 그래프이다. 최종 반복 학습에서 93.3%의 매우 높은 테스트 정확도를 기록하고 비용 함수도 매우 낮은 값으로 수렴하면서, 단방향 2 Stacked LSTM RNN보다 개선된 성능을 보였다.

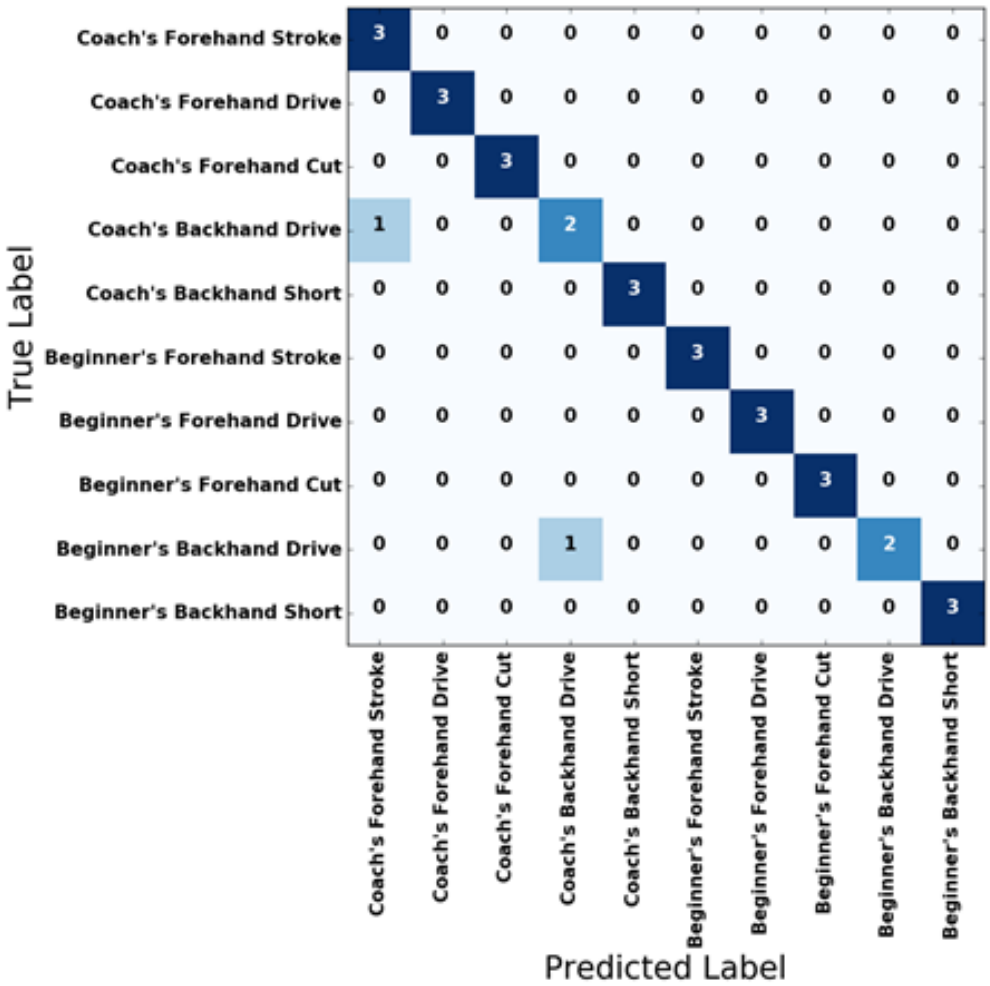


[그림 7-4] 양방향 2 Stacked LSTM RNN의 정확도



[그림 7-5] 양방향 2 Stacked LSTM RNN의 비용함수

[그림 7-6]은 양방향 2 Stacked LSTM RNN 기반 추론 시스템의 혼동 행렬이다. 해석 방법은 [그림 7-3]의 단방향 2 Stacked LSTM RNN 기반 추론 시스템의 혼동 행렬과 동일하다.



[그림 7-6] 양방향 2 Stacked LSTM RNN의 혼동행렬

[표 7-2]는 양방향 2 Stacked LSTM RNN의 추론 성능을 검증하는 4가지 측정 지표 값을 나타낸다. 모든 지표의 값들이 93% 이상 매우 높은 값을 기록하여 후순서 데이터까지 참조하는 후방 경로가 포함된 양방향 2 Stacked LSTM RNN이 단방향 2 Stacked LSTM RNN보다 추론 성능 면에서 개선된 모습을 확인할 수 있다.

