SEMANTIC AND GOAL-ORIENTED SIGNAL PROCESSING: SEMANTIC EXTRACTION

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

MASTER OF SCIENCE

IN

ELECTRICAL AND ELECTRONICS ENGINEERING

By

Mehmetcan Gök

August 2022

Semantic and Goal-oriented Signal Processing: Semantic Extraction
By Mehmetcan Gök
August 2022

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Orhan Arıkan(Advisor)

Tolga Mete Duman

Serkan Sarıtaş

Approved for the Graduate School of Engineering and Science:

Orhan Arıkan
Director of the Graduate School

ABSTRACT

SEMANTIC AND GOAL-ORIENTED SIGNAL PROCESSING: SEMANTIC EXTRACTION

Mehmetcan Gök
M.S. in Electrical and Electronics Engineering
Advisor: Orhan Arıkan
August 2022

Advances in machine learning technology have enabled real-time extraction of semantic information in signals, which has the potential to revolutionize signal processing techniques and drastically improve their performance for next-generation applications. A graph-based semantic language and a goal-oriented semantic signal processing framework are adopted for structured and universal representation and efficient processing of semantic information. In the adopted framework, preprocessing of input signals is followed by a semantic extractor which identifies components from a set of application-specific predefined classes where the states, actions, and relations among the identified components are described by another application-specific predefined set called predicates. For additional information, the resulting semantic graph is also embedded with a hierarchical set of attributes. In this thesis, we focus on the crucial semantic extractor block, and to illustrate the proposed framework's applicability, we present a real-time computer vision application on video-stream data where we adopt a tracking by detection paradigm for the identification of semantic components. Next, we show that with the adopted semantic representation and goal-filtering, the semantic signal processing framework can achieve an extremely high reduction in data rates compared to traditional approaches. Finally, we demonstrate a way to identify points of significant innovation over extended periods of time by tracking the evolution of multi-level attributes and discussing future research directions.

Keywords: semantic signal processing, goal-oriented signal processing, semantic extraction, graph-based languages.

ÖZET

ANLAMSAL VE HEDEFE YÖNELİK SİNYAL İŞLEME: ANLAMSAL ÇIKARMA

Mehmetcan Gök
Elektrik ve Elektronik Mühendisliği, Yüksek Lisans
Tez Danışmanı: Orhan Arıkan
Ağustos 2022

Makine öğrenimi teknolojisindeki ilerlemeler, sinyal işleme tekniklerinde devrim yaratma ve yeni nesil uygulamalar için performanslarını büyük ölçüde iyileştirme potansiyeline sahip olan sinyallerdeki anlamsal bilgilerin gerçek zamanlı olarak çıkarılmasını sağlamıştır. Semantik bilginin yapılandırılmış ve evrensel temsili ve verimli işlenmesi için grafik tabanlı bir semantik dil ve hedef odaklı bir semantik sinyal işleme yapısı benimsenmiştir. Benimsenilen yapıda, sinyal girdilerinin ön işlenmesine bir anlamsal çıkarıcı eşlik etmektedir. Bu anlamsal çıkarıcı, anlamsal bileşenlerin tanımlanması, tanımlanmış bileşenler arasındaki durumların, eylemlerin ve ilişkilerin uygulamaya özel önceden tanımlanmış bir küme içinden seçilmesini sağlar. Ek bilgi için, elde edilen anlamsal grafik çıktısına ayrıca hiyerarşik bir dizi öznitelikle gömülmüştür. Bu tezde, önemli semantik çıkarıcı bloğuna odaklanıyoruz ve önerilen yapının uygulanabilirliğini göstermek için video akışı verileri üzerinde gerçek zamanlı bir görüntü işleme uygulaması sunuyoruz. Daha sonra, benimsenen anlamsal temsil ve hedef odaklı filtreleme ile sinyal işleme yapısının geleneksel yaklaşımlara kıyasla veri hızlarında son derece yüksek azalma sağlayabileceğini gösteriyoruz. Son olarak, çok seviyeli özniteliklerin gelişimini izleyerek uzun zaman dilimlerinde önemli yenilik noktalarını belirlemenin bir yolunu gösteriyoruz ve ileriki araştırma konularını tartışıyoruz.

Anahtar sözcükler: anlamsal sinyal işleme, hedef odaklı sinyal işleme, anlamsal çıkarım, grafik tabanlı diller.

Acknowledgement

First, I would like to thank Prof. Orhan Arkan for his wise supervision and support during my M.S. studies. I absolutely appreciate how understanding and encouraging he is. Thanks to him I was able to conclude this thesis. I wish for a fruitful and neverending collaboration through the years.

I would like to convey my sincere gratitude to Prof. Tolga Mete Duman and Prof. Serkan Sarıtaş, who served on my examination committee.

I would like to thank Prof. Tolga Mete Duman, Mert Kalfa, Arda Atalık, Sadık Yağız Yetim and Büşra Tegin from the Huawei R&D Research Group at Bilkent for their ongoing support and advices through my studies. Our insightful and informative discussions greatly aided me in organizing my thesis.

From the Bilkent University, I thank Prof. Sinan Gezici, Prof. Serdar Yüksel, Prof. Muhammed Ömer Sayın and all my professors for their contributions to my undergraduate and graduate years through their courses.

I also thank Elif Bilge (and her overweight grumpy old cat Mevlüde Boncuk) for being the greatest of allies in my life against faced many hardships. I have been extremely fortunate to have her continuous companionship and support. I also would like to thank my gang from Gaziemir for our amusing late-night chats and adventures.

Last but not least, I want to thank and express my deepest gratitude to my family for their unending support, encouragement and shared life experiences throughout my life. I owe everything to them.

Contents

List of Figures

List of Tables

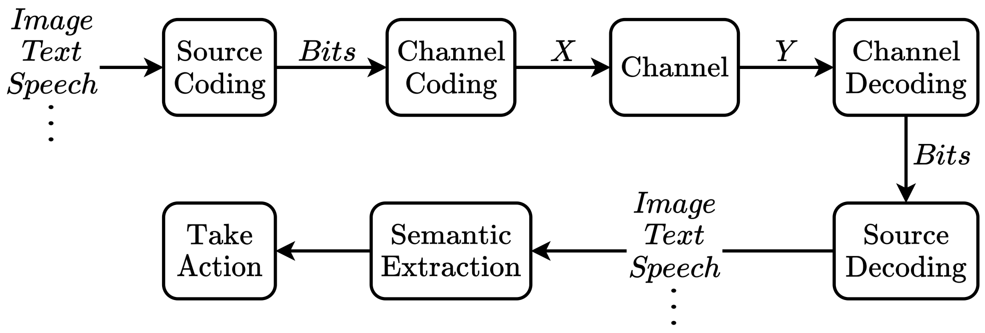
Chapter 1

Introduction

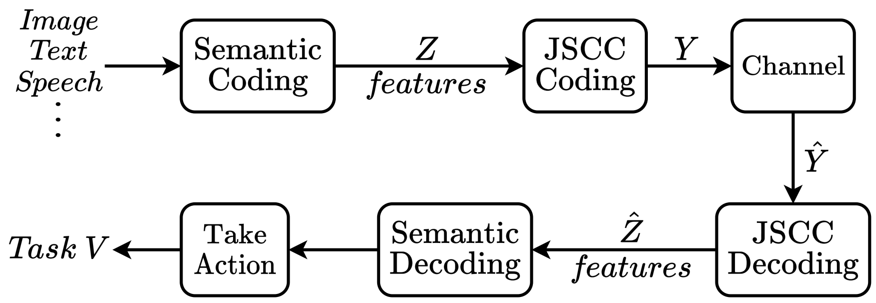
Signal processing has played a major role in the era of the digital revolution. It has been essential in the design and development of complex systems that integrate the contributions of multiple scientific disciplines. Significant challenges in critical areas including remote sensing, healthcare, finance, transportation, recommendation systems, Human-machine symbiosis, defense, virtual reality (VR), augmented reality (AR) and security have been successfully overcome with advances in signal processing techniques [1].

Semantic information theory (SIT) [2] has been around almost as long as the classical information theory (CIT) [3]. As its name implies, SIT is focused on the *semantic problem* which is the accurate generation, processing, storage, and transmission of the *intended meaning* rather than the transmission and recovery of individual bits and symbols referred as the *technical problem.* Comparison of traditional and semantic communication systems is illustrated in Fig. 1.1.

Despite its long history, semantic information has only been employed seldom and at a low level in various signal processing applications. Target tracking by an individual sensor or a sensor network is an important example of semantic information use. In radar signal processing, for example, following the detection of targets via real-time processing of radar returns, only a small portion of detected



(a) Traditional communication system.



(b) Semantic communication system.

Figure 1.1: Comparison of traditional and semantic communication systems.

target reports are usually transmitted to the tracking system to initiate new tracks or maintain the existing ones. Targets that move in close proximity can be detected and monitored as a group in such processes. The detection and tracking of targets with their indicated group features may be considered as a semantic extraction of information from radar returns in this application. Furthermore, if multiple radars are linked together in a network, they may share semantic information in the form of target tracks for better surveillance and target handover operations [4, 5].

Recent developments in machine learning have far-reaching consequences in many fields, including but not limited to healthcare [6, 7, 8], finance [9, 10, 11], entertainment [12, 13], communications [14, 15], and defence [16]. Among the many accomplishments of machine learning technology, real-time extraction of rich semantic information from streaming data is crucial in future signal processing applications. For example, recognition of objects in images and videos may be done in real-time [17, 18, 19]. The availability of such rich semantic data can drastically alter and improve signal processing techniques.

The 6G and beyond communication systems have the potential to bring about an *Internet of Intelligence* with connected people, connected things, and connected intelligence [20, 21, 22, 23, 24]. It will possess the potential to build up a wireless network of *everything sensing*, *everything connected* and *everything intelligent*. Aside from traditional radio technical advancements, 6G and beyond will investigate several novel architectures and technologies, with goal-oriented semantic signal processing and communications being a key component. Semantic communications will enable more efficient use of network bandwidth. Furthermore, network users such as self-driving vehicles will profit greatly from goal-filtered semantic data exchanges between themselves and the smart city backbone.

In this thesis, for a structured and universal representation and efficient processing of the semantic information, first we re-introduce the graph-based semantic language and the goal-oriented semantic signal processing framework proposed in our previous work [25], in which the proposed formal semantic signal processing framework incorporates a goal-oriented parsing method and can easily be tailored for specific signal processing applications and goals. A semantic information extractor in the proposed framework identifies signal components and semantic relation into a set of application-specific classes and predicates respectively. Moreover, along with the identification of components and predicates, each node in the graph has a hierarchical collection of attributes that are organized in increasing levels of complexity and offer additional detailed information. Those components that are semantically related to each other form directed bipartite graphs where only edges between a component and a predicate are allowed to exist. Since these graphs often have a very small number of nodes, operations on them can be carried out very effectively. Additionally, the proposed bipartite structure permits a comprehensive yet relatively straightforward representation of signals and significantly reduces the complexity of graph-based signal processing applications.

In the proposed semantic signal processing framework, the graphs that will be processed further and those that are currently not of interest can be grouped based on an internally or externally defined set of goals that can change over time. Spatio-temporal tracking of graph parameters and performing various operations on their attributes may be included in the later processing stages. The desired level of semantic information contained in the graphs that are of interest can be locally kept or exchanged with another processor through the appropriate communication protocols at any time in the processing chain. Since interesting events typically occur infrequently in high bandwidth sensor data, the corresponding semantic signal processing algorithms produce astounding compression rates. Here, we focus on the crucial semantic extractor block of the framework. We investigate and propose methods for communication over a sensor network where sensor data is semantically processed and semantic information is transmitted in the network to illustrate the significant reduction in communication rate compared to traditional ways. To demonstrate the wide range of applicability of the proposed goal-oriented semantic signal processing framework, we experiment on a real-time video stream use-case and provide details on how the proposed framework can be adapted for different signal modalities. Furthermore, we present a way to characterize and identify innovations across frames at various attribute-levels. In the next section, we review existing approaches to semantic information in signals.

1.1 Semantic Information

In his seminal paper [3], Claude E. Shannon introduced CIT and paved the way for modern communications as we know it. However, as Shannon and Weaver later formalized [26], CIT only addresses the low-level technical problem of accurate encoding and transmission of bits and symbols. The semantic and effectiveness problem at higher levels of abstraction are concerned with the transmission of the intended meaning and the execution of the desired effect respectively, where in this thesis we focus on the mid-level semantic problem.

Consider the conversation between two people. We utilize words as part of shared language to convey certain information to one another. If person-A describes a visual scene to person-B, the precise image that reconstructed in person-B's mind is quite certainly not the one that in person-A's mind. However, no semantic information is lost during transmission if the intended information is accurately delivered. Furthermore, in terms of throughput, a complex scene with a few meaningful components may be represented with a comparable number of words significantly reducing the amount of symbols transmitted compared to a pixel-by-pixel decomposition of the image. Another crucial aspect of human communication is the ability to ask questions, which significantly limits the domain knowledge when transmitting information. In this thesis, we refer to questions that can be asked internally or externally as goals. As a result, throughout the thesis, the term goal-oriented information refers to semantic material that has been filtered by pertinent queries.

With recent developments in Internet of Things (IoT), massive-machine type communications (mMTC) will dominate next generation communication networks [21]. Using a language as a medium to describe signals and construct inquiries about them enables a human-like way of communication between machines, which can significantly reduce the overall throughput while increasing efficiency. Despite this increased efficiency has been widely anticipated and has been known virtually from the introduction of CIT [26, 2], efficient extraction of semantic information was not achievable until recent advancements in machine learning (ML) and artificial intelligence (AI) technology.

ML and AI approaches enable the development of unique signal processing algorithms and systems capable of extracting and using semantic information in real-time inputs by allowing for the efficient extraction of semantic information from signals of varying modality, such as speech, image, and video signals. The generated semantic signals enable the next generation of signal processing techniques to be applied at a semantic level in real-time using a goal-oriented approach.

In contrast to traditional signal processing methodologies, goals in semantic signal processing are specified in a semantic language. To be able to establish suitable performance metrics and apply compression/coding methods based on the communicated or inferred meaning of transmission, a language that maps these meanings to a predetermined syntactic structure must be determined. As a result, it is vital to develop a semantic information and language model that is generic enough to be used in a wide range of signal processing applications while being simple enough to be used by low-end agents with limited power and computational capabilities. The remainder of this section provides a quick overview of previously suggested semantic language modalities.

1.1.1 Natural Languages

Natural Languages (NL) are the most popular and obvious alternatives for a shared language in signal processing and mMTC applications. Natural Language Processing (NLP) has generated a large amount of research for the synthesis of NL sentences given an input signal, particularly for human-machine interaction applications including visual question answering, image/video captioning, and discourse parsing. Relatively recently, there has been an increasing focus on semantic communications using NL as a basis as well [27, 28, 29, 30].

NLs are appealing because they provide a common language for all agents, independent of their individual capabilities. However, this universality entails the processing of a huge knowledge base (e.g., the English language), which has inherent ambiguities and contradictions. Consequently, the NL techniques are overly complicated for simple IoT sensors and similar machine-type applications. In such instances, a clear and straightforward linguistic structure is essential.

1.1.2 Propositional Logic

Carnap and Bar Hillel's pioneering work [2] on semantic communications provides propositional logic as a semantic language. Common logical operations such as NOT ($\neg$), AND ($\wedge$), OR ($\vee$), IMPLIES ($\Rightarrow$) etc., are used to combine several symbols to form molecular sentences describing a state (e.g., $\neg(A \Rightarrow B) \wedge C$). Goals in this language can be formed similarly using propositional logic, effectively parsing the existing truth table of symbols for the desired pattern. Specifically, based on logical probabilities, the authors measured semantic information as the degree of confirmation:

$$H(\mathcal{H}, e) = -\log c(\mathcal{H}, e) \tag{1.1}$$

where $\mathcal{H}$ is hypothesis, e is evidence, and $c(\mathcal{H}, e)$ is the degree of confirmation of the hypothesis and the evidence, e.g, $\mathcal{H}$ may be new message and e may refer to knowledge. In [31], the semantic entropy of a sentence s and its logical probability defined as:

$$H(s) = -\log_2 m(s), \tag{1.2}$$

$$m(s) = \frac{p(\mathcal{W}_s)}{p(\mathcal{W})} = \frac{\sum_{w\in\mathcal{W}, w\models s} p(w)}{\sum_{w\in\mathcal{W}} p(w)} \tag{1.3}$$

where $\models$ is proposition of satisfaction, $\mathcal{W}$ is the symbol set of the conventional source, $\mathcal{W}_s = \{w \in \mathcal{W} : w \models s\}$ is the set of symbols which s is true, and $p(w)$ is the probability of w. Furthermore, in [32], the authors introduced the concept of matching degree with the membership degree (i.e., a concept in fuzzy set theory which is difficult to measure analytically) to define semantic entropy in fuzzy systems. In [32], the membership degree is defined manually by following expert intuition and experience. Specifically, for semantic concept ζ (e.g., transmission of task) and membership degree $\mu_\zeta(X)$ for each $X \in \mathcal{X}$, the matching degree for the class C_j is defined as:

$$D_j(\zeta) = \frac{\sum_{X\in\mathcal{X}_{C_j}} \mu_\zeta(X)}{\sum_{X\in\mathcal{X}} \mu_\zeta(X)} \tag{1.4}$$

which characterizes the semantic entropy of X on concept ζ. Furthermore, with D_j, the semantic entropy on class C_j defined as:

$$H_{C_j}(\zeta) = -D_j(\zeta)\log_2 D_j(\zeta) \tag{1.5}$$

It is important to note that, the definitions above are based on the assumption that there exist a way to measure and quantify semantic information [33].

