

극한 환경에서의 견고한 벡터 통신 시스템

작성자: 30501 김도현, 30202 강민재

날짜: 2025년 12월 3일

요약: 본 연구는 물리적 거리 감쇠($\frac{1}{r^2}$)와 고강도 노이즈 환경에서도 로봇이 명령을 정확히 수행할 수 있도록 하는 **벡터 임베딩 및 Denoising Autoencoder(DAE)** 시스템을 제안한다. 실험 결과, 제안된 시스템은 기존 방식 대비 유효 통신 거리를 미약하게 확장시켰으며, 신호 소멸 구간에서도 높은 신뢰도(Confidence)를 유지함을 확인하였다.

1 서론 (Introduction)

로봇 원격 제어에 있어 통신 안정성은 필수적이다. 기존의 심볼릭 명령은 노이즈에 취약하여 비트 하나만 바뀌어도 오작동을 일으킬 수 있다. 본 프로젝트에서는 의미 기반의 **고차원 벡터 통신(Vector Communication)**을 도입하고, 극한의 노이즈를 제거하는 **잔차 기반 DAE**를 적용하여 시스템의 견고함(Robustness)을 확보하고자 한다.

2 방법론 (Methodology)

2.1 통신 임베딩 (Multi-task Embedding)

30개의 통신 명령을 328차원의 벡터로 변환하였다. 이때 벡터가 서로 구분되면서도(Identity), 유사한 동작끼리는 뭉치도록(Category) 다음과 같은 Multi-task Loss를 사용하여 학습하였다.

$$L_{\text{total}} = \alpha \cdot L_{\text{category}} + (1 - \alpha) \cdot L_{\text{identity}}$$

2.2 물리적 채널 시뮬레이션

현실적인 통신 환경을 모사하기 위해 **역제곱 법칙(Inverse Square Law)**을 적용하였다. 거리가 멀어질수록 신호는 급격히 감쇠하며, 배경 잡음(Gaussian Noise)이 상대적으로 커지게 된다.

$$\mathbf{x}_{\text{rx}} = \mathbf{x}_{\text{tx}} \cdot \frac{1}{r^2} + \varepsilon, \quad \varepsilon \sim N(0, 1)$$

여기서 r 은 거리(m), ε 은 표준정규분포를 따르는 노이즈이다.

2.3 잔차 기반 노이즈 제거 (Residual DAE)

수신된 신호의 복원을 위해 Autoencoder를 도입하였다. 특히, 원본을 직접 예측하는 대신 **노이즈 성분만을 예측하여 제거**하는 잔차 학습(Residual Learning) 기법을 사용하여 학습 효율을 높였다.

$$\mathbf{x}_{\text{clean}} = \mathbf{x}_{\text{rx}} - \text{Net}(\mathbf{x}_{\text{rx}})$$

3 실험 및 결과 (Experiments)

3.1 실험 환경

- 거리 범위: 1.0m ~ 7.0m (0.5m 단위 정밀 측정)
- 반복 횟수: 각 거리당 10회
- 비교군: 일반 수신(Standard) vs DAE 필터 적용(Proposed)

3.2 성능 비교 그래프

아래 그래프는 거리에 따른 통신 성공률과 모델의 확신도 변화를 보여준다.