측정 지표	**성능**
전체 정확도	93.3%
평균 Precision	95.0%
평균 Recall	93.3%
F1 점수	93.1%

[표 7-2] 양방향 2 Stacked LSTM RNN의 추론 성능

[표 7-3]은 초기 설계 신경망이 가지는 전체 매개변수의 수와 제 5장에서 설명한 프루닝 기법을 사용하여 재설계한 신경망의 매개변수의 양을 나타낸다. 매개변수의 양이 임계값 설정을 통해 목표한 만큼 제거되고 남은 모습을 확인할 수 있다.

	초기 설계	프루닝 후 (30%)	프루닝 후 (60%)	프루닝 후 (90%)
단방향 2 Stacked LSTM RNN	17.58×10^3	12.30×10^3	7.03×10^3	1.76×10^3
양방향 2 Stacked LSTM RNN	34.54×10^3	24.18×10^3	13.82×10^3	3.45×10^3

[표 7-3] 프루닝 기반 신경망의 매개변수의 양

[표 7-4]는 4가지 측정 지표들을 사용하여 프루닝 후 재설계한 신경망의 성능을 검증한 결과들을 나타낸다. 90%의 신경망 내 매개변수를 제거하였음에도 불구하고, 단방향 2 Stacked LSTM RNN의 정확도 성능은 3.4%만이 감소하였고, 양방향 2 Stacked LSTM RNN은 정확도 포함 모든 측정 지표들에서 변화가 없음을 확인할 수 있다.

측정지표	초기 설계	Pruning 후 (30%)	Pruning 후 (60%)	Pruning 후 (90%)
전체 정확도 (단방향)	86.7%	86.7%	86.7%	83.3%
평균 Precision (단방향)	87.5%	87.5%	87.5%	84.2%
평균Recall (단방향)	86.7%	86.7%	86.7%	83.3%
F1 점수 (단방향)	86.3%	86.3%	86.3%	82.4%
전체 정확도 (양방향)	93.3%	93.3%	93.3%	93.3%
평균 Precision (양방향)	95.0%	95.0%	94.2%	95.0%
평균 Recall (양방향)	93.3%	93.3%	93.3%	93.3%
F1 점수 (양방향)	93.1%	93.1%	93.3%	93.1%

[표 7-4] 프루닝 기반 신경망의 추론 성능

[표 7-5]는 초기 설계 신경망과 프루닝 후 재설계한 신경망이 모든 추론을 마치고 시스템을 종료하였을 때까지 소요된 시간을 초 단위로 나타낸 것이다. 매개변수의 양을 큰 폭으로 줄일수록 추론에 소요되는 시간이 줄어드는 모습을 확인할 수 있다. [표 7-4]와 [표 7-5]를 통해 설계한 프루닝 기법을 사용하여 신경망을 재설계 하였을 시 초기 설계 신경망에 비해 추론 성능 면에서 거의 차이가 없으며, 추론 속도 또한 증가하였음을 검증하였다.

	초기 설계	프루닝 후 (30%)	프루닝 후 (60%)	프루닝 후 (90%)
단방향 2 Stacked LSTM RNN	0.23s	0.21s	0.19s	0.15s
양방향 2 Stacked LSTM RNN	0.26s	0.24s	0.22s	0.19s