The propositional logic as a semantic language is attractive, since it can be tailored for specific applications and as a result, may not suffer from the complexity and ambiguity of NLs. However, it is challenging to incorporate numerical attributes such as position and velocity into the language; therefore, the resulting descriptions may lack complete information about the signals of interest.

1.1.3 Graph Based Languages

Graph-based languages as mathematical constructions have major benefits over NLs due to their ability to describe components in the signal along with the inherent connections and states. Most popular applications of graph-based languages include scene graph generations from images and videos [34, 35, 36, 37, 38, 39, 40, 41], knowledge graphs and graph-based question answering [42, 43, 44, 45], and semantic web applications [46, 47, 48]. The question answering problems use sub-graphs called motifs to search for a pattern within a graph representation, enabling a goal-oriented approach [49, 50, 51, 52]. Moreover, graph nodes and edges can include additional attributes to convey a more complete description of the underlying scene [53].

We believe that graphs with attributes can give a structured and complete description for a wide range of signals due to the aforementioned appealing properties. However, because the bulk of graph-based language models continue to rely on NLs as their foundation, they might be overly complicated for basic machine-type applications with strict efficiency requirements. Therefore, we recommend the utilization of a graph-based language with a unique and relatively modest knowledge base that can be adapted to specific signal domains and applications of interest, which would be extremely useful for machine-type applications.

1.2 Organization of the Thesis

The rest of the thesis is organized as follows. In the next chapter, we present a detailed literature survey on the recently developed semantic information extraction and semantic communication techniques. The proposed goal-oriented semantic language and signal processing framework in [25] is re-introduced in Chapter 3. In Chapter 4, we first demonstrate the proposed semantic signal processing framework on video-stream signals and discuss customizations for adaptation of proposed framework on other signal modalities and applications. Then, we investigate the data compression potential of the proposed framework and introduce a way to identify innovations over the extended periods of time using the associated attributes at multiple levels. Finally, we conclude the thesis and discuss the future directions in Chapter 5.

Chapter 2

A Review of Semantic Transformations and Goal-Oriented Semantic Communications

In this chapter, existing semantic transformations that can effectively process signals of different modalities, and current state of the goal-oriented semantic communications are described. Here, we provide detailed survey of the state-of-the-art semantic transformations.

Semantic information exists in many signal modalities such as a textual description of an image, a knowledge graph derived from a paragraph, and even in correlation functions of random processes. We refer to semantic transformation or semantic extraction as the mapping from an input modality to a target semantic modality where the target semantic modality can be anything ranging from vectors, texts, and graphs. It is important to point out that although there are many input and target semantic modality pairs, we only discuss a small, albeit an intuitive subset of transformation modalities mostly focused on visual input types.

2.1 Object Detection

Object detection can be interpreted as one of the core semantic transformations from visual domain to domain of object classes. Convolutional Neural Networks (CNNs) [54] constitute fundamental processing modules of object detection methods. Recurrent CNN (RCNN) models are introduced in [55], where the selective search is utilized to propose candidate regions and each region is processed independently by a CNN and classified using support vector machines (SVMs). The use of RCNN models is further diversified in [56, 17]. In [56] instead of extracting region features separately, a single forward pass through a CNN is performed and the resulting feature map is branched into regions using Region-of-Interest (RoI) pooling. RCNNs with real-time processing capabilities are introduced in [17], where object regions are proposed by Region Proposal Networks (RPNs) instead of selective search. YOLO is introduced in [57] as the very first attempt on fast object detection task and it is further improved in [58, 59]. Unlike Faster-RCNN [17], YOLO, and its later versions, do not utilize region proposals. Instead, they split the input image into cells and perform inference on a limited number of boxes in each cell [57] as shown in Fig. 2.1. Moreover, recent works [18, 19] use network scaling approaches to achieve higher detection accuracy within shorter inference times, and obtain state-of-the-art results for real-time object detection. Particularly, [18] proposes a weighted bi-directional feature pyramid network to fuse features from multiple levels, and in [19] a network scaling method is applied to the YOLO architecture. Performance, architecture, input size and speed of various object detection models are reported in Table 2.1.

2.2 Semantic Segmentation

Segmentation is another semantic transformation that is applicable for visual inputs or inputs that are transformed to 2-D domains such as time-frequency or time-scale. Semantic segmentation can be considered as a higher resolution version of object detection where a category label is assigned to each pixel. More

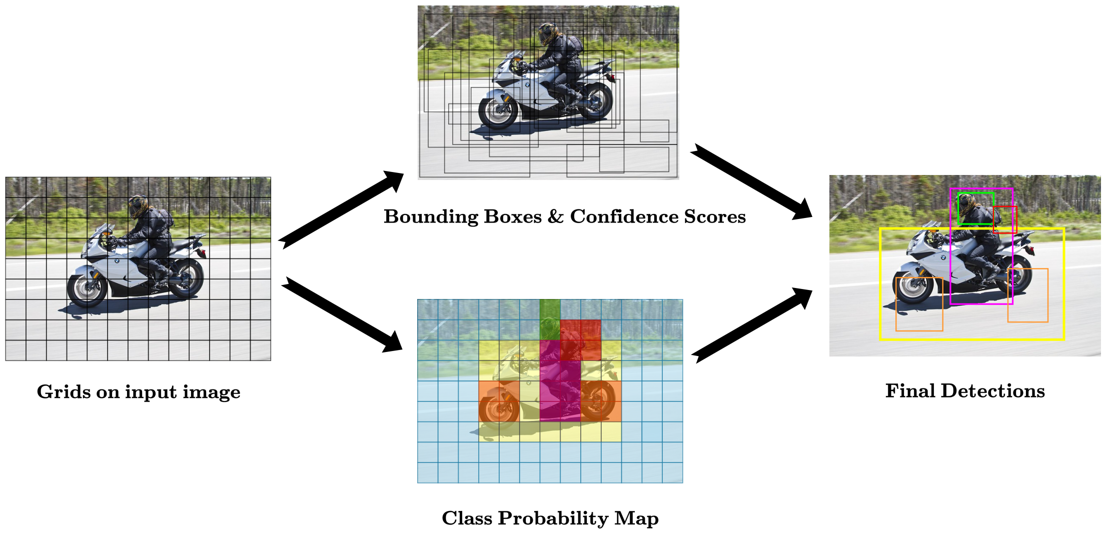


Figure 2.1: YOLO [57] object detection model. Unlike Faster-RCNN [17], YOLO does not use region proposals, instead, it splits the input image into grids and performs inference on each grid.

formally, semantic segmentation aims to find a way to assign a label from categories to a set of random variables that correspond to each pixel in the image. Here, each label can represent an object class such as “person”, “plane”, “car”, etc., or labels can be the distinct but unspecified clusters in an unsupervised setting [61, 62, 63]. In [64], texton forests are proposed as efficient low-level features for image segmentation. Alternative approaches include random forest classifiers [65] and combination of SVMs and Markov Random Fields (MRFs) [66]. The state-of-the-art in semantic segmentation typically employs convolutional architectures in supervised, semi-supervised, and weakly-supervised settings [67]. It is important to note that, the features extracted by the deeper layers of a CNN are more concentrated on concise semantics with low spatial details whereas shallow layers of a CNN are more aware of spatial details such as edge orientations. Convolutional architectures for image segmentation include dilated convolutions [68] to incorporate context information from multiple scales, Fully Convolutional Network (FCN) architectures [69] using skip-connections [70], a symmetrical encoder-decoder structure called DeconvNet [71], and its simpler version in [72].

Recurrent architectures are also employed in semantic segmentation. In [73], Recurrent Fully Convolutional Networks are proposed for multi-slice Magnetic

Table 2.1: Performance comparison of various object detection models on MS COCO dataset. mAP@0.5 is abbreviation for mean average precision (AP), i.e., the area under precision-recall curve, over all object classes at 0.5 IoU threshold. Models are classified as real-time detectors if they operate faster than 30 FPS.

Model	**Backbone**	**Size**	**mAP@0.5**	**FPS**
RCNN [55]	AlexNet	224	58.50%	∼0.02
Fast-RCNN [56]	VGG-16	Variable	65.70%	∼0.43
Faster-RCNN [17]	VGG-16	600	67.00%	5
Mask-RCNN [60]	ResNeXt-101-FPN	800	62.35%	10.6
YOLO [57]	GoogLeNet	448	57.90%	45
YOLO9000 [58]	DarkNet-19	352	44.00%	81
YOLOv3 [59]	DarkNet-53	320	51.50%	45
YOLOv4 [19]	CSPDarkNet-53	512	64.90%	31
EfficentDet-D2 [18]	Efficient-B2	768	62.30%	39.7

Resonance Imaging (MRI) segmentation where Gated Recurrent Unit (GRU) is incorporated into the bottleneck of the U-Net architecture given in [70]. Moreover, several Generative Adversarial Network (GAN) architectures are proposed [74, 75, 76], where adversarial training is introduced to semantic segmentation in [74].

Semantic segmentation architectures that offer real-time operation are proposed in [77, 78, 79]. In [60], Faster-RCNN [17] is modified for instance segmentation and Mask-RCNN is introduced, the process of which is illustrated in Fig. 2.2. Unlike semantic segmentation, instance segmentation aims at assigning labels to pixels at an object level as opposed to a class level. Union of instance and semantic level segmentation is called panoptic segmentation [80], where each pixel is associated with both instance and class level labels, as shown in Fig. 2.3 along with different segmentation types. Readers interested in semantic segmentation can find detailed taxonomy of models, applications, discussion on challenges, and possible future directions in [67, 81, 82].

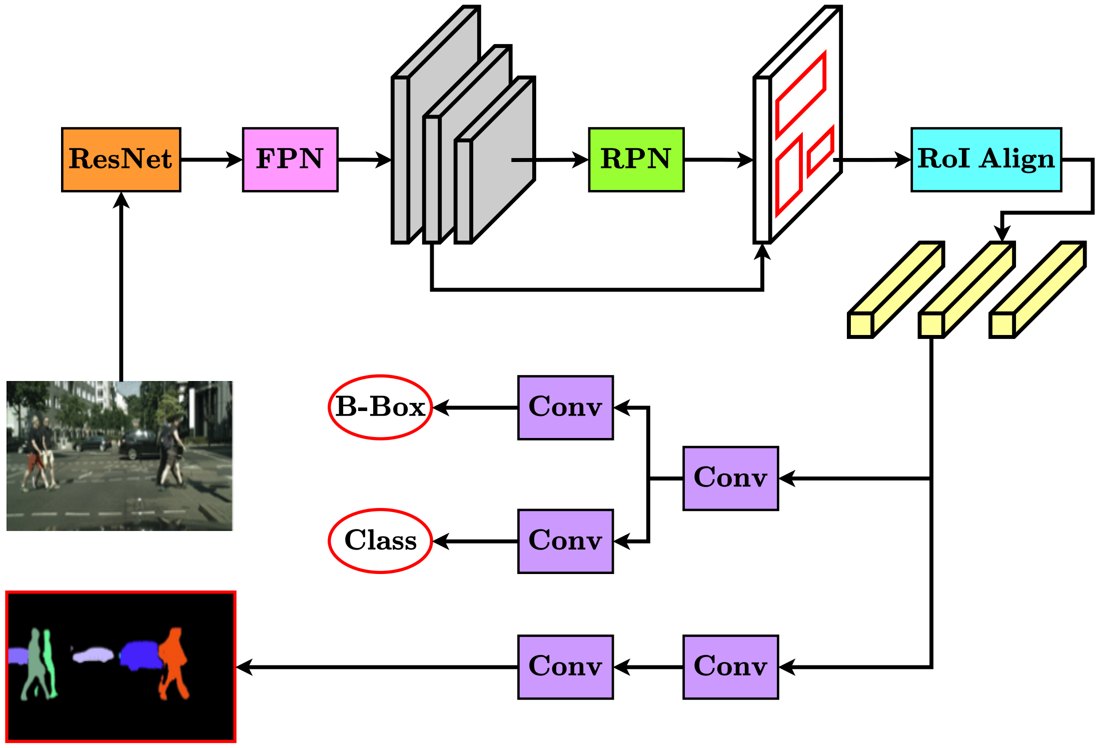
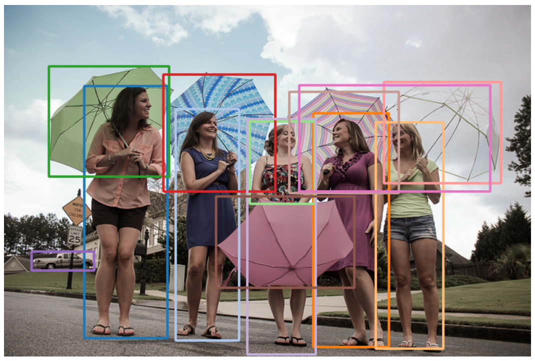


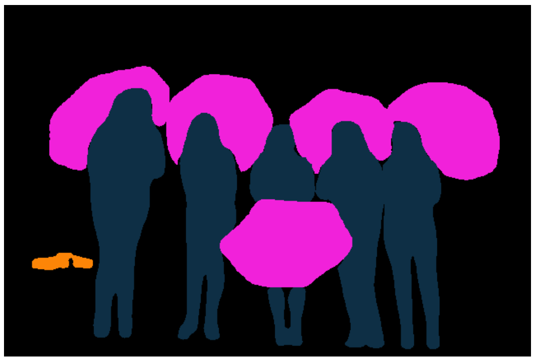
Figure 2.2: Mask-RCNN [60] model for instance segmentation. As its backbone, Mask-RCNN uses ResNet which is followed by a Feature Pyramid Network (FPN) and a Region Proposal Network (RPN). Features for proposed regions are extracted with RoI Alignment. Finally, bounding-box regression, instance classification, and segmentation mask inference are performed.

2.3 Image and Video Captioning

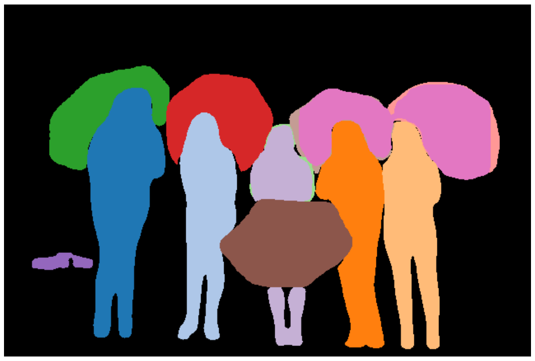
An intuitive way of representing the semantic information embedded in images or videos is to describe them via natural languages (NLs). Image caption or annotation generation is defined as the process of generating textual descriptions for images, which include not only the descriptions of objects within frames but also their interrelations and states. Typical fundamental building blocks of captioning models include CNNs and Recurrent Neural Networks (RNNs). More specifically, a CNN backbone is utilized to extract the visual features and RNNs are used for sequence modeling [83, 84, 85, 86, 87, 88]. To improve the caption quality, visual attention on CNNs [87] or captioning on multiple image regions are proposed [84]. In [88], dense captioning is introduced where object detection



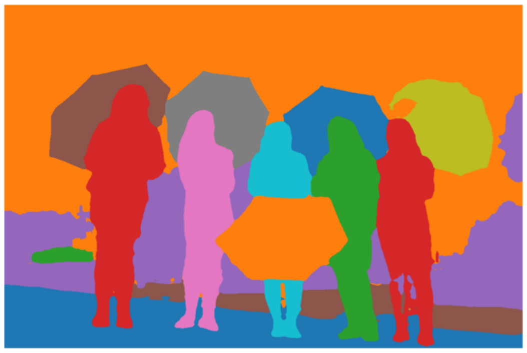
(a) Object Detection



(b) Semantic Segmentation



(c) Instance Segmentation



(d) Panoptic Segmentation

Figure 2.3: Different object detection and segmentation techniques applied to an image. (a) In object detection, detected/classified objects are localized with bounding boxes. (b) Semantic segmentation aims to assign class labels at the pixel level. (c) Instance segmentation is applied to the object detection for finer localization. (d) In panoptic segmentation, both background and foreground are segmented such that each pixel is associated with an instance and a class simultaneously.

and caption generation tasks are tackled jointly in such a way that the detected visual concepts are described with short NL phrases.

To overcome localization issues of visual concepts due to overlapping target regions, global image features are fused with region features for dense captioning in [89]. In [90], the dense captioning task is extended to relational captioning where multiple captions are generated for each object pair based on spatial, attentive, and contact relation information [91]. Examples of image captioning techniques are illustrated in Fig. 2.4.

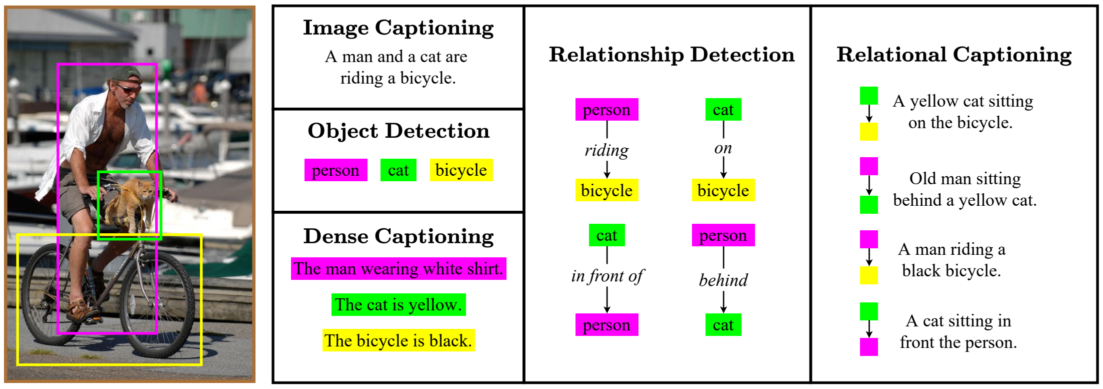


Figure 2.4: Some semantic transformation examples on the image domain. Image captioning simply generates a textual description of the image. Object detection identifies the objects residing in the image. Dense captioning generates a caption for each salient region or detected object in the image. Relationship detection identifies relations among each object pair. Relational captioning generates a caption for each detected object pair.