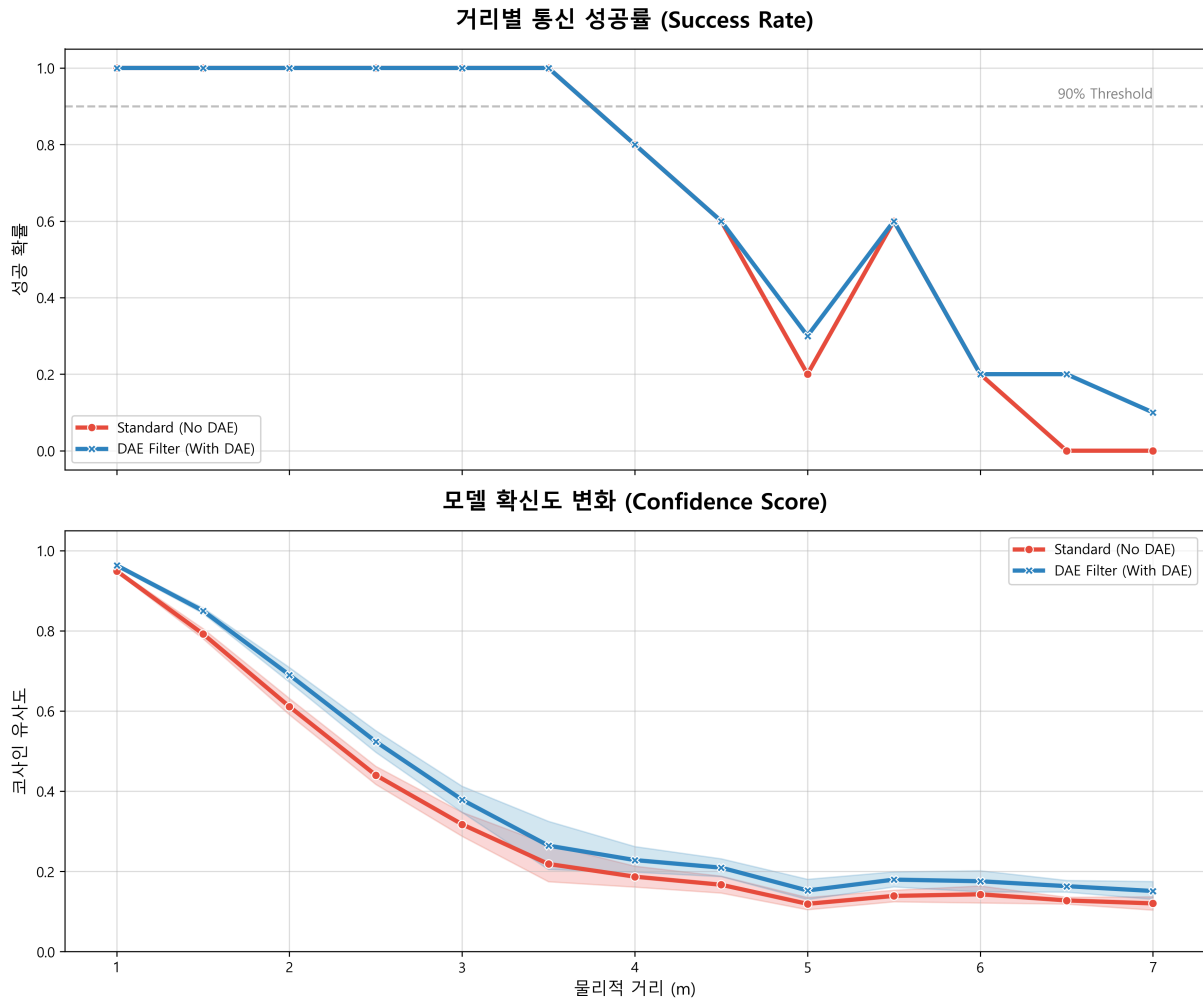


그림 1: 거리별 통신 성공률 및 확신도 비교 (빨강: 일반, 파랑: DAE)

3.3 결과 분석

위 그래프에서 **DAE 모델(BLUE)**이 대부분의 경우 **일반 모델(RED)**보다 나은 성능을 보임을 알 수 있다. 특히 하단의 확신도(Confidence) 그래프를 보면, DAE 모델이 **물리적 거리 감소**로 인해 신호가 거의 소멸한 5m 이후의 구간에서도 더 높은 코사인 유사도를 유지하며 안정적인 모습을 보임을 알 수 있다.

4 결론 (Conclusion)

본 프로젝트를 통해 물리적 거리 감쇠가 적용된 극한 환경에서도 딥러닝 기반의 필터링 기술(DAE)이 통신 거리를 늘릴 수 있음을 증명하였다. 향후 연구로는 보다 **희소성 있는 벡터 공간을 형성하기 위한 Triplet Loss**를 사용하여 성능을 극대화할 예정이다.

해당 프로젝트는 https://github.com/kimknifestring/vector_communication-for-Extreme-situation에서 확인해 볼 수 있다.