[표 7-5] 프루닝 기반 신경망의 추론 속도

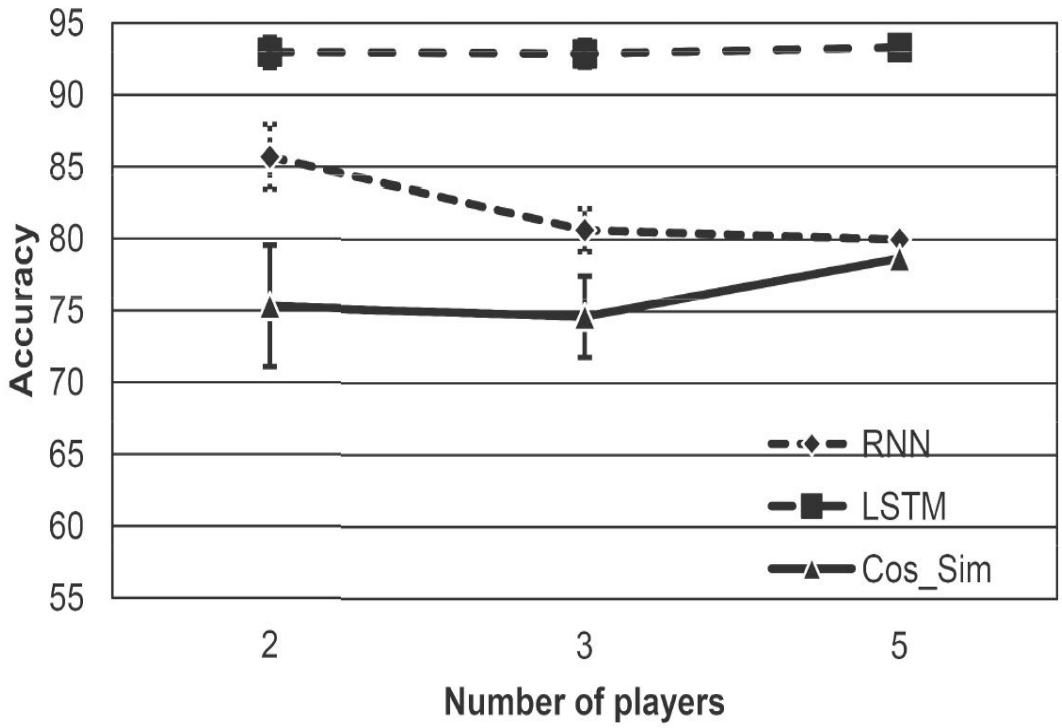
7.2.2 2차 검증

2차 검증 과정에서는 설계한 전체 추론 시스템을 전처리 과정을 거치지 않은 데이터와 전처리 과정을 거친 데이터 모두를 고려하여 다시 한 번 검증하였다. 첫 번째 검증방법과는 다르게 가속도와 자이로 센서 값을 모두 사용하지 않고 가속도 값만을 사용하였으며, 수집한 5명의 운동자 데이터를 모두 사용하여 데이터 세트를 구성하였다. 본 논문에서는 5명의 운동자 데이터만을 고려하지만, 향 후 다양한 응용 분야에서 설계한 시스템이 사용될 때 데이터 셋의 크기가 증가할 수 있는 가능성이 있다. 데이터 셋의 크기의 증가하게 되면 시스템의 계산량 또한 증가하게 되므로 휴대용 장치에서의 시스템 구현에 있어 부담이 될 수 있다. 또한 데이터 분석 과정에서 가속도 센서 값이 각각의 운동자 및 자세에 따라 충분히 특징적인 데이터 배열을 나타내고 있음을 확인하여, 가속도 센서 값만을 이용하여 데이터 셋을 구성하였다. 2명과 3명의 운동자 데이터만을 사용하는 실험 또한 진행하였으며, 데이터 셋에 구성할 운동자 선택의 모든 경우의 수를 고려하여 실험을 진행하였다.

2차 검증은 1차 검증의 결과 및 분석을 기반으로 저비용 시스템과 추론 정확도 사이의 균형을 목표로 진행하였다. 코사인 유사도 기반 추론 시스템을 중심으로 검증을 진행하였고, 1차 검증에서 단방향 2 Stacked LSTM RNN이 충분히 높은 추론 성능을 보였기 때문에 본 검증 과정에서는 양방향 2 Stacked LSTM RNN은 고려하지 않았다. 추가적으로 LSTM RNN보다 더 간소화된 신경망의 크기를 나타내는 RNN을 이용한 단방향 2 Stacked RNN을 고려하였다.

