Mediapipe를 활용한 수어 번역 프로그램

Sign Language Translation Program with Mediapipe

전 우 진, 이 득 영, 안 은 영

Woo-Jin Jeon, Deuk-Yeong Lee and Eun-Young Ahn

요 약

본 연구에서는 단일 동작으로 표현되는 지문자를 포함한 연속 동작으로 표현되는 수어를 효과적으로 인식하고 번역하기 위한 알고리즘을 제안한다. Mediapipe Hands를 통해 관절 추출점을 뽑고 이를 벡터와 각도로 변환하여 특징점(Feature)으로 활용한다. 다양한 학습 모델의 특성을 파악하여 인식률을 늘리기 위한 방안으로 Random Forest Classifier를 학습 모델로 사용하였으며, 또한, 카메라로 입력된 연속 동작을 실시간으로 처리하기 위해 슬라이딩 윈도우(Sliding Window) 알고리즘을 제안하였다. 이 알고리즘은 실시간 환경에서 31개의 지문자와 11개의 감정을 나타내는 연속 동작을 동시에 인식하면서도 95.98% 이상의 인식률을 보였다.

Key Words : Sign language, Mediapipe, Random Forest, Artificial intelligence

Ⅰ. 서 론

수어는 말을 배울 수 없는 청각장애인들이 사용하는 “보이는 언어” 이며 이는 구화, 필담과 함께 청각장애인의 주된 소통 방법이다.

국립국어원의 조사에 따르면 청각장애인 중 필담을 이해하지 못하는 청각장애인은 26.9%로 적지 않은 수가 문맹이다. 그러나 의료기관, 공공기관의 경우 수어가 가능한 전문 인력의 부족으로 필담이 불가능한 청각장애인과의 소통에 어려움을 겪는 것이 현실이다. 실제로 청각장애인 중 71.2%가 의료기관에서 수어 통역이 필요하다고 응답했다.

문제 해결을 위해 청각장애인들의 실질적인 의사소통 격차를 줄이고 수어를 인식, 번역할 수 있는 인공지능을 활용한 수어 인식에 대한 기존의 연구는 대부분 CNN(Convolutional Neural Network)과 LSTM (Long Short-Term Memory) 모델로 진행되어왔다[1][2]. 그러나 수어인식을 위한 기존의 학습방법은 과적합에 빠지기 쉽고 복잡한 데이터 처리가 힘들다는 단점이 있다. 우리는 이러한 단점을 극복하기 위한 수어 번역 프로그램을 이 논문을 통하여 제안한다.

Ⅱ. 본 론

1. 손가락 관절 위치 추출

1.1 Mediapipe

본 논문에서 쓰이는 Mediapipe[3]는 Google의 오픈 소스 라이브러리로 객체 감지, 이미지 분할, 손 랜드마크 감지 등의 기능을 제공해 준다. 우리

이 연구는 2024학년도 국립 한밭대학교 졸업 프로젝트의 일환으로 진행되었음.

한밭대학교 정보통신공학과(Department of Information and Communication Engineering, Hanbat National University)

는 이 중에서 손 랜드마크 감지 기능을 활용하여 손가락 관절 위치를 추출하여 활용하였다.

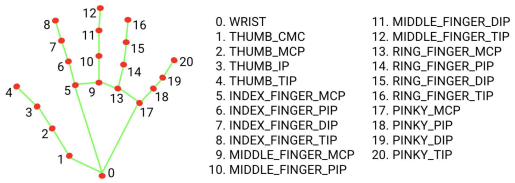

그림 1. Mediapipe Hands의 Landmarks

Fig. 1. Mediapipe Hands Landmarks

1.2 특징점 추출

Mediapipe로 추출한 손가락 관절 위치를 그대로 이용하기에는 손의 각도, 위치, 사람의 신체적 특징에 따른 변화 등 고려해야 할 변수가 너무 많다는 문제점이 존재하였다. 따라서 우리는 정확도를 높이기 위하여 연관이 있는 두 개의 인접한 관절점을 벡터로 만들어 손의 각도, 위치에 따른 문제를 제거하였다.

또한, 사람 손에 따른 크기, 특징에 의한 영향을 최소화하기 위하여 벡터값을 arccos 함수를 통하여 각도로 변환시키고, 그 값을 특징으로 사용하였다.

그러나 우리는 이 과정에서 위의 특징점만을 사용한 경우 'ㅌ', 'ㄹ'또는 'ㅓ', 'ㅕ'와 같이 비슷한 동작의 문자를 혼동하는 경우를 발견할 수 있었다. 따라서 우리는 오류를 일으키는 동작을 비교, 분석하여 손동작을 판단할 때 가장 큰 영향을 미친다고 생각하는 손허리뼈에 해당하는 벡터들 사이의 각도를 추가적인 특징으로 사용하여 정확도를 높였다.