Video captioning can be considered as a temporal extension of image captioning. Just like image captioning, the video captioning architectures also use CNNs (2D or 3D) as attention mechanisms or to extract visual semantic content. Then, RNNs or LSTMs are typically used to generate NL text sequences [92] as illustrated in Fig. 2.5. Specifically, one of the early works [93], which is only applicable for videos of short duration, employs mean-pooling to frame representations extracted by a shared CNN and utilizes an LSTM architecture for caption generation. To extend the validity of extracted features to longer durations, recurrent visual encoder architectures are used [83, 94, 95]. In [96], instead of a single caption, multiple captions are generated without their temporal localizations using a hierarchical-RNN architecture. Furthermore, a hierarchical-RNN architecture is used to generate paragraphs for videos in [97].

Dense captioning works in the image domain are also extended to video signals in [98] where multiple captions are generated for each detected event, and they are temporally localized. Dense video captioning is also referred to as joint event detection and description generation. In [98] CD3 features are extracted beforehand, and they are fed into a proposal module that uses an attention mechanism. JEDDi-Net proposed in [99] is employed for dense video captioning in an end-to-end fashion without pre-feature extraction. It uses a 3D-CNN to extract the video features and a segment-proposal-network (SPN) to generate candidate segments for events. A comprehensive survey of image and video captioning models can be found in [100] and [92], respectively.

2.4 Scene Graph Generation

A powerful form of semantic transformation is to convert images into graphs that represent a scene and encode the visual relationships presented in the image. Scene graphs are proposed in [101] describe image features and object relationships in an explicit and structured way for image retrieval. Scene graph generation models can capture a higher-level of understanding of the scenes compared to object detection models by additionally identifying object attributes and relationships among them. In essence, scene graph generator models can be considered as image captioning models where they produce parsed versions of succinct captions that are represented in the graph format, instead of NL sentences. Specifically, a scene graph is a graphical data structure that describes the contents of a scene where the nodes represent the detected objects, and edges linking them represent the inter-node relationships. An example scene graph is shown in Fig. 2.6.

Scene graph can be described as a projection of given scene, e.g., image, video or a 3D mesh, into a directed graph constituted by set of visual triplet, usually describe as *(object, predicate, subject)*, descriptions $\mathcal{G}$ defined over the product of predetermined object $\mathcal{O}$, attribute $\mathcal{A}$ and predicate set $\mathcal{P}$. In other words, $\mathcal{G} \subseteq \mathcal{O} \times \mathcal{P} \times (\mathcal{O} \cup \mathcal{A})$. Generally, the predicate set $\mathcal{P}$ contains the *is* predicate p_{is} to express the cases where only one object is involved. In the following we will use superscript S to denote belonging of sets and variables to a particular scene S. For instance, while $\mathcal{O}$ contains all possible object types such as *car* or *person* including their location, $\mathcal{O}^S$ will denote the set objects that are present in the scene S, e.g. the set that contains *the people in a landscape photo*. Each object in scene graph $\mathcal{G}^S$ indexed by k and is expressed by a semantic label c and a bounding box b as a location marker such that $o_k^S = \left(c_k^S, b_k^S\right) \in \mathcal{O}^S$ for $k \in 1, \ldots, \left|\mathcal{O}^S\right|$. For example, a particular *car* in the *right corner* of a parking lot image S, we have $c_k^S = car$

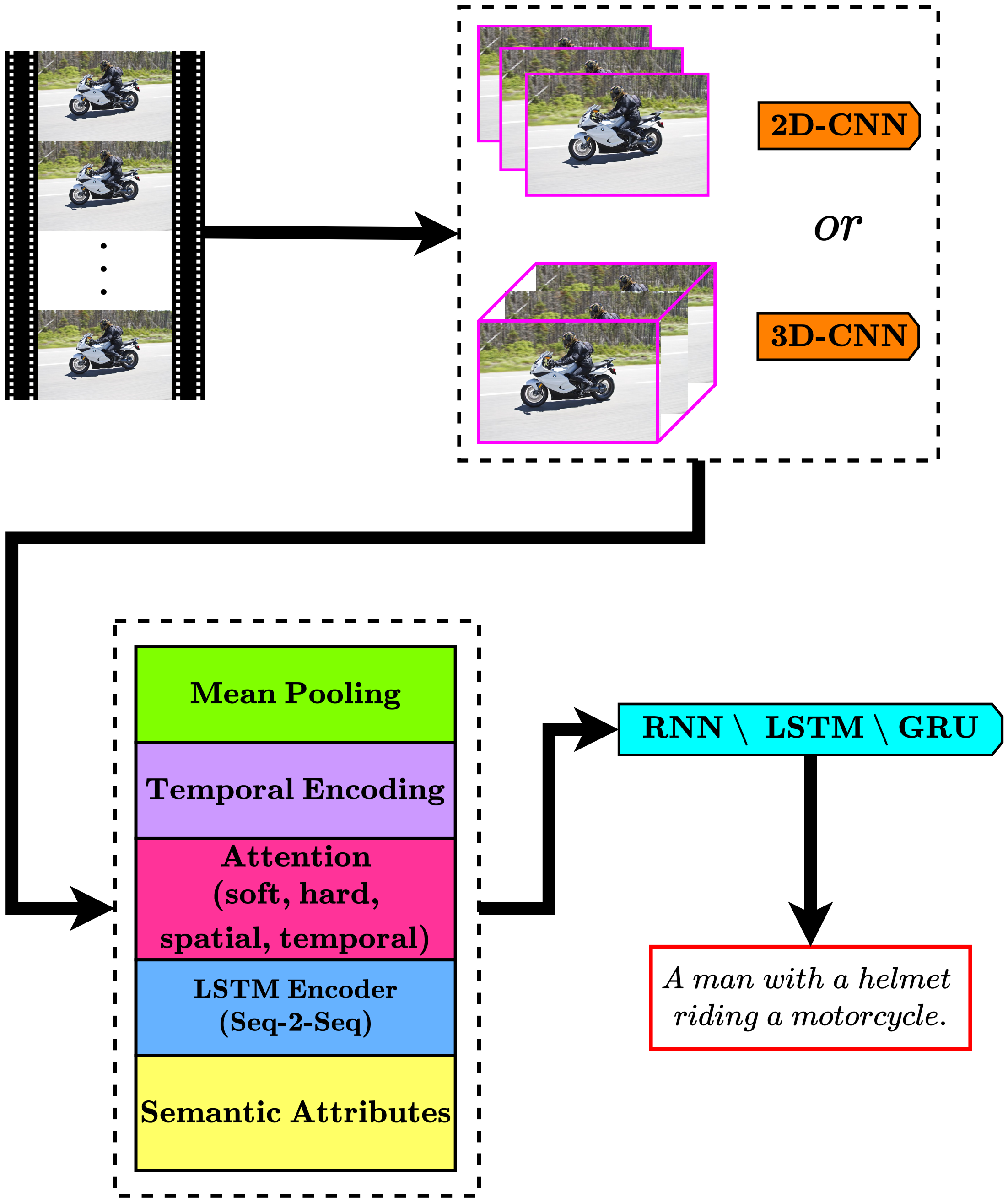


Figure 2.5: Video captioning pipeline. Often, frames are processed by 2D or 3D convolutions to extract features. Then, extracted features are processed further in the temporal domain by methods like attention, temporal encoding, LSTM, etc. Finally, a caption is generated by feeding recurrent architectures with temporally assessed features.

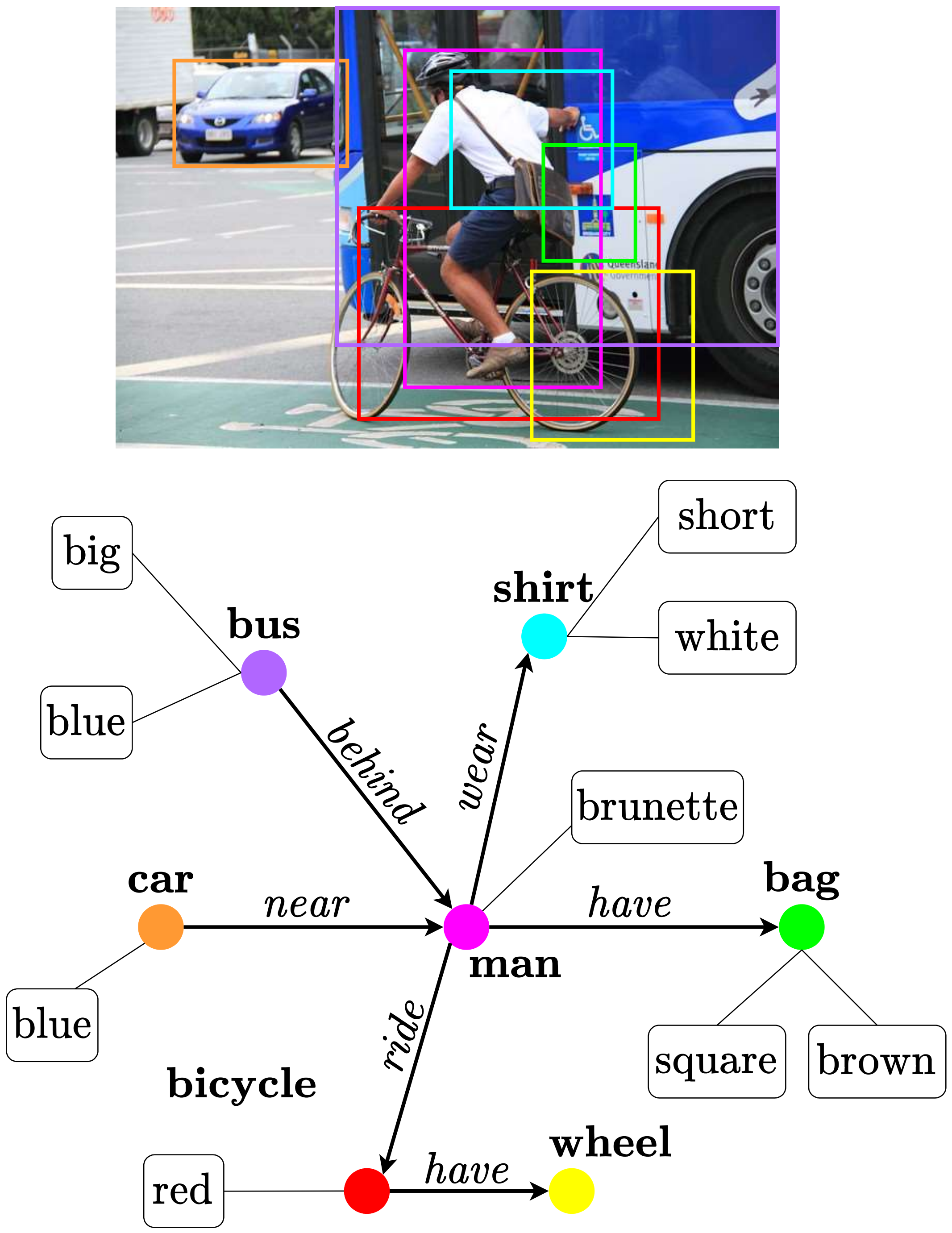


Figure 2.6: A scene graph example. Detected objects are represented as objects with circular colored nodes. Relationships among object nodes are shown with directed edges. Object attributes are denoted with rectangular uncolored nodes and they are connected to object nodes with undirected edges.

whereas b_k^S contains the coordinates of vertices that characterize the smallest rectangle covering o_k^S. Each relation between objects o_i^S and o_j^S is denoted by $p_{i\to j}^S \in \mathcal{P}^S \subseteq \mathcal{P}$ and forms the descriptive triplet $t_{i\to j}^S = \left(o_i^S, p_{i\to j}^S, o_j^S\right) \in \mathcal{T}^S$ for $i \neq j$. It is important to note that, for relation $p_{i\to j}^S = p_{\text{is}}$ we have attribute $a_{o_i,j}^S \in \mathcal{A}^S$ as the subject. Generation of scene graph $\mathcal{G}^S$ for given scene S can be formalized as below.

- $\mathcal{B}^S = \left\{b_1^S, \ldots, b_N^S\right\}$ be set of N region candidates in S where b_i^S is the bounding box of i-th candidate region
- $\mathcal{O}^S = \left\{o_1^S, \ldots, o_N^S\right\}$ be set of objects where o_i^S denoting the label of candidate region b_i^S
- $\mathcal{A}^S = \left\{a_{o_1,1}^S, \ldots, a_{o_1,k^1}^S, \ldots, a_{o_N,1}^S, \ldots, a_{o_N,k^N}^S\right\}$ be set of attributes for $k^i \geq 0$ with $a_{o_i,j}^S$ denoting j-th attribute associated with i-th object
- $\mathcal{P}^S = \left\{\ldots, p_{i\to j}^S, \ldots\right\}$ be set of predicates between objects where $p_{i\to j}^S$ predicate in triplet descriptor $t_{i\to j}^S = \left(o_i^S, p_{i\to j}^S, o_j^S\right)$

Assuming the detection of attributes and predicates as two independent processes, we can write the probability of scene graph $\mathcal{G}^S$ given scene S as:

$$\mathbb{P}\left(\mathcal{G}^S|S\right) = \mathbb{P}\left(\mathcal{B}^S|S\right)\, \mathbb{P}\left(\mathcal{O}^S|\mathcal{B}^S, S\right) \mathbb{P}\left(\mathcal{A}^S|\mathcal{O}^S, \mathcal{B}^S, S\right)\, \mathbb{P}\left(\mathcal{P}^S|\mathcal{O}^S, \mathcal{B}^S, S\right) \quad (2.1)$$

where $\mathbb{P}\left(\mathcal{B}^S|S\right)$ is the probability of proposing candidate regions given S and $\mathbb{P}\left(\mathcal{O}^S|\mathcal{B}^S, S\right)$ denotes the probability of assigning specific labels to those proposed regions.

As shown in Eq. 2.1, the scene graph generation process consists of several steps [34] and these steps are illustrated in Fig. 2.7. First, given an image, an object detection module extracts object region proposals and visual features corresponding to these regions. Generally, a Faster-RCNN [17] is used for the backbone of object detection. The extracted features are used to identify object categories and their attributes. Identified objects along with their extracted features are used as nodes in the initial graph, e.g., a fully connected dense graph.

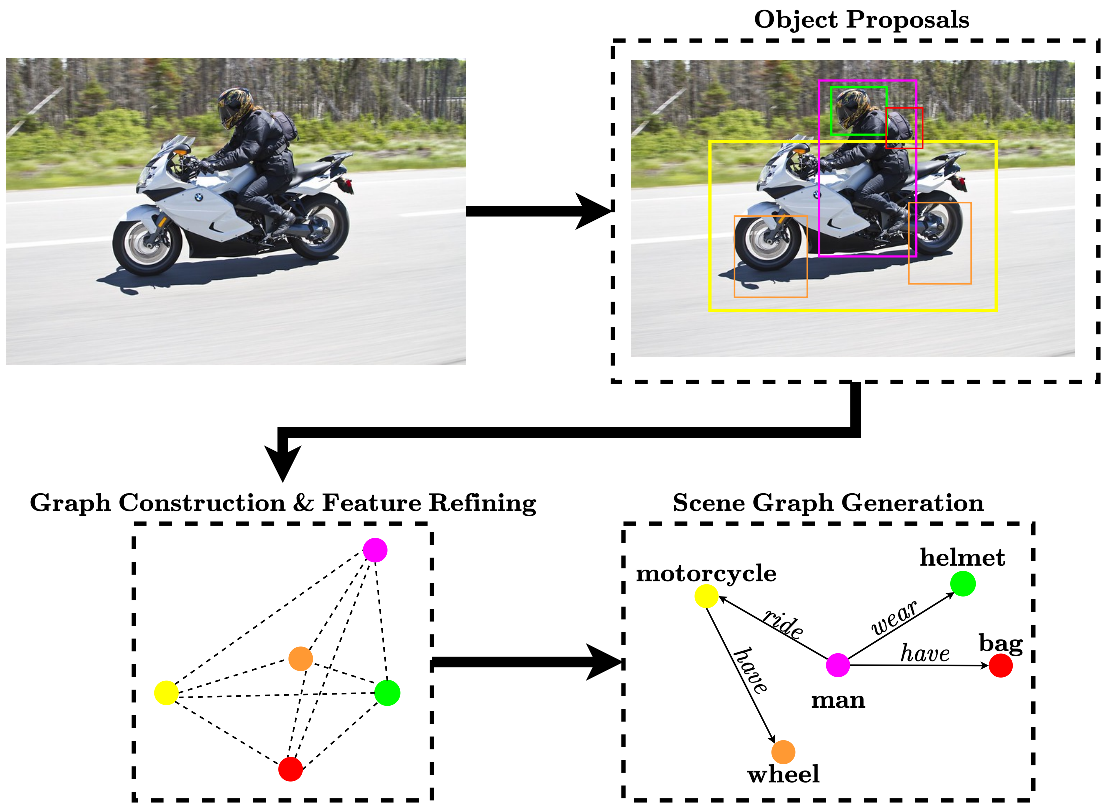


Figure 2.7: Scene graph generation pipeline. As a first step, object proposal regions are generated by object detection architectures such as RPN [17]. The proposals are used to construct a fully connected coarse graph where feature representations are established by processing region of interests (RoI) and spatial connections via backbone or additionally incorporated neural architectures. Then, the initial graph and its features are refined iteratively by utilizing techniques such as message passing, attention or visual embeddings. Refined features are used to infer node and relationship types in the final graph via outermost classifier head.