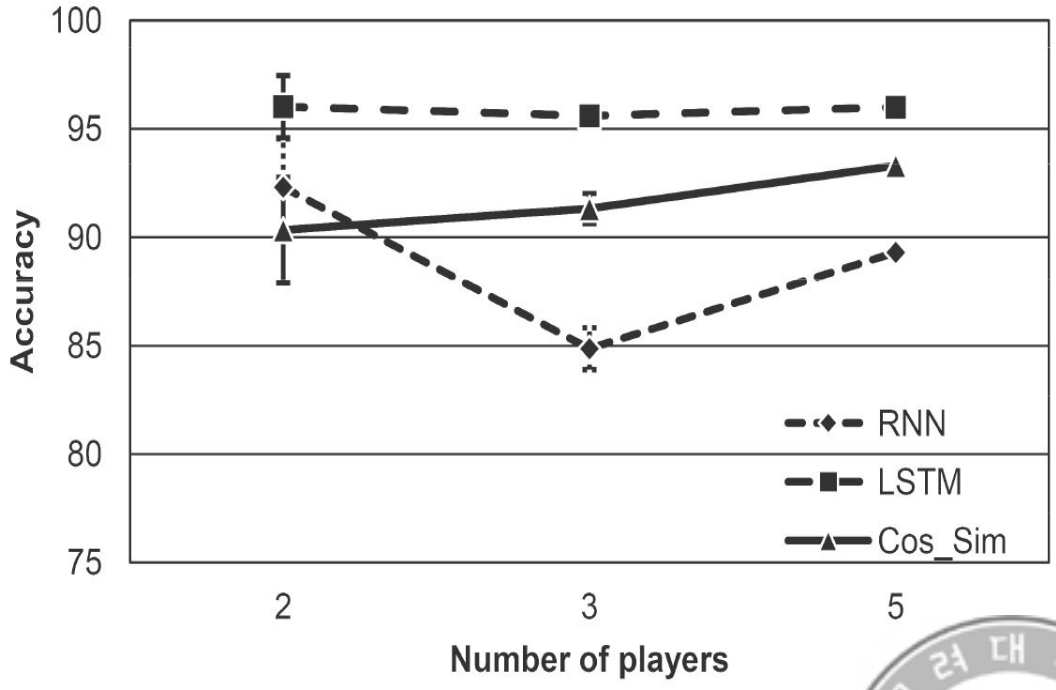
[그림 7-7]은 전처리 과정을 거치지 않은 데이터를 사용하여 추론 시스템을 설계하고 측정한 정확도를 나타낸다. 2명과 3명의 운동자의 경우 데이터 세트 구성에 있어서 운동자 선택 시 발생할 수 있는 10가지 경우의 수를 모두 고려하여 실험하였으며, 2명의 운동자로 실험할 시 10개의 라벨과 3명의 운동자로 실험할 시 15개의 라벨을 함께 학습하였다. [그림 7-7]은 모든 경우의 수의 평균 정확도 값을 기록한 그래프이며 표준 편차는 오류 막대(error bar) 형태로 나타냈다. RNN은 2 Stacked

RNN을 의미하며, LSTM은 단방향 2 Stacked LSTM RNN을 의미하고, Cos_Sim은 코사인 유사도 기반 추론 시스템을 의미한다. 가속도 센서 값만을 이용하여 데이터 셋을 구성하였음에도 불구하고 단방향 2 Stacked LSTM RNN은 1차 검증의 결과와 비교하였을 시 추론 성능이 오히려 더 증가한 모습을 볼 수 있는데, 이는 오히려 하나의 센서 패턴으로만 구성되고 비교할 수 있는 운동자의 수가 증가된 데이터 셋이 추론에 도움이 될 수 있다는 가능성을 보인다. 5명의 운동자를 고려한 경우, 단방향 2 Stacked LSTM RNN이 93.3%의 정확도를 기록하면서 78.7%의 정확도를 기록한 코사인 유사도 기반 시스템보다 14.6% 높은 정확도를 기록하였다.



[그림 7-7] 전처리 과정을 거치지 않은 데이터로 실험한 추론 시스템의 정확도

[그림 7-8]은 전처리 과정을 거치고 얻은 데이터로 실험한 추론 시스템의 정확도를 나타낸다. 그래프의 해석은 [그림 7-7] 그래프의 해석 방법과 동일하다. 모든 추론 시스템의 정확도가 향상된 모습을 확인할 수 있으며, 단방향 2 Stacked LSTM RNN만이 정확도 면에서 거의 변화를 보이지 않았다. 이는 LSTM RNN이 제 2장에서 기술한 바와 같이 신경망의 데이터 입력 배열에서 상대적으로 장거리 의존성을 포착할 수 있기 때문에 정확도 면에서 데이터 셋이 변화하여도 굉장히 안정적인 모습을 보이게 된다. 코사인 유사도 기반 추론 시스템의 성능은 매우 향상된 모습을 볼 수 있으며, 단방향 2 Stacked RNN보다 높은 추론 성능을 보였고 5명의 운동자를 고려한 경우 93.3% 정확도를 기록하며 96%를 기록한 단방향 2 Stacked LSTM RNN과 단 2.7%의 매우 작은 차이만을 보였다.



