Journal of KIIT. Vol. 23, No. 11, pp. 279-282, Nov. 30, 2025. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2025.23.11.279

건설 안전 모니터링을 위한 장면그래프와 템플릿 기반 실시간 비디오 캡셔닝 시스템 구현

이석희*, 고대식**, 송석일***

Implementation of a Real-Time Video Captioning System based on Scene Graphs and Templates for Construction Safety Monitoring

Seokhee Lee*, Daesik Ko**, and Seokil Song***

본 과제(결과물)는 2025년도 교육부 및 충청북도의 재원으로 충북RISE센터의 지원을 받아 수행된 지역혁신중심대학지원체계(RISE) 글로컬대학30의 결과입니다(2025-RISE-11-004)

요 약

기존 대규모 비전-언어 모델(VLM)과 트랜스포머 기반 비디오 캡셔닝 모델은 높은 성능을 보이지만, 해석 가능성 부족, 환각, 지연 등으로 산업현장 적용에 문제가 있다. 이러한 한계를 극복하기 위해 본 논문에서는 제안하는 캡셔닝 방법은 규칙 및 템플릿 기반의 투명한 추론 경로를 도입하여 모델을 경량화하고, 전 과정(검출 - 장면 그래프 생성 - 템플릿 선택 - 캡셔닝)의 로깅을 통해 설명 가능성과 사후 검증 기능을 보장한다.

Abstract

Large-scale vision‐language models and Transformer-based video captioners deliver strong accuracy but remain ill-suited to industrial use due to limited interpretability, hallucination risks, and inference latency. We address these limitations with a lightweight captioning framework that employs a transparent rules-and-templates reasoning path and end-to-end operability logs—spanning detection, scene-graph construction, template selection, and caption generation—to provide rigorous explainability and reliable post-hoc auditability.

Keywords
captioning, scene graph, VLM, video, template

Ⅰ. 서 론

이 논문에서는 2024년 276명의 사고 사망자가 발생한 건설현장의 안전관리를 위하여 CCTV 영상을 사람이 이해할 수 있는 설명문으로 자동 생성하는 이미지/비디오 캡셔닝(Captioning) 기술을 개발한다. 이미지/비디오 캡셔닝 분야에서는 지난 수년간 딥러닝을 활용한 접근법이 주류를 이루어왔다. 특히 트랜스포머(Transformer) 기반의 캡셔닝 모델들은 뛰어난 성능을 보여주며 대규모 데이터 셋에서 SOTA (State-of-The-Art)를 달성해왔다. 그러나 이러한 종단간 딥러닝 모델은 캡셔닝 과정이 블랙박스 형태로

* 동아방송예술대학교 뉴미디어콘텐츠과 부교수
- ORCID: http://orcid.org/0000-0002-2992-7294
** ㈜토탈시스
- ORCID: https://orcid.org/0000-0002-6232-476X
*** 국립한국교통대학교 컴퓨터공학과 교수(교신저자)
- ORCID: https://orcid.org/0000-0002-0110-7155

· Received: Nov. 03, 2025, Revised: Nov. 18, 2025: Accepted: Nov. 21, 2025
· Corresponding Author: Seokil Song
Dept. of Computer Engineering, Korea National University of Transportation, 50, Daehak-ro, Chungju-si, Chungcheongbuk-do, Republic of Korea
Tel.: +82-43-841-5349, Email: sisong@ut.ac.kr

이루어져 결과를 해석하거나 오류를 수정하기 어려우며, 종종 입력 영상에 존재하지 않는 사항을 언급하는 환각(Hallucination) 현상이 보고되고 있다[1]. 또한 거대 멀티모달 모델들은 수억~수천억개의 매개변수를 가진 거대한 아키텍처로서 실시간 응용이나 현장의 저가형 서버나 엣지(Edge) 디바이스 적용에는 부적합하고, 도메인 특화된 위험 표현을 학습시키기에도 한계가 있다[2]-[4].

본 논문에서는 인스턴스 세그멘테이션(Instance segmentation), 장면 그래프(Scene graph), 건설현장 도메인에 특화된 템플릿(Template) 기반 캡션 생성 방법을 활용하여 이러한 한계를 극복하는 시스템을 설계하고 구현한다. 본 논문에서 구현하는 시스템은 인스턴스 세그멘테이션 결과를 활용하여 장면 내 사람, 장비, 보호구 및 작업환경 등 요소를 식별하고, 이들 사이의 관계를 장면 그래프로 구성한다. 이후 장면 그래프를 기반으로 사전에 정의된 위험 유형을 인식하고, 해당 상황에 맞는 캡셔닝 템플릿을 선택한 후 이를 채워 넣음으로써 최종 캡션을 생성한다.