그림 2. 지문자 'ㄹ', 'ㅌ', 'ㅓ', 'ㅕ'

Fig. 2. Fingerspelling 'ㄹ', 'ㅌ', 'ㅓ', 'ㅕ'

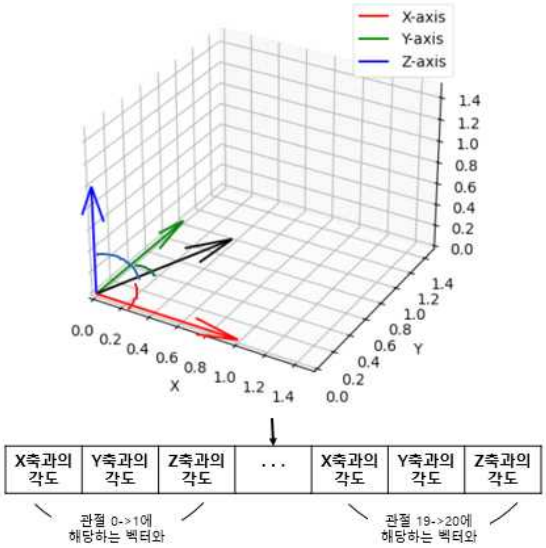

그림 3. 학습에 사용한 특징점 배열

Fig. 3. Feature point array used for learning

2. 학습 모델

본 장에서는 학습 모델 선정과 관련해 학습 모델의 대표적 방법인 Random Forest와 MLP에 대하여 살펴보고 수어 인식의 정확도를 높이기 위한 학습 모델의 선정과 이유를 설명한다.

2.1 Random Forest

Random Forest 모델은 앙상블 학습 방법의 일종으로 <그림 4>와 같다. 훈련 과정에서 구성한 다수의 결정 트리로부터 분류 또는 회귀 분석을 출력하며 동작한다. 중요한 점은 랜덤성에 의해 트리들이 서로 조금씩 다른 특성을 갖는다는 것이다. 이를 통해 각 트리들은 서로 상관이 없으며 결과적으로 일반화 성능을 향상시킨다. Random Forest의 훈련 단계에서는 훈련 목적 함수가 최대가 되는 방향의 노드로 움직인다. 결과적으로 Random Forest 모델은 그렇게 각 트리들에서 나온 결과값들을 평균 내어 최종 결과를 도출하게 된다.

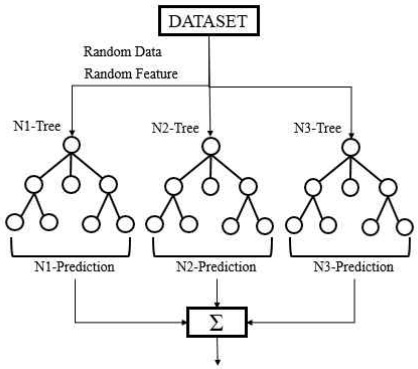

그림 4. Random Forest 모식도

Fig. 4. Random Forest Schematic Diagram.

2.2 MLP(Multi-Layer Perceptron)

MLP는 인공 신경망의 한 종류로 <그림 5>와 같다. 여러 개의 은닉층과 출력층으로 구성된 분류 모델이다. 입력층은 특징점을 나타내는 뉴런들로 구성되며, 은닉층은 여러 개의 뉴런으로 구성되어 각 뉴런은 신호에 가중치를 곱한 후 활성화 함수를 통과시킨다. 출력층은 최종 예측 결과를 나타내는 뉴런들로 구성되며, 일반적으로 클래스에 대한 확률값을 출력한다. MLP는 역전파 알고리즘을 통해 가중치를 조정하며 학습하며 역전파는 출력과 정답 간의 오차를 역방향으로 전파하여 각 층의 가중치를 조정하는 과정을 반복한다.

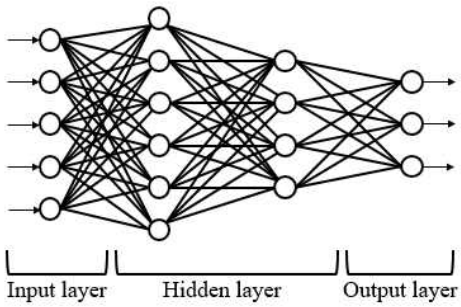

그림 5. MLP 모식도

Fig. 5. MLP Schematic Diagram

2.3 Model 선정

본 논문에서는 Random Forest Model을 사용하였다. [4]에 따르면 신경망 모델의 단점은 매끄러운 함수 즉, 기울기를 기반으로 한 미분 가능한 연속을 전제로 학습하여 불규칙한 패턴에서 약점을 보일 수 있다. 신경망 모델의 또 다른 단점은 특징점 선택이다. 신경망 모델은 입력된 모든 특징점을 동일하게 처리하는 회전 불변성 성질을 가진다. 즉, 의미 없는 특징점이 들어와도 그렇지 않은 특징점과 같이 학습에 영향을 준다.