[그림 7-8] 전처리 과정을 거친 데이터로 실험한 추론 시스템의 정확도

3명의 운동자가 고려된 경우 단방향 2 Stacked RNN의 성능이 하락하는 모습을 보이는데, 3명으로 구성한 데이터 셋 중 특정 운동자의 자세와 또 다른 운동자의 자세가 유사하게 되면 장거리 의존성을 포착할 수 없는 RNN의 단점 때문에 정확도 하락이 관측됨으로 분석하였다. 또한, 2명의 경우 비교해야 하는 운동자의 타법이 다른 한 명의 운동자 타법뿐이고 5명의 경우는 오히려 많은 운동자의 타법과 비교할 수 있어 대체적으로 정확도 면에서 더 나은 성능을 보이는 것으로 판단하였다.

본 검증 과정을 통해 전처리 과정을 거친 데이터로 실험한 코사인 유사도 기반 추론 시스템이 계산량이 방대한 딥 러닝 기술인 2 Stacked LSTM RNN과 정확도 면에서 성능 차이를 보이지만 매우 작은 차이이기 때문에 효율적인 저비용 시스템 설계의 가능성을 확인하였다. 이에 따라 제 5장 2절에서 설명한 바와 같이 코사인 유사도 기반 추론 시스템에서 가장 중요하며 큰 부분을 구성할 유사도 계산 부분을 휴대용 장치인 FPGA에 구현하고 성능을 검증하였다.

[표 7-6]는 설계한 계산기의 성능을 보여주며, FPGA의 매우 작은 부분만을 사용하는 모습을 확인할 수 있다. 최대 동작 주파수는 slow 1100mv 85℃모델에서 측정하였다. 설계한 계산기는 176MHz의 최대 동작 주파수를 기록하였으며, 이는 저장되어 있는 5명의 운동자의 타법들이 기록되어 있는 1575×16 데이터 셋을 약 0.2ms만에 처리할 수 있는 성능을 가진다.

Logic Utilization [ALMs]	Register [FFs]	DSP [Blocks]	Block Memory [bits]	Max. Op. freq. [MHz]
945	2,306	7	4,751	176

[표 7-6] 계산기 성능

제 8장 결론

본 논문에서는 건강관리의 응용 분야에서 운동 자세 측정을 통한 교정 및 재활을 목적으로 추론 시스템을 설계하고 휴대용 장치에서 추론 시스템 구현을 목표로 하였다. 운동 자세 중에서도 팔의 움직임만으로 자세가 결정되는 탁구를 선택하였으며, 미세한 움직임으로도 다른 자세로 인식될 수 있는 가능성이 있어 우수한 성능을 가지는 추론 시스템을 개발 하는데 매우 유용할 것으로 판단하여 탁구 자세를 기반으로 시스템을 설계하였다.

수집한 데이터는 측정 장치를 설계하여 수집하였고 가속도와 자이로 센서 값을 수집하였으며, 전처리 과정을 거치지 않은 원본 데이터와 전처리 과정을 거치고 얻은 데이터를 확보하였다.

추론 시스템은 딥 러닝 모델 중 시계열 형태의 데이터 처리에 강점을 보이는 LSTM RNN을 선택하였고, 이를 2 Stacked LSTM RNN으로 설계하였으며, 저비용 시스템 구현 방안을 모색하였고, 신경망의 크기를 감소시키는 프루닝 기법을 고려하여 적용하였다. 또한, 소규모 응용 분야에서 딥 러닝 기반 시스템의 부담을 덜고자 대안으로 코사인 유사도 기반 시스템을 선택하였으며 구현하였다.

설계한 추론 시스템들은 검증을 두 단계로 나누어 성능을 조사하였고, 최종 2차 검증에서 단방향 2 Stacked LSTM RNN이 전처리 과정을 거친 데이터를 기반으로 실험하였을 시, 96.0%로 가장 높은 추론 성능을 보였다. 하지만 코사인 유사도 기반 추론 시스템이 단방향 2 Stacked LSTM RNN에 비해 2.7% 정도의 차이만을 보이며 매우 높은 추론 성능을 가지는 것을 확인하였다.