본 논문의 캡셔닝 방법은 정확한 인스턴스 세그멘테이션과 풍부한 건설현장 상황에 부합하는 사전정의된 템플릿 품질에 따라서 성능이 좌우된다. 제안하는 방법에서는 과거 5년간의 건설현장 사고사례 데이터, 건설현장에서 사용하는 위험성 평가 템플릿 문서, 각종 건설현장 사고관련 문헌을 자동으로 분석하여 풍부한 템플릿을 생성한다. 이렇게 실제 사례데이터나 건설현장에서 사용하는 문서들을 기반으로 만들어진 템플릿은 예측 결과의 신뢰성이 높고 불필요한 환각을 구조적으로 억제한다. 다만, 인스턴스 세그멘테이션의 오류가 환각으로 이어질 수는 있다.

하지만, 제안하는 방법은 캡셔닝 생성 단계별로 로그를 남기기 때문에 사용자는 캡션 생성의 근거를 명확히 추적할 수 있다. 이러한 측면에서 본 논문에서 제안하는 방법은 기존 딥러닝 캡셔닝과 차별화되며, 건설 안전이라는 특수 분야에 적합한 실시간성을 가지는 설명 가능하고 검증 가능한 방법이다.

II. 제안하는 캡셔닝 방법 설계 및 구현

2.1 제안하는 캡셔닝 방법

전체 시스템 구조는 그림 1과 같다. 객체 검출, 장면 그래프 생성, 템플릿 선택, 문장 생성, 로깅의 다섯 단계로 구성된다. 먼저 객체 검출 단계에서는 입력 영상의 각 프레임에 대한 인스턴스 세그멘테이션을 수행하여 사람, 장비, 구조물 등 객체들의 영역과 종류를 식별한다.

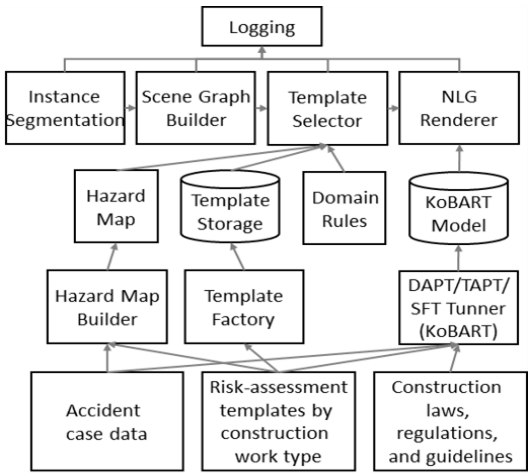

그림 1. 제안하는 건설현장 캡셔닝 시스템 구조
Fig. 1. Architecture of construction captioning system

제안하는 캡셔닝 방법은 객체 검출 지연시간이 전체 지연을 좌우한다. 이 논문에서는 YOLOv8 Seg 을 사용하였다[5]. 여러 인스턴스 세그멘테이션 모델을 분석해 본 결과 YOLOv8 Seg가 지연 대비 성능의 균형이 가장 적합하였다. 다만, 응용에 따라 정확성이 더 중요한 경우에는 다른 모델을 선택할 수 있다.

이어서 장면 그래프 구성 단계에서는 검출된 객체들을 노드로 하고, 객체 간의 공간적 관계(예: “위에(on)”, “옆에(next to)” 등)나 상호작용(예: “작업자가 크레인을 조작함”)을 엣지로 표현하는 그래프를 생성한다. 이를 통해 단순 객체 목록이 아니라 장면의 구조와 상황 맥락을 표현하게 된다.

표 1은 캡셔닝 전과정에 대해 생성된 로그를 보여준다. 이 표에서 “Scene graph“ 단계는 실제 생성된 장면 그래프 예시를 보여준다. 여러개의 노드와 엣지 들로 구성되며 노드는 객체 이름(name) 및 라벨(label), 폴리곤(poly), 바운딩박스(bbox) 등을 포함한다. 엣지는 두 노드간의 관계를 touching, far 등과 같이 표현한다.

다음으로 표 1의 ”Template selection“ 단계에서는 장면 그래프에서 추출한 정보와 사전 지식인 위험유형 사전(Hazard map)을 대조하여, 해당 장면에 적

용할 설명문 템플릿을 결정한다. 위험 유형 사전에는 다양한 재해 상황에 대한 키워드와 문장 패턴이 정의되어 있다. 예를 들어 “작업자가 안전모를 미착용하고 높은 곳에 있음”이라는 그래프 해석이 주어지면, 이는 추락 위험에 해당하므로 “추락 위험” 템플릿을 선택한다. 이 과정은 식 (1)과 같이 정리할 수 있다. 장면 그래프 G = (V, E) 가 주어질 때, 각 위험유형 h ∈ H 그리고 각 템플릿 집합 T_h, 장면 그래프에서 추출한 위험 단서 집합 R(G), 템플릿 T가 필요로하는 위험 단서 집합 K_T에 대하여, 해당 프레임에서 선택되는 템플릿은 다음의 규칙으로 정의가능하다.