반면, 트리 기반의 모델은 데이터를 세분화하여 각 트리에서 독립적인 의사결정을 내리는 방식으로 불연속적이고 불규칙한 패턴의 학습에 더 큰 강점을 가진다. 본 연구에서 사용한 특징점 역시 불규칙한 패턴의 데이터를 사용하기 때문에 트리 기반 모델에서 더욱 높은 정확도를 보인다. 수어 인식에 있어서 트리 기반 모델의 또 다른 장점은 스스로 특징점의 중요도를 평가할 수 있다는 것이다. 따라서 비정보성 변수가 많아지더라도 트리 기반 모델은 이러한 변수를 활용하지 않거나, 최소한의 영향만을 받으며 학습할 수 있다. 이는 특징점마다 많이 움직이는 관절, 적게 움직이는 관절이 나누어진 수어 인식에서 강점으로 작용한다. <그림 7>은 수어를 트리 기반 모델로 학습한 결과에서 확인할 수 있다. <그림 7>을 보기 쉽게 만든 것이 <그림 6>이다. 이를 보면 중요도가 크게 나오는 특징점은 실제로도 움직임이 활발한 부위라는 것을 확인할 수 있다.

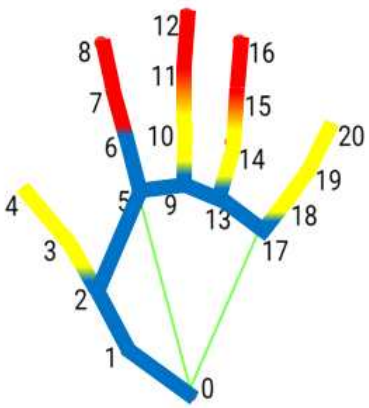

그림 6. 특징점의 중요도

Fig. 6. Importance of Feature

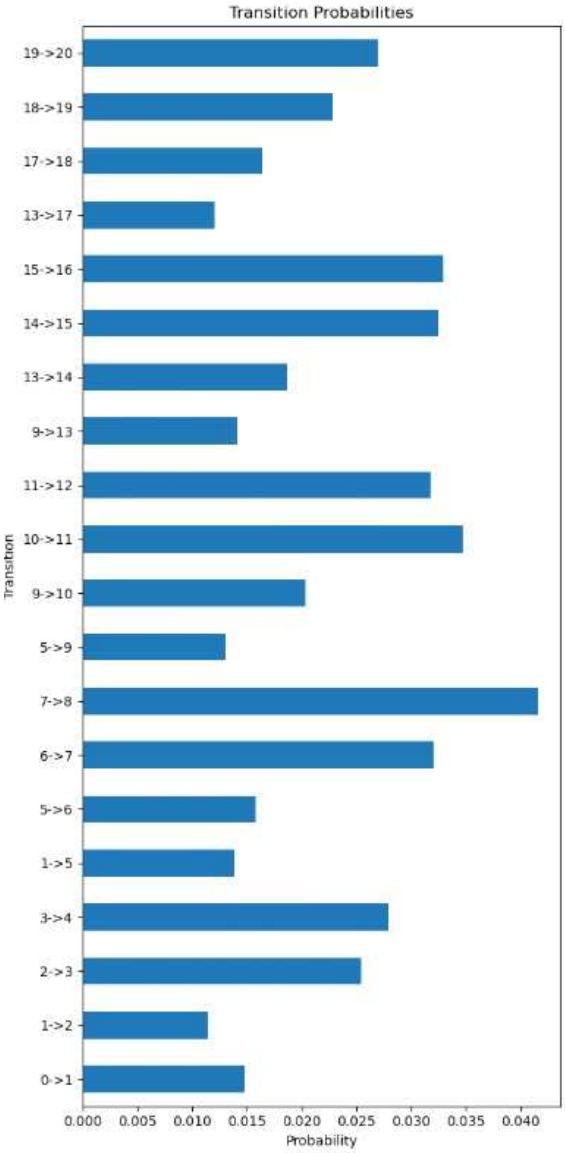

그림 7. Tree 기반 모델이 판단한 특징점의 중요도

Fig. 7. The importance of the feature in Tree-based Model

<표 1>은 관절의 각도를 특징점으로 Random Forest 모델과 MLP 모델을 실제 적용한 결과를 비교한 것이다. 여기서 트리 기반 학습모델의 인식률은 95.1%로 MLP에 비해 4.7%의 인식 향상률을 보인다.

표 1. 모델에 따른 지문자 정확도 표

Table 1. Accuracy table according to Model

Feature	Model	Accuracy
Angle	Random Forest	95.1
Angle	MLP	90.4

3. 수어 인식 알고리즘

이 장에서는 단일 동작으로 표현되는 지문자 입력 방식과 연속된 동작으로 이루어진 수어 입력 방식에 대해 설명하고 이를 효과적으로 번역하기 위한 알고리즘과 인터페이스를 제안한다.

3.1 지문자 입력 방식

본 연구에서는 슬라이딩 윈도우(Sliding Window) 기반의 동작 인식 방식을 채택하여 지문자 입력을 처리하였다. 슬라이딩 윈도우란, 고정된 크기의 데이터 그룹(윈도우)을 설정하고, 새 데이터가 추가되면 가장 오래된 데이터를 제거하여 윈도우를 지속적으로 이동시키는 방법이다. 본 연구에서는 1초 동안 초당 30프레임(fps)의 데이터를 처리하기 위해, 크기가 30프레임인 윈도우를 설정하였다.