Then, the extracted features along the nodes and edges are iteratively refined, and a final graph is inferred according to these refined features. Some recent papers [35, 36, 37] also consider joint optimization of object detection and relationship recognition parts. Specifically, Factorizable-Net is proposed in [38] where an RPN is used to extract object proposals and proposed objects are paired to obtain a fully-connected initial coarse graph. In [39], Graph-RCNN is introduced. Graph-RCNN uses relation-proposal-network (RePN) to prune the connections in the initial graph and an Attentional Graph Convolutional Network [40] refines the features on the graph. On the other hand, VCTree model [41] constructs a dynamic tree from a scoring matrix where visual context is encoded into the tree structure. An illustration of the VCTree model is given in Fig. 2.8. Furthermore, some recent works [35, 36] use recurrent architectures for graph inference. Particularly, in [36] a feature refining module consisting of edge and node Gated Recurrent Units (GRU), and in [102], stacked bi-LSTMs are used.

There are approaches to incorporate external prior knowledge for scene graph generation tasks as well. In [104], natural language priors are incorporated and visual relationships and textual relationships are jointly learned and aligned. A linguistic knowledge distillation framework is proposed in [105] where statistics obtained from external texts are used to regularize visual models. In [106], knowledge-graphs are employed as prior information where the generated scene and knowledge graphs are abridged and iteratively refined.

Scene graphs can also be extracted from video signals. In [107], each frame in the video is converted into a scene graph as an intermediate semantic representation. Then, using frame and cross-frame level relationships of intermediate scene graphs, a story of the video is generated. Joint parsing of cross-view videos is introduced in [108] where scene-centric and view-centric graphs are hierarchically generated. For more details, we refer the readers to survey papers [34, 109].

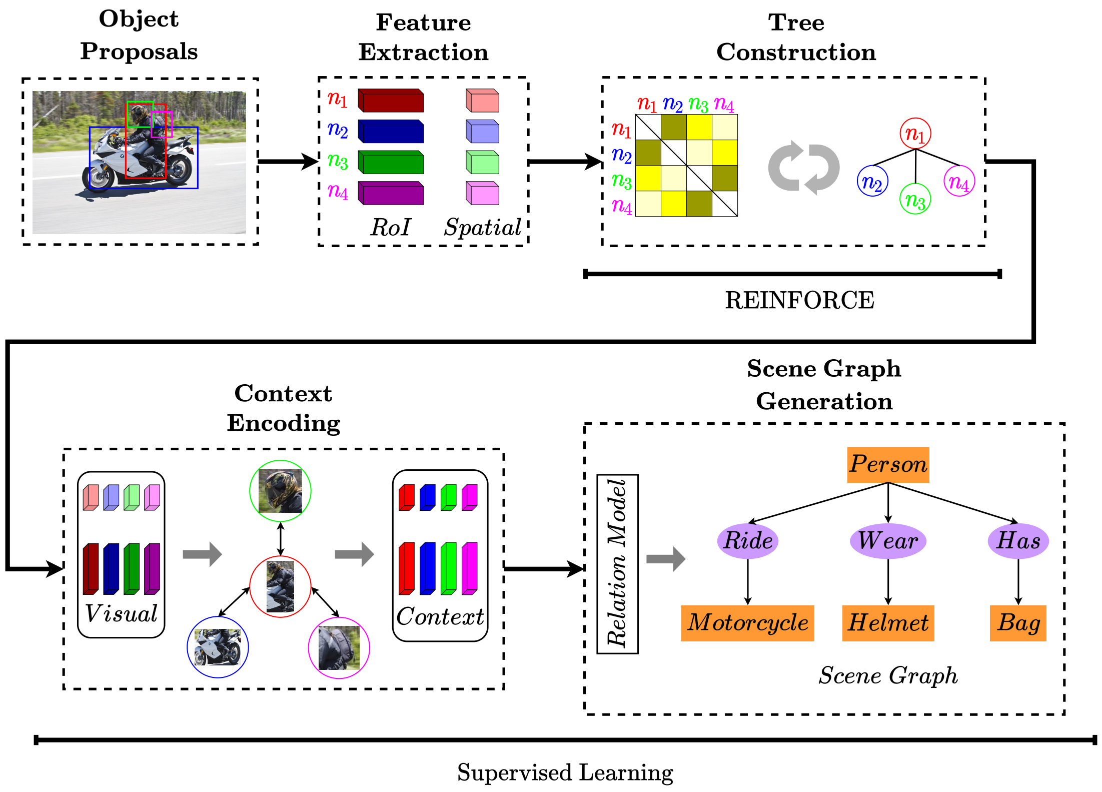


Figure 2.8: VCTree model proposed in [41]. Visual features are extracted from object proposals. Extracted features are used to compute a scoring matrix. Based on the score matrix, a dynamic tree structure is constructed using REINFORCE algorithm [103]. Finally, visual features are encoded into context features and a scene graph is generated via supervised learning.

2.5 Automatic Speech Recognition

Speech is one the most natural sources of semantic information. The most popular application of semantic transformation on speech signals is the conversion of audio signals into NL texts via automatic speech recognition (ASR) systems [110]. This semantic transformation can be explained as follows: for a given length n input sequence $\{x_i\}_{i=1}^{n}$ the role of an ASR system is to generate corresponding length m output sequence $\{y_i\}_{i=1}^{m}$ such that $\{y_i\}_{i=1}^{m}$ has the highest probability given the input sequence $\{X_i\}_{i=1}^n$:

$$\begin{aligned} \{y_i\}_{i=1}^m &= \underset{\mathbf{Z}=\{Z_i\}_{i=1}^m}{\arg\max}\, \mathbb{P}\left(\mathbf{Z} \middle| X_1 = x_1, \ldots, X_n = x_n\right) \\ &= \underset{\mathbf{Z}=\{Z_i\}_{i=1}^m}{\arg\max}\, \frac{\mathbb{P}\left(X_1 = x_1, \ldots, X_n = x_n \middle| \mathbf{Z}\right)\mathbb{P}\left(\mathbf{Z}\right)}{\mathbb{P}\left(X_1 = x_1, \ldots, X_n = x_n\right)} \end{aligned} \qquad (2.2)$$

where $\mathbb{P}(X_i = x_i)$ is the probability that x_i is present in the i-th component in the signal, $\mathbb{P}(\mathbf{Z})$ denotes the probability of occurrence of sequence of words, and $\mathbf{P}(\mathbf{Z}|X_1 = x_1, \ldots, X_n = x_n)$ is the probability that natural language output $\mathbf{Z}$ is occurring in correspondence to the audio signal $\{x_i\}_{i=1}^n$. It is important to note that, similar to the image/video captioning applications, the range of target semantic modality is formed by a predetermined portion of the desired natural language. The extent of the covered portion of the target natural language is continuously expanding throughout the years from digits [111] to tens of thousands of words [112].

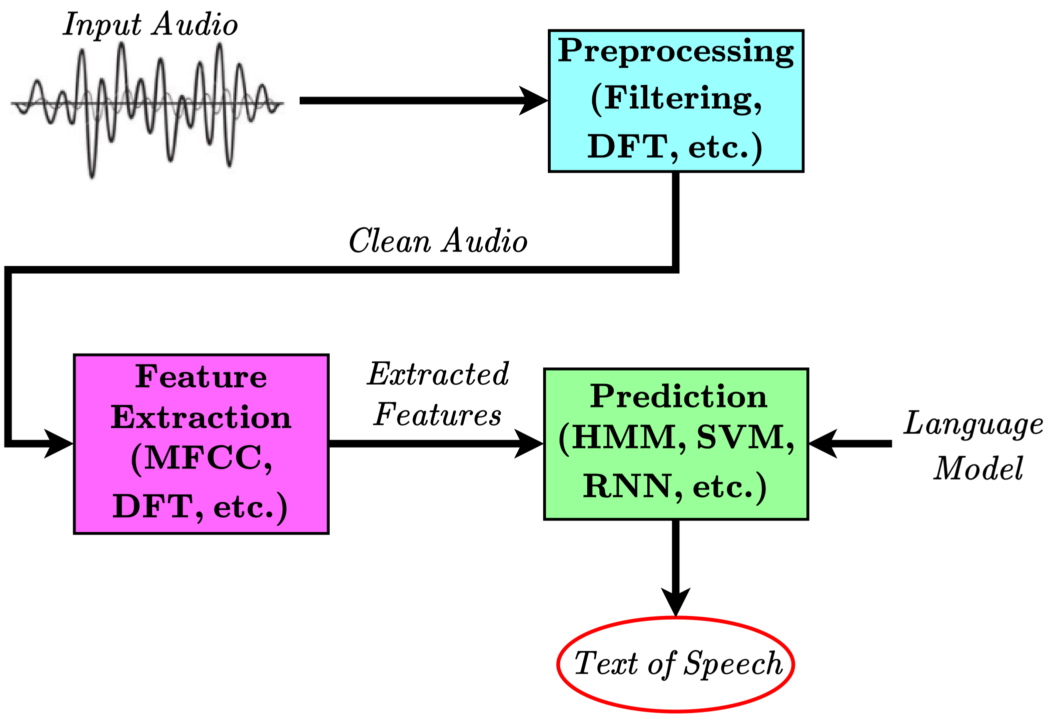

Input Audio
Preprocessing (Filtering, DFT, etc.)
Clean Audio
Feature Extraction (MFCC, DFT, etc.)
Extracted Features
Prediction (HMM, SVM, RNN, etc.)
Language Model
Text of Speech

Figure 2.9: Automatic speech recognition pipeline. The semantic features of a preprocessed speech audio signal are extracted and fed into a predictor for text generation. The prediction module is designed for a target language model.

Even though the requirements of an ASR system may vary for different applications (e.g., speaker dependency, vocabulary size, and utterance-awareness), most of the approaches conform with the process depicted in Fig. 2.9, as described in [110]. Often, an ASR system consist of four modules, namely *pre-processing*, *feature extraction*, *prediction* and *language model*. The preprocessing module includes different filters and methods such as *framing*, *pre-emphasis*, *windowing* and *normalization* to increase signal-to-noise ratio which can vary based on the utilized feature-extraction method since certain feature extraction techniques necessitate the use of a particular preprocessing procedure on its input audio signal.

The next step after obtaining the clean audio signal is the feature extraction process which is quite crucial for the performance of upcoming prediction module. Usually the extracted features are treated as intermediate semantics by the ML community. In an ASR system, the desired qualities of extracted features are robustness to noise and echo effects. The most often used feature extraction methods are Mel-frequency Cepstral Coefficients (MFCC) [112, 113, 114], Discrete Wavelet Transform (DWT) [115, 116, 117] and Linear Predictive Coding (LPC) [118, 119, 120, 121].

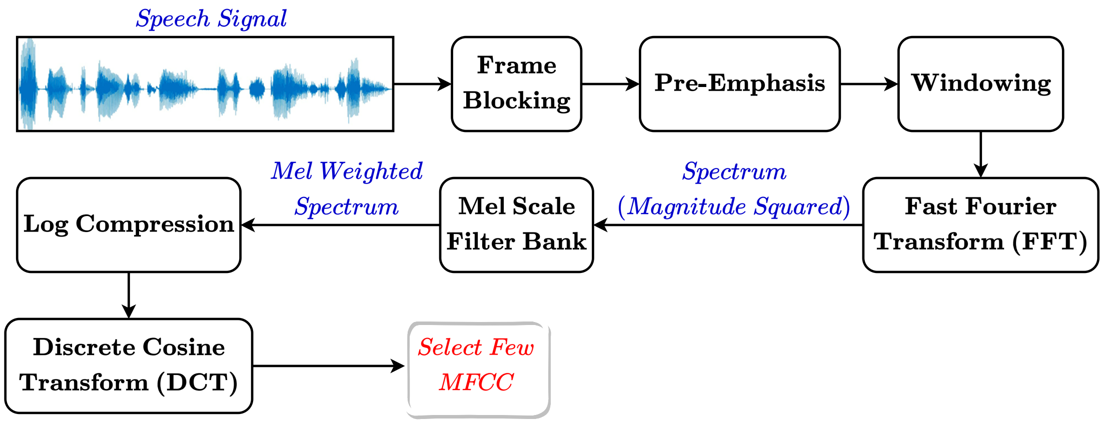


Figure 2.10: MFCC feature extraction. Input speech is generally 16 bit, 16 kHz signal. As preprocessing steps, the input audio initially divided into frames, then passed through a pre-emphasis filter to preserve higher frequency components and multiplied by a window function such as Hamming.

The schematic for extraction of MFCC is shown in Fig. 2.10. A continuous audio function, as we know, has varying values at different moments in time. To simplify MFCC extraction process, the audio stream is split into smaller overlapping frames. After dividing the audio signal into frames, each frame is multiplied by the window function such as Hamming. Then, as illustrated in Fig. 2.10, Fast Fourier Transform (FFT) is applied to obtain spectrum of the audio signal where the results are then used to calculate log energy output via Mel scale filter bank as follows:

$$Y_i = log_{10}\left(\sum_{k=0}^{M-1} H_i(k)\left|X(k)\right|\right) \text{ for } i = 1, \ldots, N \tag{2.3}$$

where $X(k)$ is the k-th window for audio source signal, N is the length of FFT, $H_i(k)$ is the Mel scale filter bank and Y_i denotes log energy outputs. Finally, Discrete Cosine Transform (DCT) is applied to obtain Mel frequency cepstral coefficients c_j which allows us to preserve most of the contained energy while reducing the dimensionality by discarding coefficients with high values but low energy:

$$c_j = \sum_{i=1}^{N} Y_i\, cos\left(j\frac{\pi}{N}(i-0.5)\right) \text{ for } j = 0, \ldots, J-1 \tag{2.4}$$

where J is the number of selected coefficients.

LPC is another feature extraction method for ASR systems first proposed in [118]. LPC imitates the structure of vocal tract in the human auditory system. The idea is to represent the current sample as linear combination of all previous samples. As in the case of MFCC extraction, to perform LPC analysis, first the input audio is split into frames and multiplied by a windowing function to ensure absence of discontinuities in the beginning and end of each frame. Then the auto-correlation between frames are calculated and LPC analysis is performed:

$$z[n] \approx \sum_{k=1}^{p} z[n-k]\, c[k] \tag{2.5}$$

where $z[n]$ denotes the current sample point of the calculated auto-correlation signal and p is the total number of previous sample points. The main goal of LPC analysis is to find coefficients $\hat{\mathbf{c}}$ so that the mean squared error E is minimized:

$$\hat{\mathbf{c}} = \underset{c[1],\ldots,c[p]}{\arg\min}\, E = \underset{c[1],\ldots,c[p]}{\arg\min} \sum_{n}\left(z[n] - \sum_{k=1}^{p} z[n-k]\, c[k]\right)^2 \tag{2.6}$$

The final module of an ASR system is the prediction module, to output the most likely text sequence. The prediction module takes the extracted features as input from the previous stage and can adopt generative, e.g. Hidden Markov Models (HMM) [122, 123] and Gaussian Mixture Models (GMM), parametric, e.g. SVMs [124, 125, 126], RNNs [127, 128, 129] and CNNs [130, 131, 132, 133], or hybrid approach to obtain the text equivalence in the desired language. The language model is employed to impose predefined vocabulary and grammar rules such as phonemes. More details about ASR can be found in the pertinent survey papers [110, 134].

2.6 Semantic Radar Signal Processing

Up to now, we only reviewed semantic transformation examples that are applied to visual, audio and lexical data. Nevertheless, it is also possible to define appropriate semantic transformations in other modalities such as radar signals. In recent years, many works utilized ML and DL techniques to process radar signals effectively to perform tasks such as waveform design, target recognition or interference mitigation [135, 136]. In this section, we briefly review some works that can be considered as semantic transformation in radar signal processing. A coarse framework of semantics in radar signal processing is illustrated in Fig. 2.11.

Radar has well-known reputation of being robust against weather and lighting conditions, a desired attribute especially for emerging applications such as autonomous driving. In [137], it is proposed to use CNNs to extract semantic representation of environment from raw radar signals by cell-wise classification of the objects contained in the radar grid which enables foregoing priory object extraction. Indoor mapping problem with mmWave radars are considered in [138] where the dense grid map is constructed using the received mmWave signals and additional semantic mapping is applied to identify object construction and space accessibility such as walls, doors or lifts. In [139], several CNN based DL architectures proposed to tackle semantic segmentation of radar data cube into multiple views namely range-Doppler and range-Angle maps.