표 1. 캡셔닝 단계별 로그
Table 1. Log by captioning stage

Stage	Log
Frame metadata	{ "frame": { "id": "frame_00000", "image": "vision/InstanceSegmentationData/fra "json": "vision/InstanceSegmentationData/fram "started_at": "2025-09-30T16:30:59", "W": 1920, "H": 1080, "finished_at": "2025-09-30T16:30:59" },
Hazard map	"ko_terms_alias_size": 2531, "class_alias": { "person": "근로자", "worker": "근로자", "scaffold": "비계", "horse_scaffolding": "말비계",
Scene graph	"graph": { "nodes": [{ "id": 0, "name": "support", "label": "동바리", "confidence": 0.96 "poly": ["edges": [{ "src": 0, "dst": 1, "rel": "touching", "dist_px": 9.0, "dist_m": 0.036084 "iou": 0.005444734
Template selection	"selection": { "selected": { "situation": { "template_id": "DOM_4be2fabab15b", "src_type": "domain", "hazard_type": "추락", "risk": { "template_id": "RISK_eab3319ce24a", "src_type": "risk", "hazard_type": "감전",
Captioning	"rendered": { "situation": "[0초] 현장 구역, 거푸집/동바리 작업 인지. 지 "risk": "[0초] 현장 상황 요약:\n- 공정/작업: 수배전설비\n-

$$T^* = \arg\max 1(R(G) \cap K_T \neq \varnothing) \quad (1)$$

템플릿은 Jinja2 문법으로 작성된 문장 틀로서 “현장에 {{작업자}}가 안전모 없이 작업하고 있어 {{위험유형}} 발생 가능”과 같이 변수 자리에 객체명이나 위험 유형이 들어갈 수 있게 되어있다. 문장 생성 단계에서는 선택된 템플릿에 실제 객체 명칭과 상황 정보를 채워 넣어 최종 한글 문장을 렌더링한다. 이때 객체 명칭은 앞서 정의된 국문 용어표에 따라 한글 표현으로 대체된다(예: “worker” : “근로자”, “scaffold” : “비계” 등). 최종적으로 공사 현장에 튜닝된 KoBART 모델을 통해 Jinja2로 채워진 틀문장을 조사/어미/어순 측면에서 자연스러운 한국어로 정제한다.

이 논문에서는 건설 현장 위험 데이터를 이용하여 KoBART를 파인튜닝 하였다. 학습 데이터는 템플릿에 따라 생성된 정답 문장과 실제 위험 상황 설명 문장을 포함한다. KoBART는 경량 모델이므로 지연시간이 몇 밀리초에 불과하여 전체 시스템의 실시간성을 저해하지 않는다.

지금까지의 캡셔닝 과정은 표 1에서와 같이 모두 로그로 기록된다. 구체적으로, 검출된 객체 리스트와 장면 그래프 구조, 선택된 템플릿 ID, 최종 생성된 상황 및 위험 유형 캡션을 JSON 형태로 기록하고, 주요 화면에는 생성 문장을 오버레이 하여 표시한다. 이 로그 데이터를 통해 운영자는 AI의 판단 근거를 사후에 검토할 수 있으며, 오탐지나 미검지 사례가 발견될 경우 어떤 단계에서 문제가 있었는지 역추적이 가능하다. 예를 들어 잘못된 캡션이 생성되었을 때, 로그를 보면 잘못 매칭된 템플릿이나 누락된 객체 등이 원인임을 파악할 수 있다.

표 1에서 “Hazard map”은 주요 사고 유형 각각에 대해서 관련이 있는 요소들을 매핑해 놓은 Hazard map을 이용해서 입력 프레임의 인스턴스들을 매핑한다. 예를 들어 “추락” 사고 유형은 “말비계”, “작업발판” 등과 같은 객체와 관련이 있음을 보인다. 이 내용은 기존에 확보한 사고사례 데이터를 분석하여 자동으로 생성된 것이다. 추가로 다양한 사고 관련 용어들을 통일하는 역할을 수행한다.

템플릿은 이미지 프레임에서 생성된 장면 그래프를 이용하여 캡셔닝을 수행하는 역할을 한다. 템플릿을 이용해서 작업자와 가까운 객체가 수배전 설비일 경우 그 거리에 따라 감전 위험을 캡션이 등을 할 수 있다. 또한, 작업자와 PPE(Personal Protective Equipment) 간의 관계에 따라 PPE 미착용 가능성을 캡셔닝 하게 되어있다.