윈도우는 매 프레임마다 갱신되며, 새로운 프레임이 추가될 때마다 가장 오래된 프레임이 삭제된다. 이렇게 과거 데이터와 현재 데이터가 겹치도록 유지함으로써, 연속적인 데이터 흐름 속에서 신뢰도 높은 결과를 도출할 수 있다.

윈도우는 시간 t를 기준으로 지속적으로 이동하며 새로운 프레임을 수집하고, 윈도우 내의 동작 인식 결과를 바탕으로 최종 결정을 내린다. 1초(윈도우 크기) 동안 수집된 30개의 프레임에 대해 동일한 동작이 인식된 빈도를 카운팅 하며, 누적 확신도(Accumulated Confidence) 분석을 통해 최종 결과를 도출한다.

슬라이딩 윈도우 내에서는 특정 동작의 빈도를 누적 계산하여 누적 확신도를 산출한다. 만약 동일한 동작이 윈도우 내 90%(=27프레임 이상) 이상 반복되면 해당 동작은 신뢰도 높은 결과로 판단되어 화면에 출력된다. 반면, 특정 동작이 90% 미만으로 관찰되거나 중간에 다른 동작이 포함될 경우, 해당 윈도우는 무효로 처리되어 결과가 출력되지 않는다.

누적 확신도 분석은 윈도우 영역 내에서 특정 동작이 관찰될 때마다 해당 동작의 빈도수를 누적하여 확신도를 계산하는 것이다. 만약 동일한 동작이 90% 이상(= 27프레임 이상) 일치하면, 이를 신뢰할 수 있는 인식결과로 판단하고 화면에 출력한다. 반대로, 윈도우 내에서 다른 동작이 관찰되어 특정 동작의 확신도가 90%에 미치지 못하는 경우, 해당 윈도

우는 무효로 처리되며 인식결과를 확정하지 않는다.

예를 들어 <표 2>과 같은 경우, 1초 동안 총 30프레임이 수집되었을 때, ‘ㄱ’ 이라는 수어가 29프레임에서 동일하게 인식되었다면, ‘ㄱ’ 은 96.7%의 신뢰도로 인정되며 화면에 출력된다. 반면, 중간에 다른 동작이 10프레임 인식된 경우 ‘ㄱ’ 의 빈도가 20프레임이 되어 ‘ㄱ’ 은 66.7%의 신뢰도를 갖고, 현재 윈도우에서는 출력되지 않게 된다. <그림 8>에서는 <표 2>에 대하여 각각 ‘ㄱ’ 이 출력되는 경우(파랑), ‘ㄱ’ 이 출력되지 않는 경우(빨강)로 그래프를 나타내고 있다.

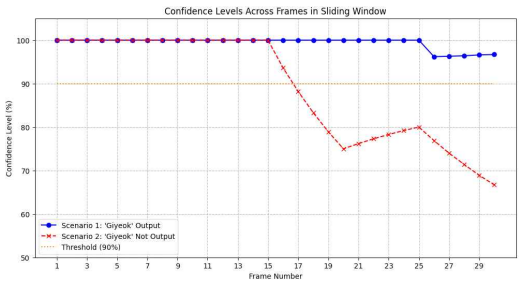

그림 8. 타임 윈도우를 이용한 ‘ㄱ’ 출력의 예시
Fig. 8. Example of an “Giyeok” output using a time window

표 2. 타임 윈도우를 이용한 ‘ㄱ’ 출력의 예시
Table 2. Example of an “Giyeok” output using a time window

< ‘ㄱ’ 이 출력되는 경우>

프레임 번호	동작 인식 결과	누적 ‘ㄱ’ 프레임 수	확신도 (%)	출력 여부
1~25	‘ㄱ’	25	100.0	-
26	다른 동작	25	96.2	-
27~30	‘ㄱ’	29	96.7	‘ㄱ’ 출력

< ‘ㄱ’ 이 출력되지 않는 경우>

프레임 번호	동작 인식 결과	누적 ‘ㄱ’ 프레임 수	확신도 (%)	출력 여부
1~15	‘ㄱ’	15	100.0	-
16~20	다른 동작	15	75.0	-
21~25	‘ㄱ’	20	80.0	-
26~30	다른 동작	20	66.7	최종 출력 없음

이 방식은 프레임마다 독립적으로 처리하는 기존의 단일 프레임 방식과 달리, 연속된 프레임을 관찰하면서 확신도를 누적하기 때문에 보다 안정적이고 신뢰성 높은 인식결과를 제공할 수 있다. 슬라이딩 윈도우 기반 누적 분석은 랜덤한 입력으로 인한 오인식을 방지하고, 신뢰성 높은 결과를 도출함으로써 효율적이고 직관적인 수어 인식을 가능하게 한다.