코사인 유사도 기반 추론 시스템의 가능성을 확인하고 휴대용 장치에서의 구현에 있어서 가장 큰 부분을 차지할 유사도 계산기 부분을 FPGA에 설계하였다. 계산기는 FPGA의 매우 작은 부분만을 사용하고, 5명의 운동자 타법이 모두 저장된 데이터를 전부 계산하였을 시 약 0.2ms의 시간만을 소모하여 저비용 및 계산 처리 능력에서 매우 높은 성능을 보였다.

참고 문헌

[1] Matthias Kranz, Andreas MöLler, Nils Hammerla, Stefan Diewald, Thomas PlöTz, Patrick Olivier, and Luis Roalter. “The mobile fitness coach: Towards individualized skill assessment using personalized mobile devices.” Pervasive and Mobile Computing, vol. 9(2), pp. 203-215, 2013.

[2] Taehwan Kim, Jeongho Park, Seongman Heo, Keehoon Sung, and Jooyoung Park. “Characterizing dynamic walking patterns and detecting falls with wearable sensors using Gaussian process methods.” Sensors, 17(5), 1172, 2017

[3] Abdulmajid Murad, and Jae-Young Pyun. “Deep recurrent neural networks for human activity recognition.” Sensors, 17(11), 2556, 2017

[4] 임세민, 양종운, 박주영, 오형철. “딥 러닝을 이용한 탁구연습 보조시스템”, 한국정보과학회 KSC 2017 논문집, pp. 960-962, 2017년12월.

[5] 임세민, 김건우, 양종운, 박주영, 오형철. “저비용 탁구연습 보조시스템”, 한국정보과학회 KCC 2018 논문집, pp. 1042-1044, 2018년6월.

[6] Se-Min Lim, Hyeong-Cheol Oh, Jaein Kim, Juwon Lee, and Jooyoung Park. “LSTM-Guided Coaching Assistant for Table Tennis Practice”. Sensors, 18(12), 4112, 2018.

[7] Se-Min Lim, Jooyoung Park, and Hyeong-Cheol Oh. “Low-cost Method for Recognizing Table Tennis Activity.”, Korea Univ, PV-Lab Technical Report TR-2018-3.

[8] Ming Zeng, Le T. Nguyen, Bo Yu, Ole J. Mengshoel, Jiang Zhu, Pang Wu, and Joy Zhang. “Convolutional neural networks for human activity recognition using mobile sensors.” In Mobile Computing, Applications and Services (MobiCASE), 2014 6th International Conference on, pp. 197-205, IEEE, 2014, November.

[9] Yuqing Chen, and Yang Xue. “A deep learning approach to human activity recognition based on single accelerometer.” In Systems, man, and cybernetics (smc), 2015 ieee international conference on, pp. 1488-1492, IEEE, 2015, October.
[10] Hans-Olav Hessen, and Astrid Johnson Tessem. “Human activity recognition with two body-worn accelerometer sensors”, Master’s Thesis, Norwegian University of Science and Technology, Trondheim, Norway, 2015.
[11] Sepp Hochreiter, and Jürgen Schmidhuber. “Long short-term memory.” Neural computation, vol. 9(8), pp. 1735-1780, 1997.
[12] Kaiyuan Guo, Shulin Zeng, Jincheng Yu, Yu Wang, and Huazhong Yang. “A Survey of FPGA Based Neural Network Accelerator.” arXiv preprint arXiv:1712.08934, 2017.
[13] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. “A Novel Connectionist System for Improved Unconstrained Handwriting Recognition.” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31(5), pp.855-868, 2009, May.
[14] Heiga Zen, and Hasim Sak, “Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis.” In Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on, pp. 4470-4474, IEEE, 2015, April.
[15] Michael Zhu, and Suyog Gupta. “To prune, or not to prune: exploring the efficacy of pruning for model compression.” arXiv preprint arXiv:1710.01878, 2017.
[16] Yangda Zhu, Changhai Wang, Jianzhong Zhang, and Jingdong Xu. “Human activity recognition based on similarity.” In Computational Science and Engineering (CSE), 2014 IEEE 17th International Conference on, pp. 1382-1387, IEEE, 2014, December.