이 논문의 템플릿은 건설 도메인 용어·객체 사전

을 구성하고, 과거 사고사례 데이터 및 위험성평가 문서에서 위험유형 - 행위 - 대상 표현을 규칙 및 빈도 기반으로 자동 추출한 뒤, 추출 결과를 Jinja2 템플릿의 변수 자리로 정리하여 자동 직렬화한다. 필요 시 문장 표면형(Surface form)은 표현 정규화만 수행하며, 위험유형 및 객체명은 템플릿 변수로 고정되어 캡셔닝 내용의 일탈을 구조적으로 억제한다. 이 과정은 오프라인으로 수행되어 실시간 추론 지연에는 영향을 주지 않는다.

2.2 제안하는 캡셔닝 방법 구현 및 성능평가

제안하는 캡셔닝 방법을 Python, Pytorch, jinja2, KoBART, YOLOv8 Seg를 이용하여 구현하고 실제 건설현장 이미지에 적용하여 캡셔닝을 수행하였다. 구현 및 실험은 MacBook Pro(M1, 8-core CPU/8-core GPU/16-core Neural Engine) 환경에서 수행하였다. 실험에 사용한 데이터는 유투브(Youtube)에서 수집한 거푸집 관련 작업을 하는 실제 영상이다.

실험을 통해 생성된 캡션의 정합성 및 캡셔닝 전체 파이프라인에 대한 지연시간을 측정하였다. 정합성은 두 가지 보조 지표로 기술한다. 첫째, 캡션이 실제로 인스턴스 세그멘테이션된 핵심 객체를 얼마나 충실히 반영하는지를 객체 - 문장 정합(명사 집합 일치도) 관점에서 확인하였다. 캡셔닝 결과는 문장 내 명시 매핑이 안정적으로 이루어져 높은 수준의 일치가 관찰되었다. 둘째, 선택된 위험 템플릿이 Hazard Map의 규칙과 일관되는지를 점검하였고, 모든 프레임에서 추출한 데이터가 위험 단서(말비계·동바리·거푸집 등)와 템플릿이 정합적으로 대응했다.

표 2는 전체 파이프라인의 지연시간을 측정한 것이다. 표에서 볼 수 있듯이 전체 수행시간의 88%는 인스턴스 세그멘테이션이 차지하였으며 장면그래프 생성, 템플릿선택 및 적용, KoBART 적용은 12% 에 불과 했다 전체 수행시간은 평균적으로 79.5ms 로 초당 10 프레임이상 처리가 가능한 성능이다.

표 2. 제안하는 방법의 지연시간 측정 결과
Table 2. Measured latency of the proposed method

Phase	Mean delay (ms)	Std. delay (ms)	Ratio (%)
Instance segmentation (YOLOv8-Seg)	70.2	8.5	88
Scene graph, Template selection, KoBART	9.3	2.7	12

III. 결 론

본 논문에서는 건설현장 CCTV 영상을 이용하여 상황과 위험 요소를 캡셔닝하는 시스템을 설계하고 구현하였다. 제안하는 시스템은 인스턴스 세그멘테이션, 장면 그래프, 도메인 지식과 조건부 템플릿을 기반으로 캡셔닝을 수행하는 경량 모델이다. 적은 비용으로 건설 현장에 직접 설치가 가능하며 실시간/준실시간으로 동작이 가능하다. 기존 거대 멀티모달 모델에 비해 환각에 대한 우려가 적으며 문제 발생 시 추적이 가능하다. 하지만, 제안하는 방법은 구조적으로 환각을 억제하고 있지만, 인스턴스 세그멘테이션의 인식 오류가 캡셔닝 오류로 이어질 수 있다. 또한, 사전에 정의된 템플릿 에 존재하지 않는 상황의 경우에는 캡셔닝을 할 수 없는 한계가 있다.

References

[1] A. Rohrbach, et al., "Object hallucination in image captioning", Proc. 2018 Conf. on EMNLP, Brussels, Belgium, pp. 4035-4045, Oct.-Nov. 2018. https://doi.org/10.18653/v1/D18-1437.

[2] A. Sharshar, et al.,, "Vision-language models for edge networks: a comprehensive survey", IEEE Internet of Things Journal, Vol. 12, No. 16, pp. 32701-32724, Aug. 2025. https://doi.org/10.1109/JIOT.2025.3579032.

[3] W. Chai, et al., "AuroraCap: efficient, performant video detailed captioning and a new benchmark", Proc. 13th Int. Conf. on Learning Representations (ICLR), Singapore, Apr. 2025.

[4] K. Zhou, et al.,, "Multimodal situational safety", Proc. 13th Int. Conf. on Learning Representations (ICLR), Singapore, Apr. 2025.

[5] R. Bai, M. Wang, Z. Zhang, J. Lu, and F. Shen, "Automated construction site monitoring based on improved YOLOv8-Seg instance segmentation algorithm", IEEE Access, Vol. 11, pp. 139082-139096, Dec. 2023. https://doi.org/10.1109/ACCESS.2023.3340895.