지문자 관리 기능은 슬라이딩 윈도우 개념을 확장하여 구현하였다. 글자 나누기, 한 글자 삭제, 스택 초기화와 같은 기능이 구현되었는데, 이 기능들은 일정 시간 동안의 윈도우를 설정하여 처리된다. 예를 들어, 기능을 위한 특정 동작이 윈도우 내에서 안정적으로 인식되면 해당 기능이 수행된다. 반대로, 윈도우 범위 내에서 불규칙하거나 튀는 동작이 많이 발생하면 기능 수행은 취소된다. 더하여, 지문자가 인식된 상태에서 사용자가 키보드 “엔터”를 눌러 인식결과를 수동으로 입력하는 등의 기능도 지원하였다. 이를 통해 사용자가 원하는 시점에 글자를 입력할 수 있는 유연성을 제공하였다.

본 연구에서는, 한글의 자음, 모음을 결합하여 단어와 문장을 표현할 수 있도록, 글자 스택을 제안한다. 글자 스택은 초성, 중성, 종성을 결합하여 완성된 글자를 형성하는 구조로 설계되었다. “자음 + 모음”, “자음 + 모음 + 자음” 등의 조합 상황에 맞춰 스택에 글자가 추가되며, 이를 통해 수어 입력 결과를 적절히 표현할 수 있다. 또한, 특정 자음이 연속으로 입력된 경우 이를 쌍자음 혹은 겹받침으로 변환하여 출력한다.

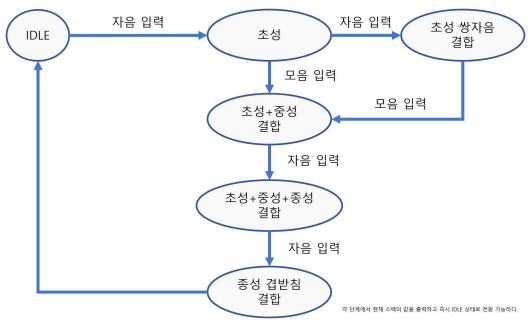

그림 9. 음운 조합 유한 상태 머신(FSM)
Fig. 9. phonological combination Finite State Machine(FSM)

3.2 연속된 동작으로 이루어진 수어 입력 방식

본 연구에서는 타임 윈도우(Time Window)를 기반으로 한 간단한 연속된 동작 인식 알고리즘을 채택하였다. 이 알고리즘은 두 개의 주요 동작을 순차적으로 인식하여 최종적으로 하나의 의미 있는 동작을 출력하는 방식으로, 실시간 처리에서의 지연을 최소화하고 신뢰성을 확보할 수 있다.

수어는 단일한 정지 동작이 아니라, 연속적인 움직임을 통해 표현되는 동적 언어이다. 연구 과정에서 관찰한 결과, 수어 동작의 대부분은 몇 개의 구분되는 주요 동작으로 구성됨을 알 수 있었다. <그림 10>은 본 연구에서 학습한 11개의 감정을 표현하는 수어이다. 여기서 보면 각 수어 동작들은 두 개의 주요 동작만으로 인식이 가능함을 확인하였다. 따라서 본 연구에서는 이러한 수어의 특성을 반영하여, 연속적인 움직임 중 두 개의 주요 동작을 인식의 기본 단위로 설정하였다.

두 개의 주요 동작의 인식은 타임 윈도우 내의 연속된 데이터를 기반으로 한다. <그림 11>의 예와 같이 첫 번째 동작이 인식된 후 윈도우 범위 내에서 두 번째 동작이 인식되면 표현하고자 하는 최종 결과가 출력된다. 구체적으로, 첫 번째 동작(예: 48번)이 일정 시간(예: 0.5초) 동안 연속적으로 인식된 후, 두 번째 동작(예: 49번)이 이어서 일정 시간(예: 0.5초) 동안 연속적으로 인식되면 두 동작이 결합되어 하나의 의미 있는 결과(예: “묻다”)를 출력한다. 두 개의 주요 동작 사이에서 일어나는 일정 시간(예: 0.5초) 미만의 움직임에 대한 인식 결과는 잡음으로 판단하여 무시한다. 정리하면, 중간에 다른 동작 혹은 0.5초 미만의 짧은 동작이 인식되더라도, 첫 번째와 두 번째 동작이 시간 조건을 충족하면 최종 동작이 출력된다.

본 연구에서 제안하는 방식은 두 개의 주요 동작만으로 수어를 인식하기 때문에 입력된 데이터에서 무효 동작이나 튀는 데이터를 무시함으로써 신뢰성을 높이고 오인식을 방지한다. 이와 같은 구조는 연속된 동작 인식에서 유연성과 신뢰성을 동시에 확보한다. 또한 기존의 RNN 등의 학습을 기반으로 하는 수어 인식에 비해 간단한 방식으로 인식이 가능하기 때문에 실시간 적용에 매우 유리하다.