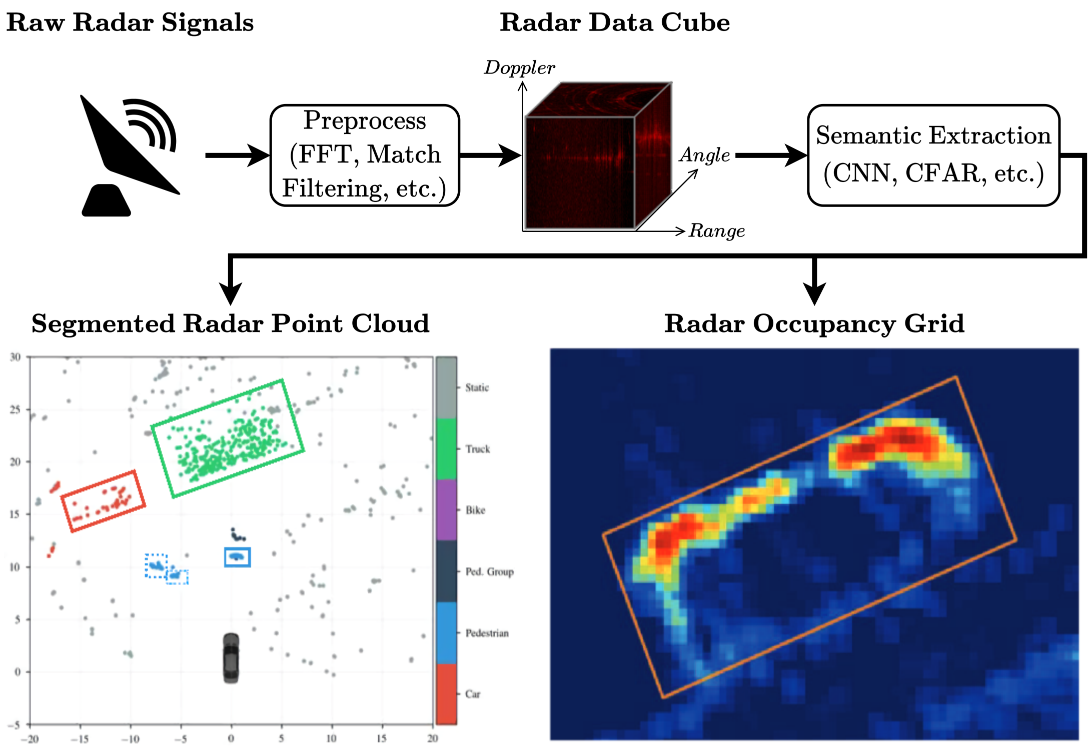


Figure 2.11: Semantic extraction in radar signal processing. The received raw time series data first preprocessed to obtain radar data cube. Next, the target semantics such as segmented radar point cloud map or occupancy grid are generated via DL architectures or conventional methods.

Several works utilized logical-mathematical constructions to describe radar operations to formalize the processes of semantic radar. Air object detection is considered in [140], where it is proposed to convert the received radar signals into spectral images using algebra on finite predicates on the set of spectral channels, i.e., spectral patterns, to identify spectral types. In [141], this approach extended to detection of low-visible air objects in multi-survey radar signal processing such that radar images of noise markers and objects are converted into logical dependencies similar to the human logic, and in [142], fuzzy transformations are used to identify the semantic dependencies among radar raw data and images where finite predicates are processed via boolean algebra and basic graph theory. Furthermore, methodology for evaluation of efficiency of the spectral-semantic radar signal processing model is presented and results are discussed in [143].

2.7 Goal-Oriented Semantic Communications

In recent years, there has been a revived interest in semantic communications as a result of enormous improvements in machine learning, notably on deep learning. Many studies envisioned that semantic communication will pave the way for next generation wireless communications [27, 33, 21, 144, 145, 146, 147]. For instance, in [21] the authors proposed to adopt semantic communications for 6G networks to improve the effectiveness and sustainability without increasing the usage of communication resources such as bandwidth and energy. The authors argued that the next generation systems should focus on identification of the relevant meaning rather than the bit-level recovery of transmitted data. In [146], the authors pointed out the possible benefits of joint information generation, transmission, and consumption. They further argued that transmission of most meaningful data samples can give end users access to the latest and the most crucial data which can improve the utilization of network resources especially in next generation massive intelligent networks. On the other hand, in [145], the authors proposed to utilize the significance of messages relative to the purpose of data exchange such as value of information (VoI) in contrast to the meaning, and analysed how it can be utilized for tasks including sampling, multiple access, resource allocation and scheduling. Therefore, in this section we review the literature on semantic and goal-oriented communications. Illustration for the common goal-oriented semantic communication framework and multiple communication levels is shown in Fig. 2.12. As it is shown, the structure consists of three levels, namely effectiveness, semantic and technical level. The technical level of communication focuses on bit-level transmission accuracy and utilizes channel coding and modulation techniques to recover transmitted symbols from their corrupted counterparts by the physical channel. The semantic level of communication deals with the extraction and encoding of essential information relevant to the task based on the knowledge bases, and often interpreted as the intelligent part of the communication which leads to significant reduction in data traffic. Compared to the technical level of communication, this level focuses on the recovery of the meaning rather than the recovery of the transmitted bit-sequence, i.e., the expressed message and the inferred message may not have identical bit symbols, however, they may convey the same desired meaning just like synonymous words. The robustness against the semantic noise, i.e., an ephemeral noise that adds ambiguity or diverges the desired meaning, imposed by the implicit semantic channel is a desired quality for expressed messages. Many works utilize ML and DL techniques at this level for the conversion of input modality into an arbitrary semantic message whereas some works merge the semantic and technical level of communications under the AI-enabled joint source-channel coding (JSCC) framework. As the highest level of communication, the effectiveness level considers the interaction between the communicating agents and the environment. In particular, the contained effectiveness factors quantify the performance of the agents on their respective tasks, i.e., measures whether the communication assisted the agents to conduct their work in the desired way. Many studies adopt a form of reinforcement learning approach to address the problem of effective communications.

As a recent example, in [28] a deep learning enabled semantic communication architecture called DeepSC is proposed for text transmission. Unlike traditional approaches where the objective is to minimize bit errors, this work focused on minimizing semantic errors while maximizing the system capacity with a mutual information estimation model for semantic channel coding. The authors utilized sentence similarity metrics, namely BLUE and BERT, to quantify the semantic error at sentence level and illustrated robustness of the proposed architecture compared to traditional technical level approaches, especially in the low signal-to-noise ratio (SNR) regime. In [29] previous work is extended to internet-of-things (IoT) applications where lite and distributed version of [28] called L-DeepSC is proposed. The authors utilized channel state information (CSI) aided training scheme to mitigate the effects of fading channels while pruning and sparsifying the model parameters for affordable IoT devices.

The problem of adaptive and responsive sentence level semantic communication is tackled in [148]. The authors proposed to adopt Adaptive Computation Time (ACT) [149] halting on transformer based semantic extractor, i.e., a circulation mechanism to refine encoded semantic message, for more flexible and simultaneous transmission of sentences with varying semantic information. The

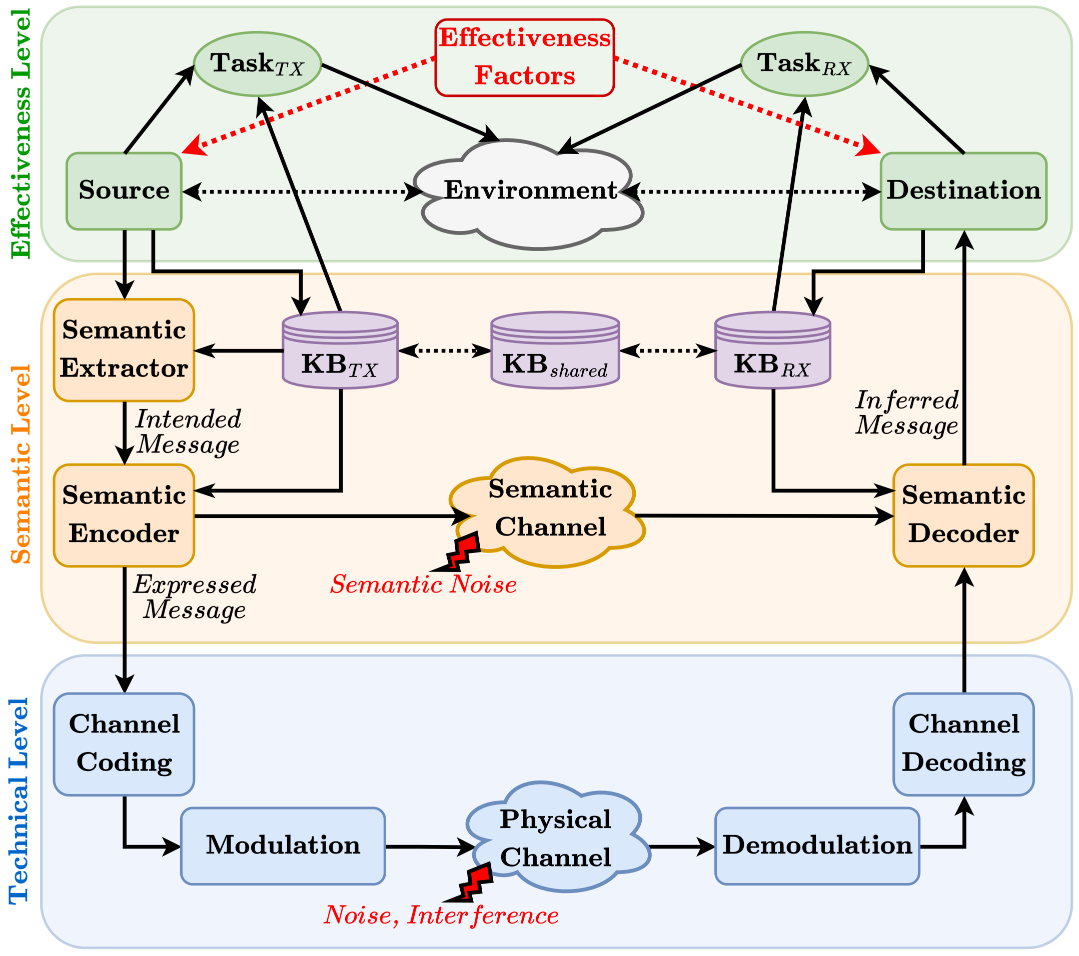


Figure 2.12: Multi level structure of goal-oriented semantic communications. The structure consists of three levels, namely effectiveness, semantic and technical level. The technical level of communication focuses on bit-level transmission accuracy and utilizes channel coding and modulation techniques to recover transmitted symbols from their corrupted counterparts by the physical channel. The semantic level of communication deals with the extraction and encoding of essential information relevant to the task based on the knowledge bases where many works utilize recently developed ML and DL techniques. The effectiveness level quantifies the performance of the communication agents on their respective tasks and often a form of reinforcement learning approach is adopted at this level.

signal shaping method to minimize semantic loss (SSSC) is proposed in [150] for text transmission where the problem is formulated as power constrained vector optimization solved by a projected gradient descent algorithm. In [151] the authors proposed to leverage both deep learning based and traditional methods (SCHARQ) for text transmission in semantic communication. Specifically, semantic source coding is utilized together with Reed-Solomon (RS) channel coding and hybrid automatic repeat request (HARQ) to reduce semantic error further which trained by similarity acknowledgments.

Transmission of speech data for semantic communications is considered in [152]. The authors proposed a deep learning enabled system called DeepSC-S which extracts essential semantic information from speech audio signals by using squeeze-and-excitation (SE) networks [153] and applies attention based weighting to emphasize the essential information where source and channel encoder/decoder is jointly designed to mitigate channel distortion. The proposed architecture shown to outperform traditional approaches almost in any SNR regime.

Other works on semantic communication address the transmission of image modality through a wireless channel [30, 154, 155, 156, 157, 158, 159, 160]. In [156], reinforcement learning based training scheme for joint semantics and noise coding is adopted to transmit images over phase-invariant fading (FIF) channels. The authors proposed a learnable confidence module over a transformer architecture to distil the semantic message embeddings. The fusion of high-level, e.g., captioning and segmentation results, and low-level semantics, e.g., local spatial details, for image transmission over wireless channel is proposed in [155]. The authors utilized pre-trained segmentation and image captioning model and trained another model (MLSC-image) in end-to-end manner for image reconstruction task. It is illustrated that proposed method outperforms both traditional methods and newer approaches such as DeepJSCC [30] on image reconstruction task upon transmission over AWGN channel. In [154], a semantic communication architecture (SC-AIT) is introduced where both effectiveness, semantic and technical level of communication is inter-connected via neural network supported module. The proposed architecture is realized on a real-world test-bed for image classification task to detect surface defections. The authors showed that the proposed architecture traditional approaches in any SNR regime while having lower latency and higher compression ratio.

Real-time semantic communications is first studied in [159] where the authors developed a prototype for wireless image transmission based on the field-programmable gate array (FPGA), Vision Transformer (ViT) [161] and denoising auto-encoder. The implemented prototype has been shown to be superior to traditional 256-quadrant amplitude modulation (256-QAM) in the low SNR regime on a measure of structural similarity index (SSIM). In [160] constellation constrained version of DeepJSCC [30] (DeepJSCC-Q) is proposed for wireless image reconstruction. DeepJSCC-Q utilizes a differentiable soft quantization layer to map latent semantic vectors to transmitted symbols such that each quantization level corresponds to a learnable constellation point. It is illustrated that DeepJSCC-Q achieves comparable performances with respect to its unconstrained counterpart and outperforms the traditional methods which rely on separate source and channel coding scheme.

Task unaware semantic communication in a dynamic data environment is studied in [157] where the receiver's pragmatic task is not known by the transmitter and the distribution of observed data by the transmitter is non-identical to the one in the background knowledge library. The authors proposed to incorporate a domain adaptation module that comprised of cycle-GAN [162] prior to the semantic extractor module to mitigate the problem of observation and knowledge mismatch which allows utilization of the same semantic extractor without re-training as shown in Fig 2.13. The proposed method is tested on image classification and image segmentation tasks without an information leakage to the transmitter regarding the underlying task at the receiver. In [158] the impact of semantic noise to the performance of semantic communication systems is investigated. The authors first proposed an iterative adversarial fast gradient sign method (FGSM) [163] for generating semantic noise, and then used an adversarial training technique with weight perturbation to combat the semantic noise. The proposed architecture comprised of masked-ViTs as semantic source encoder/decoder and vector quantized variational auto-encoders (VQ-VAE) [164] to learn shared discrete codebook for encoded features which further improves robustness against semantic noise and reduces transmission overhead.

Multi-user semantic communication system with multi-modal data transmission is studied in [165] where authors performed visual question answering (VQA) task subsequent to transmission over wireless channel. The proposed deep learning enabled semantic communication system (MU-DeepSC) utilizes two distinct semantic extractor and channel encoders to process image and textual (question) input data whereas the receiver employs a single unified architecture to decode concatenated semantic message of both transmitters and outputs the answer text as illustrated in Fig. 2.14. The proposed method shown to be superior compared to traditional approaches in all SNR-regimes.

Deep learning aided end-to-end JSCC for wireless video transmission scheme (DeepWive) is introduced in [167] where the proposed architecture performs video compression, channel coding and modulation with a single neural network, resulting in direct mapping from video signals to channel symbols whereas the trained semantic decoder predicts the residuals without distortion feedback. Furthermore, to maximize the overall visual quality, the authors adopted reinforcement learning approach to simultaneously train another model for optimization of channel bandwidth allocation among video frames where the resulting policy outperforms naive uniform allocation by a significant margin.

Inference and error correction capabilities of semantic communication framework is studied in [168]. The authors proposed a cognitive semantic communication framework that utilizes triplet components of knowledge graphs as extracted semantic features. Specifically, the input text data first converted into a knowledge graph to identify semantic redundancies and semantic ambiguities then distilled semantic message is encoded into semantic channel symbol. To improve the robustness of proposed system, the authors further utilized a correction algorithm involves traversing the semantic symbols in the knowledge graph to locate the most comparable one observed at the receiver. The illustration of proposed cognitive semantic communication framework is shown in Fig. 2.15. The proposed framework is tested against conventional separate source and channel

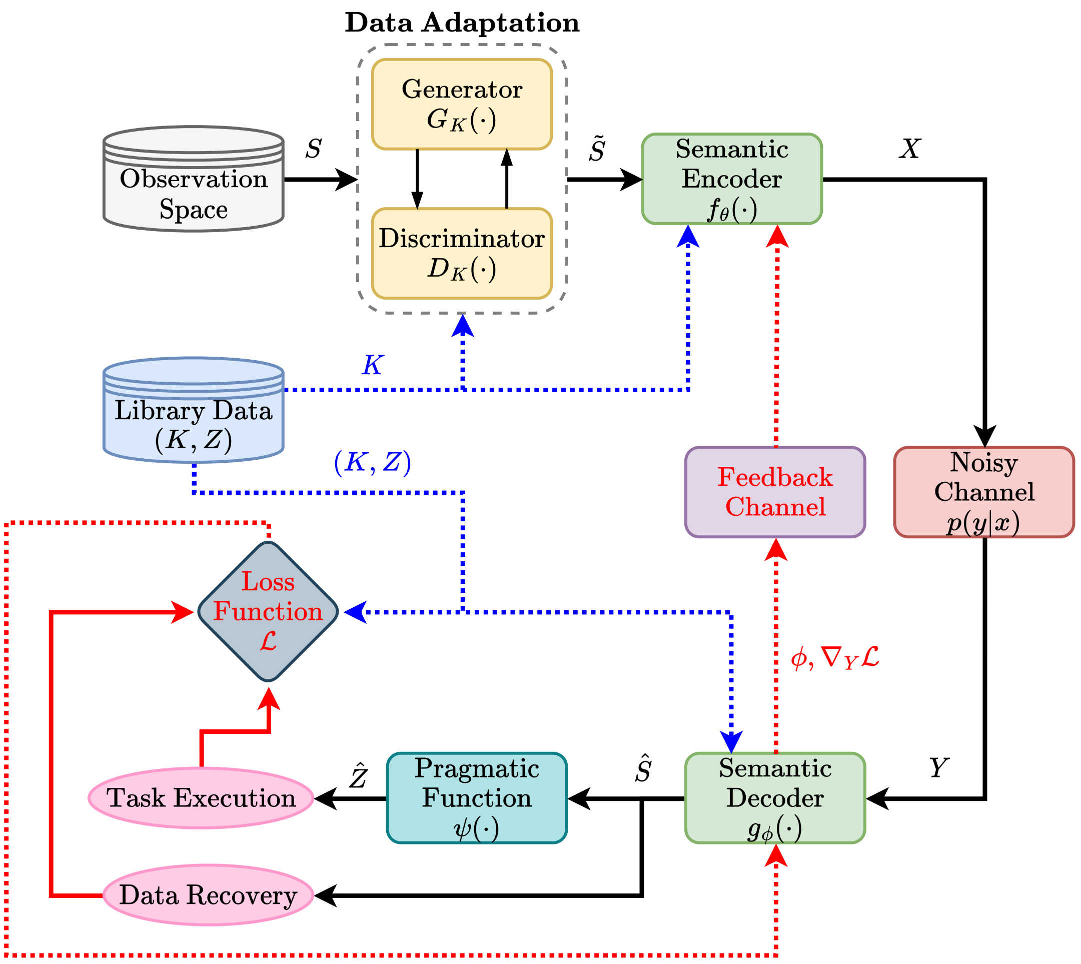


Figure 2.13: Task unaware semantic communication system with dynamic data environment proposed in [157]. The transmitter has no empirical knowledge regarding the observable data S. The empirical data K and its corresponding pragmatic task Z, e.g., image classification and image segmentation, comes from the shared library data between the transmitter and the receiver. Data Adaptation module comprised of cycle-GAN [162] to handle distribution mismatch between observations and the shared knowledge library. Modules that are utilized only in training of semantic encoder and decoder are denoted in red. The loss function quantifies the quality of extracted (or recovered) semantic information by the semantic encoder (or decoder) to perform the task and to reconstruct the input data for human usage.