[17] Seongjin Choi, and Hyeong-Cheol Oh. “Pipelined squarer for unsigned integers of up to 12 bits.” IEICE TRANSACTIONS on Information and Systems, vol. 101(3), pp. 795-798, 2018.
[18] Martín Abadi, et al. “Tensorflow: a system for large-scale machine learning”, In Operating Systems Design and Implementation (OSDI), Vol. 16, pp. 265-283, 2015.
[19] Intel Corp., “IP and Megafunctions” http://www.intel.com/content/www/us/en/programmable/support/literature/lit-ip.html.
[20] Intel Corp., “Cyclone V device overview” CV-51001, http://www.intel.com, accessed 2018.

감사의 글

학부 연구생을 시작으로 대학원 석사과정까지 어느덧 4년이라는 시간이 흘러, 졸업이라는 마침표 앞에 서있는 현재가 저에게 뿌듯함과 아쉬움을 연이어 주고 있습니다. 먼저 저의 지도교수님이신 오형철 교수님께 고개 숙여 감사의 말씀 올리겠습니다. 또한, 부족한 모습을 너무나 많이 보여드린 것 같아 송구스럽다는 말씀도 올리겠습니다. 제가 병렬연산 및 VLSI 구조 연구실에서 머물렀던 시간동안 교수님께서 학자란 어떤 자세로 배워야하며, 어떤 습관을 가져야하고, 어떤 마음가짐으로 살아가야하는지 뿐만 아니라, 사람이라면 응당 가져야 할 덕목들에 대해서도 가르침을 받고 세상에 나아갈 수 있어 너무나 영광스럽고 행복한 마음만이 가슴에 자리합니다. 그릇된 생각을 멀리하고, 매순간 청렴하며, 배움을 게을리 하지 않아야 한다는 교수님의 가르침 잊지 않고 살아가겠습니다.

저의 두 지도교수님 중 한 분이신 박주영 교수님께도 진심으로 감사의 말씀 올리겠습니다. 매 순간 최선의 노력으로 채우시고, 학자의 길을 갈고 닦으시는 교수님을 뵈며 존경과 동경의 마음을 항상 가졌으며, 이러한 마음들이 저의 연구에 대한 열정을 더욱 강하게 키울 수 있었던 원동력이었습니다. 교수님께 안부 인사 자주 여쭐 수 있도록 하겠습니다. 건강에 항상 유의하시고 빛과 행운이 가득하신 시간들만 교수님과 함께 하기를 진심으로 바라겠습니다.

학부 과정부터 많은 시간을 함께하며 대학원을 함께 진학하였지만, 졸업은 함께하지 못한 저의 동기 양종운에게도 고맙고 항상 응원한다는 말을 전하고 싶습니다. 그리고 제가 정말 누구보다 아꼈던 저의 후배 건우... 걱정이 앞서지만 잘하리라 믿어 의심치 않고 교수님의 제자로서 잘 나아갈 것이라 생각됩니다. 눈이 오나 비가 오나 저의 곁을 머물며 동반자가 되어준 후배 김건우에게 너무나 고마웠고 항상 응원하겠다는 말을 전하고 싶습니다.

항상 아낌없이 베풀어주시고 가르침을 주셨던 선배님들 또한 너무나 감사드리며, 마지막으로 저를 여기까지 이끌어 주시고 부족함 없이 주시려고 노력해주신 부모님께 정말 감사하고 사랑한다는 말씀 올리겠습니다. 너무나 힘든 시간들을 겪으시면서도 오직 아들 걱정 뿐이셨던 사랑하는 저의 부모님께 정말 존경한다는 말씀 또한 올리겠습니다.

이 논문을 시작으로 저는 학자의 길로 가려합니다. 저의 지도교수님과 부모님, 그리고 선배님들과 동기들 및 후배님들이 저에게 아낌없이 베풀어주셨던 가르침과 열정, 그리고 온정을 힘으로 삼아 평생을 세상 어두운 곳에 불을 밝히고 사회 구성원으로서 도움이 될 수 있는 과학자로 살아가겠습니다. 저에게 힘이 되어주신 모든 분들께 다시 한 번 감사의 말씀 올리며 이 글을 마치겠습니다.

임 세 민 올림.