그림 10. 감정표현 수어

Fig. 10. Sign language for emotional expressions

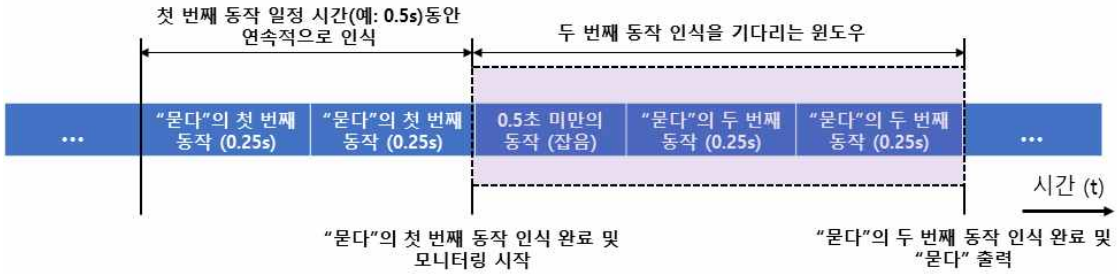

그림 11. 타임 윈도우 알고리즘

Fig. 11. Time Window Algorithms

3.3 사용자 인터페이스 설계

본 연구에서 개발된 사용자 인터페이스는 글자 스택과 실시간 수어 입력 방식을 기반으로 설계되었으며, 효율적인 텍스트 출력과 감정표현을 지원하기 위해 다양한 기능을 통합하였다. 연속 동작으로 표현되는 수어와 지문자의 혼합 사용은 실제 사용 사례에서 필요성이 낮기 때문에 별도로 분리하여 처리하였다. 이를 통해 연속된 동작으로 이루어진 수어인식의 모니터링 과정에서도 다른 수어 입력이 방해받지 않도록 설계하였다.

텍스트 스택의 길이가 제한을 초과하면 오래된 텍스트가 자동으로 삭제되며, 스택은 데큐(deque)로 처리되어 하단에서 스크롤 형태로 표시된다. 또한, 프로그램 오른쪽에는 로그 창을 배치하여 실시간 동작 결과와 메시지를 제공한다. 사용자 편의를 고려해 <그림 12>와 같은 방식으로 GUI를 설계하였다.

그림 12. 손동작 인식 사진 및 전체적인 GUI

Fig. 12. Hand gesture recognition photo and overall GUI

4. 실험 및 분석

4.1 실험 환경

본 논문에서는 Google에서 제공하는 Mediapipe를 사용하여 나온 손가락 관절 위치를 가공하여 특징점으로 사용하였다. 학습에 사용된 데이터는 직접 촬영한 영상들을 활용하였으며, 테스트는 720p 해상도와 30fps의 성능을 가진 웹캠을 통해 진행되었다. 학습 환경은 OS Windows 10, CUDA 10.2버전, Python 3.11 버전에서 Anaconda 가상환경을 활용하여 구현되었다.

4.2 학습 데이터 세트

학습 데이터는 <그림 13>의 31개 지문자와 <그림 10>의 감정표현 11개에 대해 총, 27,531개의 이미지를 사용하였다.

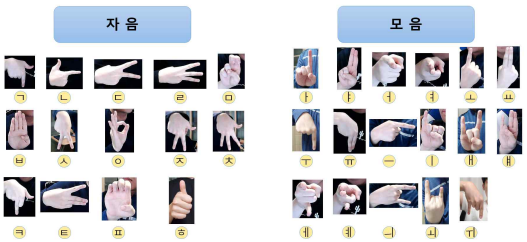

그림 13. 지문자

Fig. 13. fingerspelling

4.3 실험 결과 및 분석

1) 연속 동작 실험 결과

지문자와 연속 동작으로 표현되는 수어의 인식률을 검증하기 위해 학습에 참여하지 않은 테스트 데이터를 수집하여 직접 검증하였다. 우선 연속 동작으로 표현되는 수어의 인식 정확도를 검증하기 위해 각 연속 동작에 대한 실시간 테스트를 51번씩 수행하였다. 테스트 조건은 3초 이내에 연속 동작을 올바르게 판별할 수 있는지의 여부로 정확도를 측정하였다. 결과는 <그림 14>과 같으며, 막대 위에는 “성공 횟수/총 테스트 횟수”(예: 48/51)와 함께 해당 정확도가 표시되어 있다.

실험 결과, 11개의 모든 연속 동작에 대해서 최소 90% 이상의 정확도를 보였으며, 평균적으로는 약 95.19%의 정확도로 인식하는 것을 확인할 수 있었다.

2) 지문자 실험 결과

지문자의 정확도를 검증하기 위해서, 학습 데이터와 별개로 유튜브에서 각 문자당 51장씩 총 1,581장의 테스트 데이터를 수집하여 실험을 진행하였으며 결과 95.1%의 인식률을 보였다.