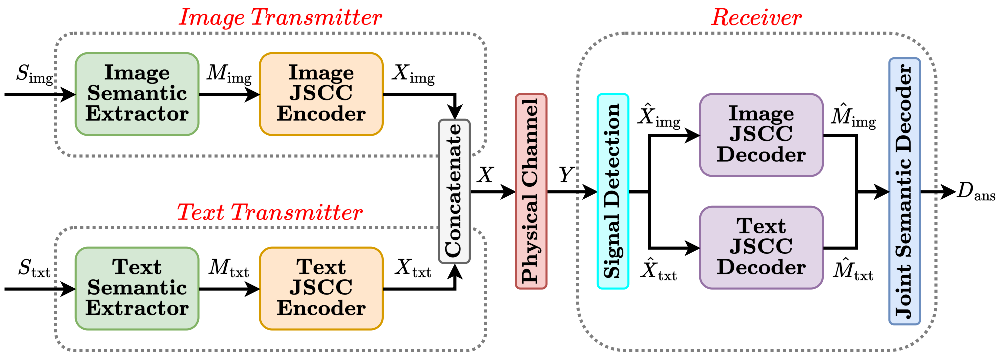


Figure 2.14: MU-DeepSC [165] framework for multi-modal semantic communication on visual question answering task. The image and text source denoted by S_{img} (query image) and S_{txt} (question) respectively. The image transmitter utilizes CNNs for semantic extractor to obtain semantic information M_{img} and for channel encoder to transmit symbols X_{img} whereas the text transmitter takes advantage of Bi-directional LSTM and MLP architectures to acquire M_{txt} and transmit X_{txt} respectively. The signal detection module employs channel estimation to mitigate the effects of noisy channel. Then, the semantic information $\hat{M}_{\text{img}}$ and $\hat{M}_{\text{txt}}$ is reconstructed by CNN and MLP architectures respectively. Finally, the reconstructed information is processed by a memory attention and composition network (MAC-network) [166] to output the answer D_{ans}.

coding technologies under the binary symmetric channel (BSC) and shown to be superior in terms of compression rate and reliability.

Another interesting study on semantic communications with knowledge graphs is performed in [169] where the authors focused on semantic communication system that can adjust the transmitted semantics contents adaptively according to estimated channel quality. Then, the system sorts the extracted triplets according to their semantic importance and allocates more transmission sources to semantically important ones such that when the channel quality is extremely poor the system transmits only the most significant triplet. To transform textual inputs into knowledge graphs, the proposed semantic extractor uses named entity recognition (NER) units, LSTM, and MLP, whereas the semantic decoder utilizes graph attention network (GAT), RNN, and MLP units to recover semantic messages corrupted by the channel medium.

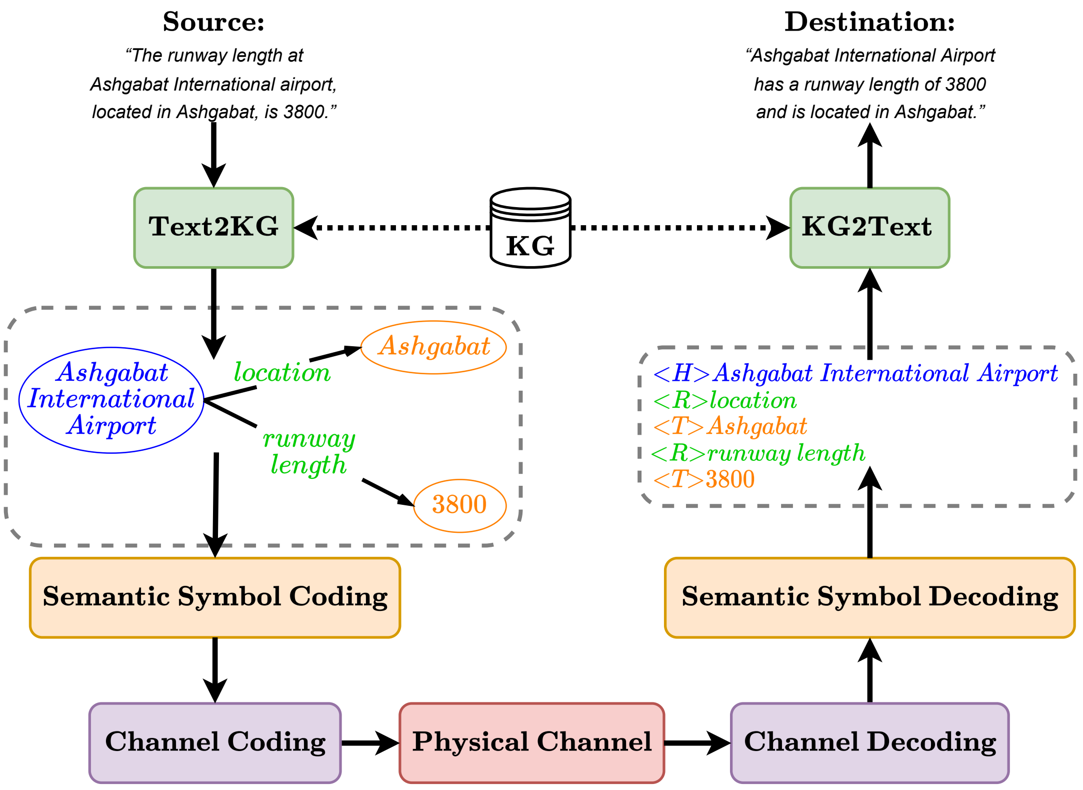

Source:
"The runway length at Ashgabat International airport, located in Ashgabat, is 3800."
Destination:
"Ashgabat International Airport has a runway length of 3800 and is located in Ashgabat."
Text2KG
KG
KG2Text
Ashgabat International Airport
location
Ashgabat
runway length
3800
<H>Ashgabat International Airport
<R>location
<T>Ashgabat
<R>runway length
<T>3800
Semantic Symbol Coding
Semantic Symbol Decoding
Channel Coding
Physical Channel
Channel Decoding

Figure 2.15: Cognitive semantic communication framework for text transmission proposed in [168] leverages knowledge graph as arbitrary semantic modality. The text-to-knowledge graph (Text2KG) module is a simple alignment algorithm which sequentially processes each sentence and constructs a concise knowledge graph consist of triplets (*head, relation, tail*). The Semantic Symbol Coding module is a simple dictionary (*key, value*) that maps each possible triplet to a unique integer and encodes the integers to fixed length bit-sequence. The Semantic Symbol Decoding module is an correction algorithm which traverses the possible triplets in the knowledge graph to find most similar ones observed by the receiver. Finally, the recovered set of triplets are processed by the knowledge graph-to-text (KG2Text) module comprised of a fine-tuned NLP transformer module to output semantically identical text.

In [170], effectiveness of semantic communications is studied within *joint learning and communication framework*. The authors adopted multi-agent reinforcement learning (MARL) approach to develop a systematic structure for collaborative agents participating in treasure hunts. The problem is formulated as a multi-agent partially observed Markov decision process (MA-POMDP). Agents are intended to learn policies to effectively exchange messages over a shared noisy wireless channel for improved coordination and collaboration while taking long-term rewarding actions. The proposed formulation incorporates noisy wireless medium as environment dynamics and transmitted messages as part of actions where the authors employed deep Q-learning (DQN) [171] and deep deterministic policy gradient (DDPG) [172] for training of agents solely based on reward with no information regarding the channel. In particular, the authors focused on guided robot problem in grid world as illustrated in Fig. 2.16. As a result, the proposed joint learning and communication framework achieves higher performance than treating action-taking and communication aspects of MARL separately.

A reasoning based semantic communication architecture (R-SC) is proposed in [173] that can adaptively expand the set of recognizable semantic meanings. In [173], semantic information is represented in a graph-based structure which also contains inference rules between semantic entities and relationships. The proposed approach utilized embedding functions, i.e., additive, linear and multiplicative embeddings, to map high dimensional semantic graph structures to low dimensional representations for transmission over channel, and a novel inference method on the receiver side to figure out incomplete entities and relations that cannot be observed in the transmitted graph message. Furthermore, the introduced life-long learning model updates allows receiver to exploit prior messages to discover new semantic entities and relations.

A theoretical work [174] introduced a semantic communication framework inspired by human communications. The authors proposed a stochastic model (SNC) in which the transmitter eliminates nuisances and quantize the observations into finite set of semantic elements (concepts), then, transmits their symbolic representations with the intent of referring to corresponding semantic entities. In order to reduce the communication overhead and increase the reliability,

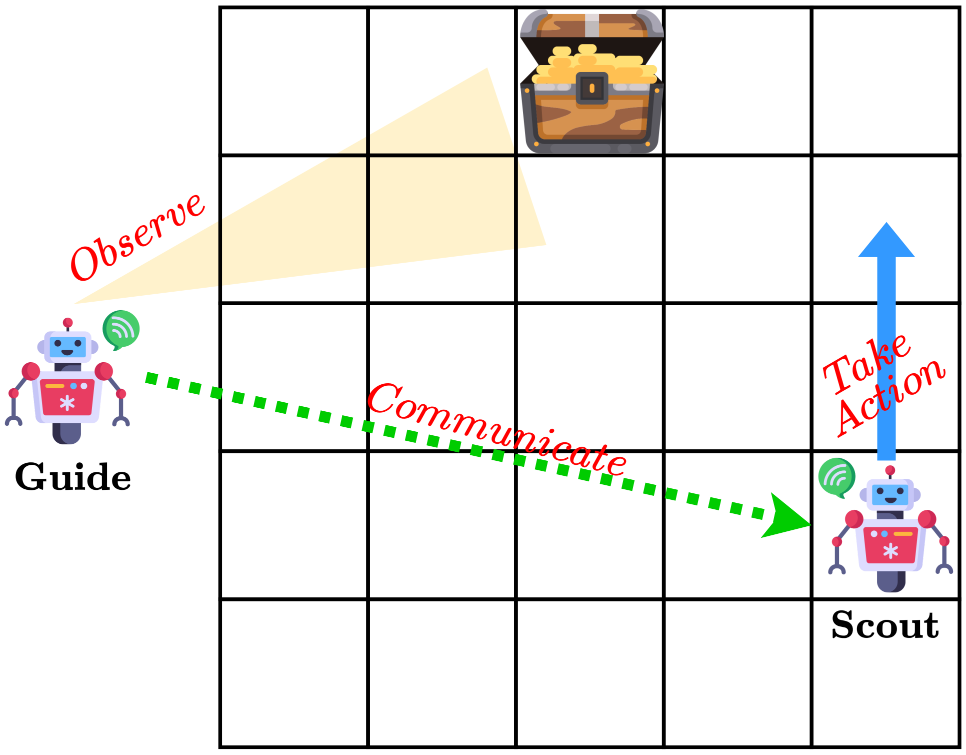


Figure 2.16: Proposed formulation of effectiveness level communication [170] as a guided robot problem in grid world as MARL. The guide has perfect information about the state of environment but it is unable to take any action on the grid world whereas the scout can take actions, i.e., 16 possible movements and their corresponding hand crafted codewords, however cannot observe the environment state at all. Hence, the guide leads the scout to capture the treasure chest by communicating through noisy channel. The authors models both the guide and the scout as 3-layer MLPs and utilize deep reinforcement learning algorithms to train the agents. In particular, the guide is trained to learn a semantic encoding robust against the variations in the channel for the transmission of the action that the scout should take. On the other hand, the scout is trained to learn a semantic decoding technique to accurately recover the transmitted action-to-take.

the proposed transmitter iteratively and locally self-communicates with a virtual receiver before transmitting the final symbols to the actual one. Furthermore, the authors present theoretical results on compression capabilities of the proposed method whereas experiments unfold the trade-off between latency and reliability of this contextual reasoning aided semantic communication system.

In [175] effectiveness aspect of goal-oriented semantic communications is investigated in which the transmitter and receiver cooperate to perform sequential tasks in a dynamic environment using a common language structured by a hierarchical belief set among them. In the proposed framework, the transmitter observes the environment and transmits the initial description based on the beliefs whereas the receiver infers and completes the transmitted information by incorporating additional related beliefs and takes the corresponding action to affect the environment as illustrated in Fig. 2.17. The authors formulated an constrained optimization problem to determine appropriate semantic descriptors for each observation while minimizing transmission and reception costs such that the adopted bottom-up curriculum learning (CL) approach, based on RL, enables gradual identification of the structure of hierarchical belief set. It is illustrated that the proposed CL framework outperforms traditional RL approaches.

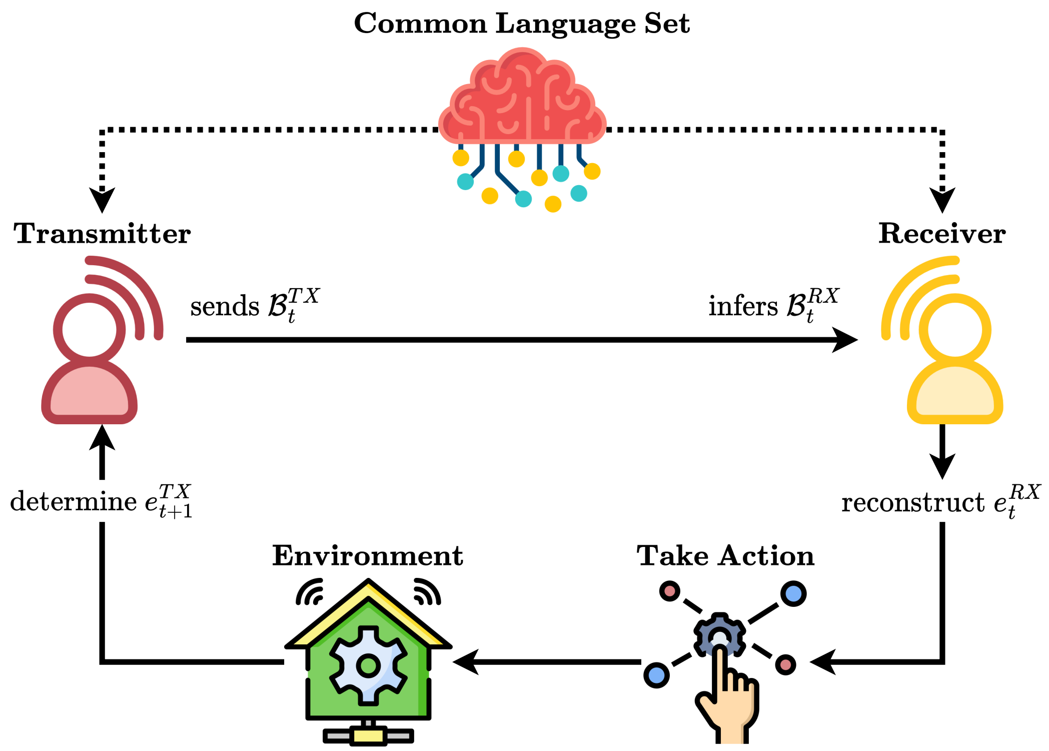


Figure 2.17: Schematic of goal-oriented semantic communications with common language set proposed in [175]. The transmitter observes an event e_t^{TX} at time t as the environment state and describes it with its believes $\mathcal{B}_t^{TX}$, i.e., with elements from the common language set independent of the data modality. Next, the receiver infers $\mathcal{B}_t^{RX}$ based on $\mathcal{B}_t^{TX}$ and reconstructs the event e_t^{RX} based on the combined believes $\mathcal{B}_t = \left\{\mathcal{B}_t^{TX}, \mathcal{B}_t^{RX}\right\}$. Finally, the receiver takes an action based on the reconstructed event e_t^{RX} and triggers transition of the environment to the next event e_{t+1}^{TX} observed by the transmitter. Since the transmitter and receiver have no information about the usefulness of any descriptor, they gain this knowledge by experiencing rewards while executing various tasks. Particularly, constructed belief sets hierarchically expand at each iteration based on the incurred transmission, inference and task execution costs. Therefore, the problem is formulated as constrained infinite horizon discounted cost Markov decision process. Due to sparseness of rewards and massive number of possible descriptor the authors adopt curriculum learning approach of MARL to solve the formulated constrained optimization problem such that the transmitter learns to choose useful beliefs to describe each event as concise as possible whereas the receiver learns to complete the transmitted description.

Chapter 3

Adopted Goal-Oriented Semantic Language and Signal Processing Framework

In this chapter, first, we re-introduce the proposed graph-based semantic language in [25] that is able to express any signal modality with a well-organized and easy-to-parse directed bipartite structure. Then, we rigorously define fundamental building-blocks of the proposed multi-graph semantic description, namely node and attribute sets, and a goal-oriented filtering operation to parse and extract the relevant information from the input signal. Finally, we end this chapter by re-introducing the proposed semantic signal processing framework in [25] and describing the contained conceptual blocks in detail.

3.1 Graph-based Semantic Language

The proposed semantic language in [25] adopts directed bipartite graph structure, illustrated in Fig. 3.1, as semantic representation of a signal composed of *semantically connotated components.* In this representation, the signal components are first categorized into a predefined set of classes (denoted with solid circles in Fig. 3.1), then, based on (again predefined) set of predicates (denoted with dashed circles in Fig. 3.1) their states such as $c_1 \rightarrow p_0$, and relationships with other components such as $c_2 \rightarrow p_2 \rightarrow c_3$, are labeled as tuples or triplets respectively. As illustrated in Fig. 3.1, both the signal components and their semantic states (or relationships) are depicted as nodes with labels c_i and p_i respectively. It is important to note that, some applications might require a preprocessing stage which projects the raw sensor signal to an appropriate domain prior to the identification of the signal components and the corresponding semantic relationships. Furthermore, unlike general graph structures where feature extraction and pattern matching like operations can be difficult, especially as the graph complexity grows, the proposed hierarchical bipartite structure grants considerably simpler yet complete semantic representation of a signal while possess the qualities of being easy to generate and operate on.

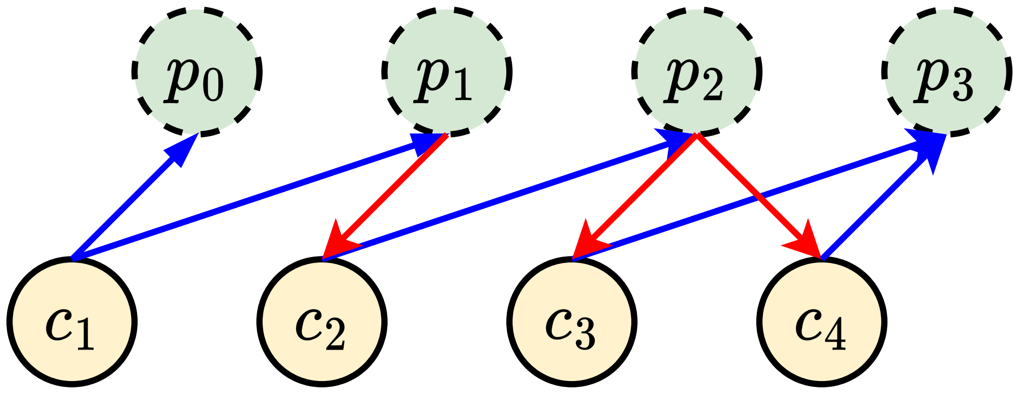


Figure 3.1: Proposed semantic language graph in [25] utilizes directed bipartite graph structure formed by signal components and predicates. The signal components are drawn with solid yellow circles and depicted as nodes c_i whereas the predicate nodes are drawn with dashed green circles and denoted by p_i. This structure allows us to express both state indicating information such as $c_1 \rightarrow p_0$ or $c_4 \rightarrow p_3$, and relation denoting information such as $c_2 \rightarrow p_1 \rightarrow c_3$ or $c_2 \rightarrow p_2 \rightarrow c_4$, simultaneously. Note that, a signal component can be connected to more than one predicate and vice-versa.

3.2 Multi-Graph Semantic Description

As previously mentioned, the semantically connotated components of the proposed bipartite graph structure utilizes predefined sets to label the signal components and the predicates, and we denote these sets as $\mathcal{C}$ and $\mathcal{P}$ respectively:

$$\mathcal{C} = \{c_1, c_2, \ldots, c_{N_c}\} \tag{3.1}$$

$$\mathcal{P} = \left\{p_0, p_1, \ldots, p_{N_p}\right\} \tag{3.2}$$

where N_c and $N_p + 1$ are the number of component and predicate classes in the graph language respectively. It is important to note that, the first element of the predicate set is defined as separate p_0 class, also referred as *null* or *exist* predicate class, to denote the detected signal components with no other semantic connection to another component. In other words, these signal component nodes are connected to the predicate p_0 node to ensure the existence of non-zero edge weight in the corresponding biadjacency matrix representation and omits the presence of isolated nodes in the semantic graph. Furthermore, the predicate set $\mathcal{P}$ can contain other state describing predicates similar to the p_0 without the intention of referring to any relation among components. As illustrated in Fig. 3.1, state describing predicate nodes like p_0 have single directed connection to signal components nodes, e.g., $c_1 \to p_0$ or $c_4 \to p_3$, while relation indicating predicate nodes form triplets depending on their definitions, e.g., $c_1 \to p_1 \to c_2$ or $c_2 \to p_2 \to c_4$.

The *multi-graph semantic description* of a signal at time t defined as the set of connected bipartite graphs:

$$\mathcal{S}_t = \{S_{t,1}, S_{t,2}, \ldots, S_{t,N}\} \tag{3.3}$$

where N is the number of *atomic class graphs* in the data that represent disjoint bipartite graphs. An i-th atomic class graph at time t is denoted by:

$$S_{t,i} = (\mathcal{C}, \mathcal{P}, \mathcal{E}_{t,i}) \tag{3.4}$$

is a connected directed bipartite graph with component and predicate node sets $(\mathcal{C}, \mathcal{P})$ and edges $\mathcal{E}_{t,i}$ at time t:

$$\mathcal{E}_{t,i} = \{\ldots, (c_i, p_j), (p_j, c_l), \ldots\}. \tag{3.5}$$

Since the suggested graph structure is bipartite, we only permit the connections from signal component classes to predicate classes and vice versa. This allows to represent each connected graph $S_{t,i}$ in (3.3) via the following biadjacency matrix:

$$\boldsymbol{S}_{t,i} = \begin{bmatrix} \mathbf{0} & \boldsymbol{S}_{t,i}^{\mathcal{CP}} \\ \boldsymbol{S}_{t,i}^{\mathcal{PC}} & \mathbf{0} \end{bmatrix}, \tag{3.6}$$

where $\boldsymbol{S}_{t,i}^{\mathcal{CP}}$ and $\boldsymbol{S}_{t,i}^{\mathcal{PC}}$ represent the directed edges from the signal component nodes to the predicate nodes and the other way around respectively. In detail, each $[m, n]$-th entry of the sub-matrices in (3.6) denotes:

$$\boldsymbol{S}_{t,i}^{\mathcal{CP}}[m,n] = \begin{cases} 1, & \text{if } \exists \text{ connection from } c_m \text{ to } p_n, \\ 0, & \text{otherwise,} \end{cases} \tag{3.7}$$

$$\boldsymbol{S}_{t,i}^{\mathcal{PC}}[m,n] = \begin{cases} 1, & \text{if } \exists \text{ connection from } p_m \text{ to } c_n, \\ 0, & \text{otherwise.} \end{cases} \tag{3.8}$$

It is important to note that, the node sets $\mathcal{C}$ and $\mathcal{P}$ in the multi-graph description are class definitions, and in reality there may have multiple detections of the members of these sets in a signal. However, this class level description of the signal provides a higher level abstraction, and it is advantageous when the goals are specified over classes rather than individual instances of signal components and predicates. In order to cover the cases where goals are defined over individual instances, we introduce a lower level abstraction counterpart $\mathcal{D}_t$ of (3.3) again in the form of bipartite graphs which only utilizes the detected instances from sets $\mathcal{C}$ and $\mathcal{P}$. Therefore, we define the *detected signal component* set $\mathcal{C}_{t,i}^{D}$ and *detected predicate* set $\mathcal{P}_{t,i}^{D}$ for i-th atomic class graph $S_{t,i}$ at time t as follows:

$$\mathcal{C}_{t,i}^{D} = \left\{ (c_j, k) \;\middle|\; \begin{matrix} k \in [1, N_{t,i}^{c_j}] \text{ and } \exists m \text{ with} \\ \boldsymbol{S}_{t,i}^{\mathcal{CP}}[j,m] = 1 \text{ or } \boldsymbol{S}_{t,i}^{\mathcal{PC}}[m,j] = 1 \end{matrix} \right\} \tag{3.9}$$

$$\mathcal{P}_{t,i}^{D} = \left\{ (p_j, k) \;\middle|\; \begin{matrix} k \in [1, N_{t,i}^{p_j}] \text{ and } \exists m \text{ with} \\ \boldsymbol{S}_{t,i}^{\mathcal{CP}}[m,j] = 1 \text{ or } \boldsymbol{S}_{t,i}^{\mathcal{PC}}[j,m] = 1 \end{matrix} \right\} \tag{3.10}$$

where $N_{t,i}^{c_j}$ and $N_{t,i}^{p_j}$ are the number of detected instances of the signal component class c_j and the predicate class p_j in the atomic graph $S_{t,i}$, and k is the unique identifier for each class instance. Using the detection sets in (3.9) and (3.10) we denote the *multi-graph instance description* and *atomic instance graphs* analogously as:

$$\mathcal{D}_t = \{D_{t,1}, D_{t,2}, \dots, D_{t,N}\} \tag{3.11}$$

$$D_{t,i} = \left(\mathcal{C}^D_{t,i}, \mathcal{P}^D_{t,i}, \mathcal{E}^D_{t,i}\right) \tag{3.12}$$

where the corresponding *detected connection list* $\mathcal{E}^D_{t,i}$ consists of connections in the form of triplets as:

$$\mathcal{E}^D_{t,i} = \{\dots, \left((c_i, k_i), (p_j, k_j), (c_l, k_l)\right), \dots\}. \tag{3.13}$$

Unlike the nodes in atomic class graphs, nodes in atomic instance graphs are indexed by unique identifiers which allows us to express multiple individual components or predicates belong to the same class simultaneously. As a concrete example, the atomic class graph may contain a descriptor such as $person \rightarrow ride \rightarrow bike$ whereas the detected atomic instance graph may contain more than one person such as $(person, 1) \rightarrow (ride, 1) \rightarrow (bike, 1)$ and $(person, 2) \rightarrow (ride, 2) \rightarrow (bike, 2)$. It is also important to note that in (3.13), we allow last entry of each triplet to be empty in order to represent singular connections from a signal component to a predicate but not the other way around which is in line with the definition of p_0 predicate. Analogously, we define the instance level counterparts of biadjacency matrix in (3.6) for $D_{t,i}$ as follows:

$$\boldsymbol{D}_{t,i} = \begin{bmatrix} \mathbf{0} & \boldsymbol{D}^{\mathcal{CP}}_{t,i} \\ \boldsymbol{D}^{\mathcal{PC}}_{t,i} & \mathbf{0} \end{bmatrix}, \tag{3.14}$$

where $\boldsymbol{D}^{\mathcal{CP}}_{t,i}$ and $\boldsymbol{D}^{\mathcal{PC}}_{t,i}$ represent the connections from signal component nodes to predicate nodes, as in (3.7, 3.8), and the other way around where the instances are listed in order across the rows and columns.

As noted before, the multi-graph sets $\mathcal{S}_t$ and $\mathcal{D}_t$ provide a more simplified hierarchical representation of a signal easier to operate on rather than a general single graph structure. It is also important to note that, in machine-type applications, the predicate and component sets may scale according to the complexity and processing capabilities of sensors and devices. Specifically, for a simple sensor the multi-graph representation may not have more than a handful of signal components and predicate classes which facilitates the graph pattern matching and goal-filtering operations with simple available methods proposed in [49, 50, 51, 52].

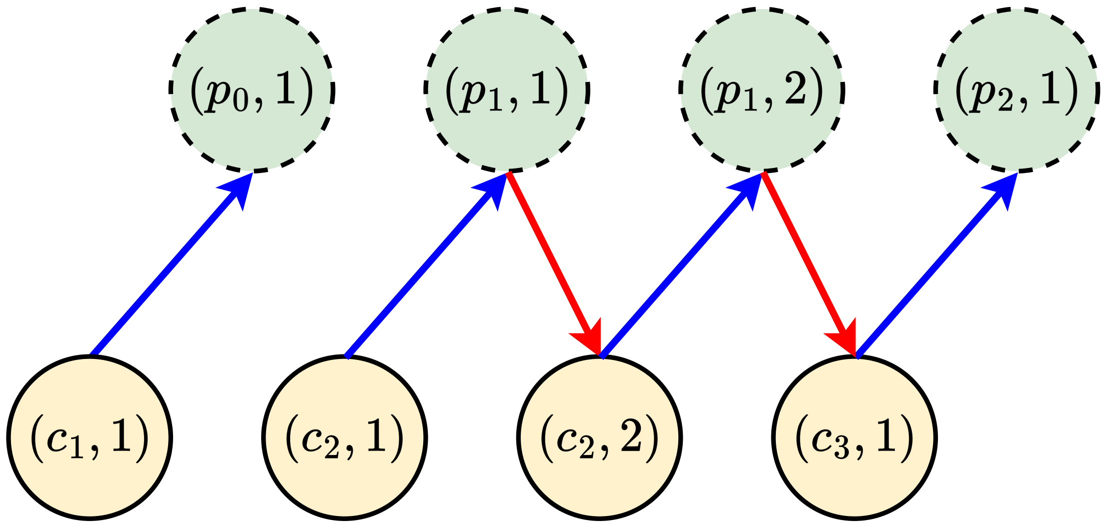


Figure 3.2: Instance level semantic graph $\mathcal{D}$. Just like class level graph $\mathcal{S}$, instance level graph $\mathcal{D}$ preserves the directed bipartite graph structure formed by signal components and predicates. As illustrated, there may be more than one instance that belongs to the same class, both for signal components and predicates which are indexed by k. The signal components are drawn with solid yellow circles and depicted as nodes (c_i, k_i) whereas the predicate nodes are drawn with dashed green circles and denoted by (p_j, k_j). This structure allows us to express both state indicating information for individual instances such as $(c_1, 1) \rightarrow (p_0, 1)$ or $(c_3, 1) \rightarrow (p_2, 1)$ and relation among the individual components and predicates such as $(c_2, 1) \rightarrow (p_1, 1) \rightarrow (c_2, 2)$ or $(c_2, 2) \rightarrow (p_1, 2) \rightarrow (c_3, 1)$, simultaneously. Note that, since $(c_1, 1)$ component is disconnected from others, it is connected to $(p_0, 1)$ by default.

3.3 Attribute Sets

The class definitions and detected instances do not necessarily constitute a complete semantic description of a signal, therefore, in [25], we utilized associated attribute sets for each signal component and predicate node to fully capture additional properties and features. Specifically, for each atomic instance graph $D_{t,i}$

in the multi-graph description the attribute superset $A_{t,i}$ is defined as:

$$A_{t,i} = \left\{\Theta_{t,i}(n_j, k) \middle| (n_j, k) \in \mathcal{C}^D_{t,i} \cup \mathcal{P}^D_{t,i}\right\} \tag{3.15}$$

where (n_j, k) corresponds to a node and instance index pair, and $\Theta_{t,i}(.)$ is defined as the multi-level attribute set of its argument nodes as:

$$\Theta_{t,i}(n_j, k) = \left\{\boldsymbol{\theta}^{(1)}_{t,i}(n_j, k), \boldsymbol{\theta}^{(2)}_{t,i}(n_j, k), \ldots, \boldsymbol{\theta}^{(L_{n_j})}_{t,i}(n_j, k)\right\} \tag{3.16}$$

where L_{n_j} is the number of levels in attributes for j-th node. Note that, the number of levels may vary for each type of signal component or predicate, or may be kept similar for all nodes to for simplicity. We like to point out that, the introduction of attribute levels allows hierarchical organization of attributes from simpler to complex representation which facilitates the goal-oriented filtering operation. Just like the $\mathcal{C}$ and $\mathcal{P}$, the exact definition of (3.16) is depends on the application, specification of goals, and the performed signal processing techniques to extract semantic information.

The companion attribute set for instance multi-graph description $\mathcal{D}_t$ defined as:

$$\mathcal{A}_t = \{A_{t,1}, A_{t,2}, \ldots, A_{t,N}\} \tag{3.17}$$

and finally, the full semantic description Y_t at time t is expressed as a triplet consist of the class level multi-graph description $\mathcal{S}_t$, the instance level multi-graph description $\mathcal{D}_t$ and the corresponding companion attribute set $\mathcal{A}_t$:

$$Y_t = (\mathcal{S}_t, \mathcal{D}_t, \mathcal{A}_t) \tag{3.18}$$

3.4 Semantic Goals and Goal-Oriented Filtering

In this section, we re-introduce a goal operator on the proposed semantic graph language to parse and filter the desired portion of the semantic information from a signal. For simplicity, we drop the time index subscript t from this point. A goal-filtering operator $\mathcal{G}$ on semantic description Y outputs filtered semantic description $\tilde{Y}$ again as a triplet:

$$\mathcal{G}(G^S, G^D, \boldsymbol{l}_g, \boldsymbol{l}_a)\{Y\} = \widetilde{Y}, \tag{3.19}$$

$$\widetilde{Y} = (\widetilde{\mathcal{S}}, \widetilde{\mathcal{D}}, \widetilde{\mathcal{A}}), \tag{3.20}$$

where the argument $G^S = (\mathcal{C}, \mathcal{P}, \mathcal{E}^S)$ is queried bipartite graph pattern in the class level graph $\mathcal{S}$, $G^D = (\mathcal{C}^D, \mathcal{P}^D, \mathcal{E}^D)$ is queried bipartite graph in the instance level graph $\mathcal{D}$, $\boldsymbol{l}_g$ is graph complexity (or neighbourhood) vector, and $\boldsymbol{l}_a$ is attribute complexity vector which determines the structure of the output in terms of multi-graph representation and companion attribute sets respectively.