또한 실시간 번역 프로그램의 동작 환경에서의 대략적인 정확도를 평가하기 위해 실시간 테스트를 진행하였다. 특정 동작의 입력 요청이 주어진

그림 14. 연속 동작으로 표현되는 수어 인식 정확도

Fig. 14. Sign language recognition accuracy expressed in continuous motion

후, 3초 이내에 동일한 수어가 1초 이상 연속으로 인식되었을 때 정답으로 처리하였다. 실시간 테스트 역시 각 문자당 51회씩 진행하였으며, 기존 단일 사진 처리 방식과 비교했을 때 연속적인 영상 처리 방식이 1.67% 높은 정확도를 나타내었다. 이는 영상 처리 방식이 연속된 프레임에서 손 모양의 일관성을 고려할 수 있기 때문에 정확도가 향상된 것으로 분석된다. 결과는 <그림 15>과 같으

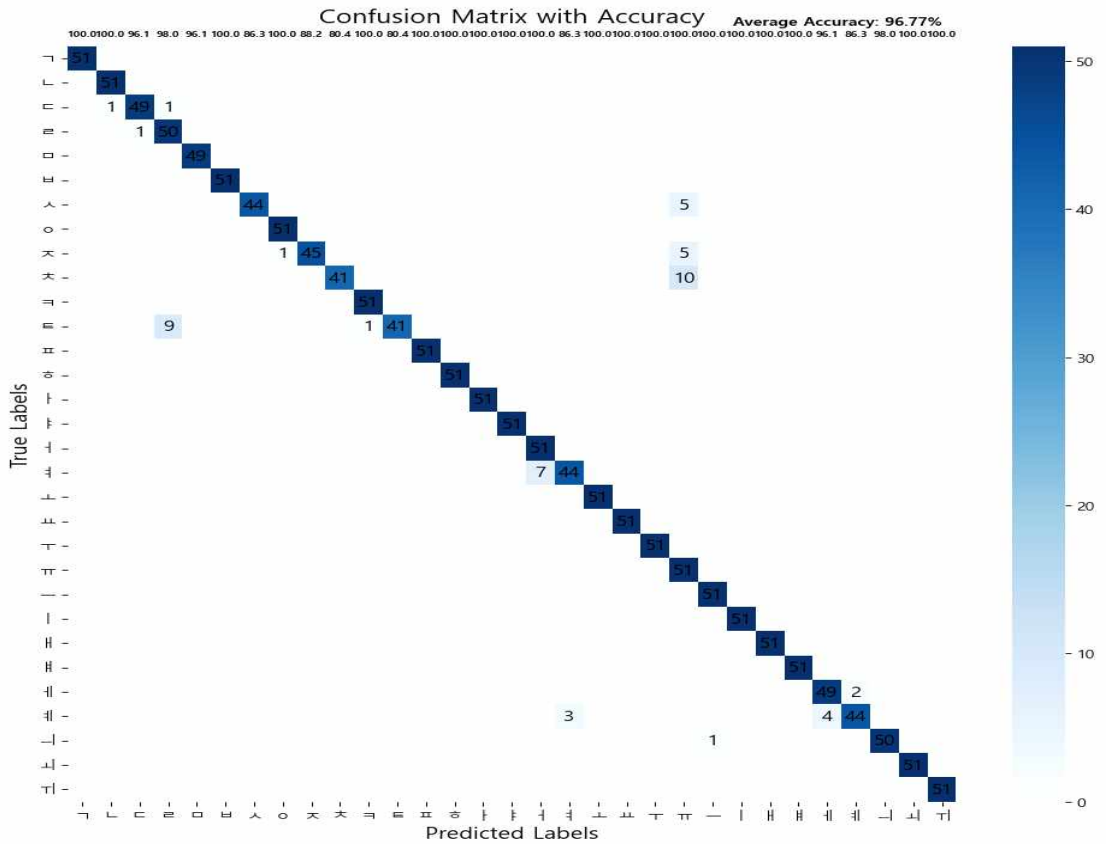

그림 15. 혼동행렬 (실시간 측정)

Fig. 15. Confusion matrix (Real-time measurement)

며, 특히 ‘ㅓ’와 ‘ㅕ’, ‘ㅔ’와 ‘ㅖ’와 같은 모음 간의 혼동이 자주 발생하였다. 또한 <그림 16>과같이, 본 실험의 모델 성능 평가에서 대부분의 클래스에서 높은 성능을 보였으나, 일부 클래스(‘ㄹ’, ‘ㅠ’)의 정밀도가 상대적으로 낮았으며, ‘ㅊ’, ‘ㅌ’ 클래스의 민감도가 낮게 나타났다. 또한, F1 score는 ‘ㄹ’, ‘ㅠ’에서 가장 낮은 값을 보였으나, 전반적인 성능은 양호한 수준을 유지하였다.

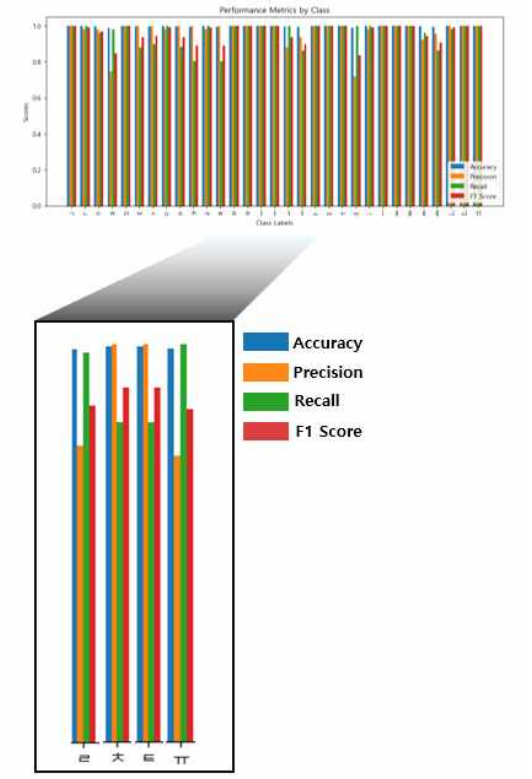

그림 16. 모델 성능 평가

Fig. 16. Metrics for Performance Evaluation

3) 실험 결과 분석

본 연구에서는 지문자와 연속 동작을 인식하여 다양한 표현을 수행해 본 결과, 번역기로 활용할 수 있을 정도의 타당한 결과를 얻을 수 있었다. 실시간 테스트 환경에서 지문자는 96.77%, 연속 동작은 95.19%로 평균 95.98%의 정확도를 달성하였다. 이는 지문자와 연속 동작을 구분하여 최적화된 알고리즘을 설계한 결과, 각각의 특성을 효과적으로 반영했음을 보여준다.

연속 동작의 경우, 일부 동작이 인식되지 않는 사례를 분석한 결과, 사람이 직접 비교하더라도 모호하게 보이는 동작이거나, 손이 빠르게 움직여 모델이 정확히 인식하기 어려운 경우가 주된 원인임을 확인할 수 있었다. 특히, 연속 동작의 복잡한 움직임은 각 동작 간의 경계가 모호하거나, 일부 손동작이 너무 짧은 시간 내에 발생할 때 혼동을 야기할 수 있었다.

지문자의 경우, 일부 인식률 저하의 원인으로는 손가락 하나를 펴거나 오므리는 정도와 같은 미세한 차이를 모델이 명확히 구분하지 못하는 문제가 확인되었다. 이러한 문제는 특히 사람이 보아도 비슷해 보이는 손동작들로 이루어진 데이터 클래스에서 더욱 두드러졌다. 예를 들어, 'ㄹ'과 'ㅌ', 'ㅓ'와 'ㅕ'와 같이 손가락의 각도와 위치가 유사한 지문자들은 모델이 충분히 학습하더라도 인식률이 떨어지는 경향이 있었다. 이는 학습 데이터의 다양성을 더욱 강화하고, 손동작 간의 미세한 차이를 명확히 구분할 수 있는 알고리즘 개선이 필요함을 시사한다. 결과적으로, 본 연구에서 제안한 슬라이딩 윈도우 기반의 인식 알고리즘은 지문자를 효과적으로 인식하면서도, 실시간 환경에서의 신뢰성과 유연성을 동시에 달성하였다. 연속 동작의 경우, 모호한 동작을 필터링하고 명확한 두 동작의 결합을 바탕으로 의미 있는 결과를 출력함으로써, 실시간 환경에서 신뢰도 높은 결과를 제공할 수 있었다. 이러한 접근 방식은 수어 번역 시스템이 실제 환경에서 활용될 가능성을 높이는 중요한 요소로 작용할 것으로 기대된다.

Ⅲ. 결 론

본 논문에서는 수어 번역 프로그램을 개발하였다. Mediapipe를 활용해 손의 관절 위치를 추출한 뒤, 관절 간의 연관성을 벡터로 표현하고 이를 각도로 변환하여 학습에 사용하였다. 또한, 트리 기반 모델인 Random Forest와 신경망 모델인 MLP를 비교·분석하여 최적의 학습모델을 선택함으로써 인식률을 향상시켰다. 이를 통해 11개의 감정표현과 31개의 지문자를 학습시킨 후 실험을 통해 정확도와 인식 결과를 검증하였다. 최종적으로, 연속된 동작 인식 알고리즘과 사용자 중심 인터페이스를 결합하여 수어 번역을 위한 의사소통 툴을 구현하였다. 다만, 본 연구에서 학습한 단어와 문자의 수가 제한적이고, 프롬프트 출력에 다소 시간이 소요되는 한계

가 있다. 향후 연구에서는 더 많은 데이터를 학습시키고, 이에 맞춰 모델의 파라미터를 최적화한다면 더욱 개선된 결과를 얻을 수 있을 것으로 기대된다.

Ⅳ. 참고 문헌

[1] 김기찬, 하란, “*딥러닝을 통한 실시간 수어 번역 프로그램 개발.*” 한국정보과학회 학술발표논문집, pp.1,774-1,776, 2022.

[2] 조현수, 허재영, 정찬봉, 남건, 백승태, 이준우, 김호용, 박정진, 김은수, "이미지 학습을 통한 수어 번역기 구현." Proceedings of KIIT Conference, pp.751-754, 2022.

[3] Google, MediaPipe

[4] Grinsztajn, L., Oyallon, E., & Varoquaux, G. “*Why do tree-based models still outperform deep learning on tabular data?*” Preprint, 2022, https://arxiv.org/abs/2207.08815