The utilization of class level and instance level queries for $\mathcal{G}$ enables us to define goals with different abstraction levels, namely, *global goals* with interest on class level information and *local goals* with the attention on specific instances and corresponding relations. For real world applications, we expect users to define initial global goals that slowly changing over time and more transient local goals to track the status of a detected component of interest.

The graph complexity vector $\boldsymbol{l}_g$ is vector on natural numbers defining the l_g-hop neighbourhood around components of each queried graph pattern in $\mathcal{G}$ such that the size of the vector is equal to the number of subgraphs to be searched which enables us to identify specific patterns. Using G^S, G^D and $\boldsymbol{l}_g$ the output semantic multi-graphs $\tilde{\mathcal{S}}$ and $\tilde{\mathcal{D}}$ can be defined as:

$$\widetilde{\mathcal{S}} = \left\{ \widetilde{S}_i \;\middle|\; \begin{array}{c} \widetilde{S}_i \subset S_i,\ G^S \subset \widetilde{S}_i \text{ and } \widetilde{S}_i \text{ includes} \\ l_g\text{-hop neighborhood of } G^S \text{ within } S_i \end{array} \right\} \tag{3.21}$$

$$\widetilde{\mathcal{D}} = \left\{ \widetilde{D}_i \;\middle|\; \begin{array}{c} \widetilde{D}_i \subset D_i,\ G^D \subset \widetilde{D}_i \text{ and } \widetilde{D}_i \text{ includes} \\ l_g\text{-hop neighborhood of } G^D \text{ within } D_i \end{array} \right\} \tag{3.22}$$

It is important to note that, one can employ specifically designed graph pattern matching and l_g-neighbourhood searching algorithms to generate filtered class level multi-graph $\mathcal{S}$ in (3.21) and filtered instance level multi-graph $\mathcal{D}$ in (3.22) or may prefer to utilize available alternative algorithms in the aforementioned literature [49, 50, 51, 52] where the computational complexity is relatively low and well-scaling according to complexity of the sensor or device. Hence, once again, we would like to emphasize the simplicity and capability to provide complete semantic description of the proposed multi-graph description compared to general single-graph structure. In Fig. 3.3, the goal filtering operation for queried class level graph (Fig. 3.3b) and graph complexity of $l_g = 1$ is illustrated. The exact matching results is denoted with magenta and the requested 1-hop neighbourhood around the matched instance is shown inside the shaded purple parallelogram in Fig. 3.3c. The output pattern which includes the component instances separated by at most $l_g = 1$ predicates with the goal-matched pattern is shown in Fig. 3.3d.

After finding out the class and instance level filtered graph outputs $\widetilde{\mathcal{S}}$ and $\widetilde{\mathcal{D}}$, their corresponding attribute sets are also filtered with the attribute complexity vector $\boldsymbol{l}_a$ which retains only the first l_a levels within the hierarchy of attribute sets:

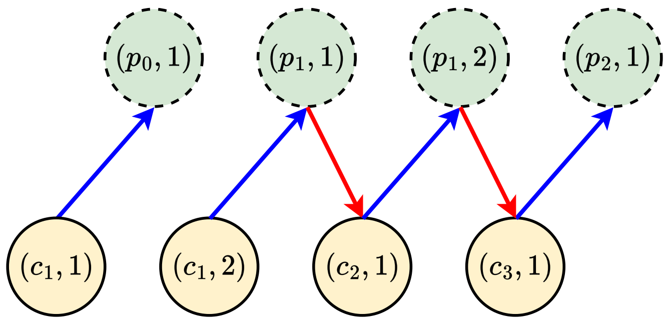
$$\widetilde{\mathcal{A}} = \{\widetilde{\Theta}_i \in \widetilde{D}_i \mid \Theta_i = \{\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \dots, \boldsymbol{\theta}^{(L_{min})}\}\}, \tag{3.23}$$

where $L_{min} = min(l_a, L)$ is the minimum of the requested and available levels of attribute complexity.

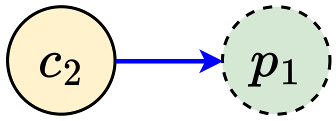
3.5 Goal-Oriented Semantic Signal Processing Framework

The proposed semantic signal processing framework in [25] with its generic form is illustrated in Fig. 3.4 which utilizes the proposed graph language re-introduced at the beginning of this chapter. In the following, we give brief descriptions for each conceptual block in Fig. 3.4 and discuss possible techniques for real-world implementation at the goal and the hardware level.

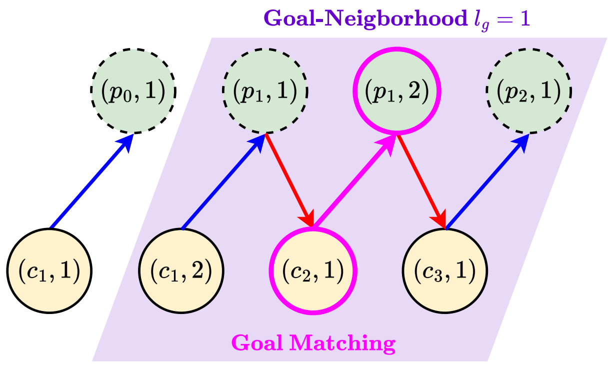
The proposed goal-oriented semantic signal processing framework consists of the conceptual blocks of preprocessing, semantic extraction, semantic filtering, semantic post-processing, and storage or transmission, depending on the application. The *preprocessing* block, as in many applications, is present to perform prefiltering for noise and interference reduction. Furthermore, it allows



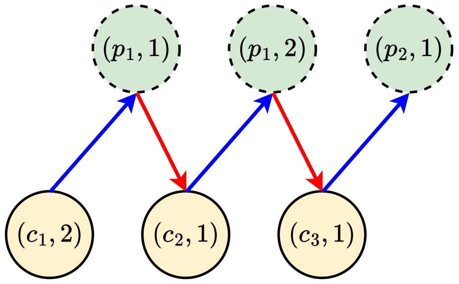
(a) Instance level semantic input graph $\mathcal{D}$



(b) Class level queried graph pattern G^S



(c) Graph pattern matching and $l_g = 1$ goal-neighbourhood search



(d) Instance level goal-filtered semantic output graph $\tilde{\mathcal{D}}$

Figure 3.3: An example goal-filtering process with G^S and l_g arguments. (a) The instance level input graph $\mathcal{D}$ contains 3 types of predicates and signal components with varying multiplicity regarding instances. (b) The class level queried graph pattern G^S seeks to gather all the nodes where a c_2 type component instance is connected to p_1 type predicate instance. (c) The input $\mathcal{D}$ contains the queried graph pattern G^S which denoted in magenta. The specified neighbourhood complexity is $l_g = 1$ retrieves the other signal component and predicate nodes within 1-hop distance with respect to predicates. (d) The instance level output graph $\tilde{\mathcal{D}}$ has lesser complexity than the input graph $\mathcal{D}$, since the isolated $(c_1, 1)$ component and the associated *null* predicate $(p_0, 1)$ instances are goal-filtered.

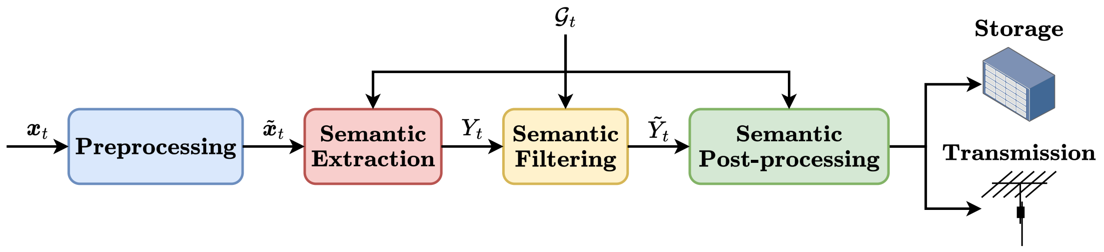


Figure 3.4: The proposed goal-oriented semantic signal processing framework in [25]. First, the input signal $\boldsymbol{x}_t$ is processed by the *preprocessing* block to obtain $\tilde{\boldsymbol{x}}_t$ version that is more suitable for semantic extraction. The latter block semantic extraction is responsible for detection and classification of signal components and predicates present in the semantic graph description Y_t. In the next block, i.e., *semantic filtering*, the generated semantic description is pruned according to the goal $\mathcal{G}_t$ of the underlying application to output refined version $\tilde{Y}_t$. The last block is the semantic post-processing where final processes such as denoising and statistical modelling is handled on goal-filtered representation $\tilde{Y}_t$. Finally, storage or transmission is realized depending on the application.

input signal $\boldsymbol{x}_t$ to be transformed to an appropriate version $\tilde{\boldsymbol{x}}_t$ where semantic extraction, i.e., detection and classification of semantic signal components and predicates, can be performed efficiently. For certain input signal modalities (e.g., 1D time series signals such as speech signal), decomposition of the signal for graphical representation may not be evident in the first place or the input signal may be highly contaminated by noise. In such cases, commonly used domain transformation approaches such as the time-frequency and the time-scale domain processing [176, 177, 178], can be used to segment and detect distinct signal components via pattern recognition and computer vision techniques [179] (e.g., see Fig. 2.10 for MFCC extraction). For text input modality, expansion of contractions (e.g., *"ain't"* $\xrightarrow{\text{to}}$ *"are not"*), removal of stopwords (e.g., *"the"*, *"a/an"*, *"be/am/is/are"*, etc.) or punctuations, tokenization, lemmatization, stemming and embedding are some of the preprocessing steps for NLP applications [180, 181]. On the other hand, for image and video applications, transformation to another domain may not be necessary, as component detection and classification in these domains can be efficiently implemented on the input signals directly.

The *semantic extraction* block is where the multi-graph description Y_t (i.e., the signal components, predicates and connections) with corresponding attribute sets are generated from the preprocessed input data $\tilde{\boldsymbol{x}}_t$. Note that, as shown in Fig. 3.4, a global goal $\mathcal{G}_0$ can be incorporated at this point. If a constant or slowly changing goal is defined either internally, or it is received from an external agent, the semantic extraction algorithm can adapt accordingly. For example, if the global goal is only related to a small subset of potential components that can be extracted, the algorithm efficiency can be increased by limiting the output classes of the classification algorithm implemented as a DNN. In the case of semantic representation of 2D signals such as images or 3D data such as video streams of MRI volume, components (or objects) can be first identified by appropriate object detection or semantic segmentation techniques (see Chapter 2 and references therein). Furthermore, for image and video input modalities, available scene graph extractors discussed in Chapter 2 can be utilized [101, 34, 35, 36, 37, 38, 39, 40, 41] to form the basis of the proposed graph structure which can be further refined. Alternatively, spatio-temporal information can be exploited with the assist of object detection and tracking methods, which will be illustrated in Chapter 4 for video-stream data, for identification of predicates. For text input modality, the input can be initially transformed to knowledge-graph by simple alignment algorithms such as [168]. Speech signals, first can be converted to textual counterparts such as sentences by ASR algorithms discussed in Chapter 2, then the usual schemes to process text input can be followed.

In the next block, *semantic filtering* operations are performed. Once the scene descriptions Y_t are generated, local and time-varying goals $\mathcal{G}_t$ can be applied to filter semantic information of interest within Y_t. In a typical application, local goals $\mathcal{G}_t$ are either generated internally via decision-making algorithms or are received from external agents. This block typically employs graph signal processing, graph search and pattern matching algorithms to match and extract goal queries from the graph-based representation of the signal. Note that the semantic output Y_t, generated by the proposed language yields relatively simple graphs due to their bipartite and goal-oriented definitions; hence, the computational complexity of any graph operation is considered to be more tractable compared to general graph structures. In certain scenarios, the goal-filtering can be reduced to subgraph isomorphism search problem where available algorithms in the literature such as VF3 [182], TurboIso [183], QuickSI [184], STW [185] and SPath [186] can be utilized. Note that, these algorithms adopt the *backtracking* strategy to recursively found and report matched vertices. In addition, class level queries can be handled by techniques developed for reasoning over knowledge graphs (including multi-hop queries) which generally utilizes vector algebra over embedding space of graph [187, 188, 189, 190, 191] (see [192] for pertinent survey on knowledge graphs).

The *semantic post-processing* block is a conceptual block where further processing of the extracted and goal-filtered semantic data can be performed which can also incorporate local goals $\mathcal{G}_t$, to schedule transmission and storage operations according to the level of desired semantic information.

Note that at this point, the semantic data are strictly organized and only include goal-filtered components and their corresponding attributes, i.e., quite distilled for the underlying application. These attributes allow the implementation of a wide spectrum of processing tasks. For instance, since the location information of the components is a probable part of the attribute sets, it can allow for tracking of components or groups of components by using standard tracking techniques. Furthermore, the detailed attributes of components such as wavelet representation of its time-scale characterization facilitate the application of more specific signal processing algorithms on the components of interest such as further denoising or statistical modelling. For instance, the post-processing block may be responsible for extraction of certain graph statistics such as degree, clustering coefficients, and orbit counts for analytics tasks. Some of these statistics are expressed below.

- For graph $G = (V, E)$ with vertices V and edges E, we define the neighbourhood for vertex v_i as $N_i = \{v_j : e_{ij} \in E \text{ or } e_{ji} \in E\}$. Let k_i be number of vertices in the neighbourhood N_i. Then, the clustering coefficient C_i for vertex v_i is:

$$C_i = \frac{\left|\{e_{jk} : v_j, v_k \in N_i,\ e_{jk} \in E\}\right|}{k_i\,(k_i - 1)} \tag{3.24}$$

- For graph $G = (V, E)$ orbit of node v is denoted by $Orb(G, v)$ and it is the set of nodes that can be mapped onto v by some automorphism $Aut(\cdot)$ of the graph, i.e.:

$$Orb(G, v) = \{w \in V : \exists \sigma \in Aut(G),\ \sigma(v) = w\} \tag{3.25}$$

In addition, this block may include techniques to quantify innovation over time, therefore it may implement metrics such as graph edit distance (GED) on original graph or maximum mean discrepancy (MMD) on extracted statistics to quantify the change across time. Some of these metrics are expressed below.

- For graphs G_1 and G_2 the Graph Edit Distance $\mathrm{GED}(G_1, G_2)$ is:

$$\mathrm{GED}(G_1, G_2) = \min_{(e_1,\ldots,e_n) \in P(G_1,G_2)} \sum_{i=1}^{n} c(e_i) \tag{3.26}$$

  where $P(G_1, G_2)$ is set of edit paths to transform G_1 into G_2 with elementary graph edit operations (i.e., vertex/edge insertion, deletion or substitution) and $c(\cdot)$ is cost function for each graph edit operation e_i.

- For two independent probability distributions P and Q, maximum mean discrepancy (MMD) can be expressed as:

$$\mathrm{MMD}(\mathcal{F}, P, Q) = \sup_{f \in \mathcal{F}} \Big|\Big|\mathbb{E}_{x \sim P}[f(x)] - \mathbb{E}_{y \sim Q}[f(y)]\Big|\Big| \tag{3.27}$$

  where $\mathcal{F}$ is class of functions and typically chosen as a unit ball in reproducing kernel Hilbert space (RKHS).

Finally, the *storage* and the *transmission* are tentative blocks that can be added to the framework depending on the application. In either case, an encoding scheme is utilized to compress the goal-filtered graphs and attributes. Then, the compressed semantic information can either be stored internally or passed forward to a transmission block where further channel coding operations can be performed prior to transmission.

The semantic signal processing framework described above can be adopted in the next generation of devices and communication networks. Although the types and implementation strategies of goals are previously discussed, the generation and dissemination of semantic goals warrant further discussion. Typically, global goals $\mathcal{G}_0$ for a semantic device can be defined either by human operators at the language level or by manufacturers at a hardware level. Alternatively, in a hierarchical network such as a Smart City application [193, 194], global goals can be defined at the highest level, and then they can be disseminated by mapping the global goals to the low-level device language by intermediate devices such as base stations. On the other hand, the generation of rapidly varying local goals $\mathcal{G}_0$ requires decision-making algorithms to be implemented at intermediate levels to optimize throughput and goal implementation. Possible approaches to the goal generation and dissemination problems address the so-called *effectiveness problem* described in Chapter 1, and constitute a future research direction for goal-oriented semantic signal processing.

The adoption of the proposed semantic signal processing framework in Fig. 3.4 will also have repercussions at a hardware level. The preprocessing and the semantic extraction stages can be implemented as part of a sensor subsystem designed to acquire signals of the desired modality such as audio or video signals. Such a sensor subsystem must provide its output in the proposed semantic structure so that the rest of the semantic processing chain can be implemented on its output. Therefore, the proposed semantic signal processing framework may form a basis for sensor standardization efforts for future applications of semantic signal processing and communication systems.

Chapter 4

Application of the Adopted Semantic Signal Processing Framework

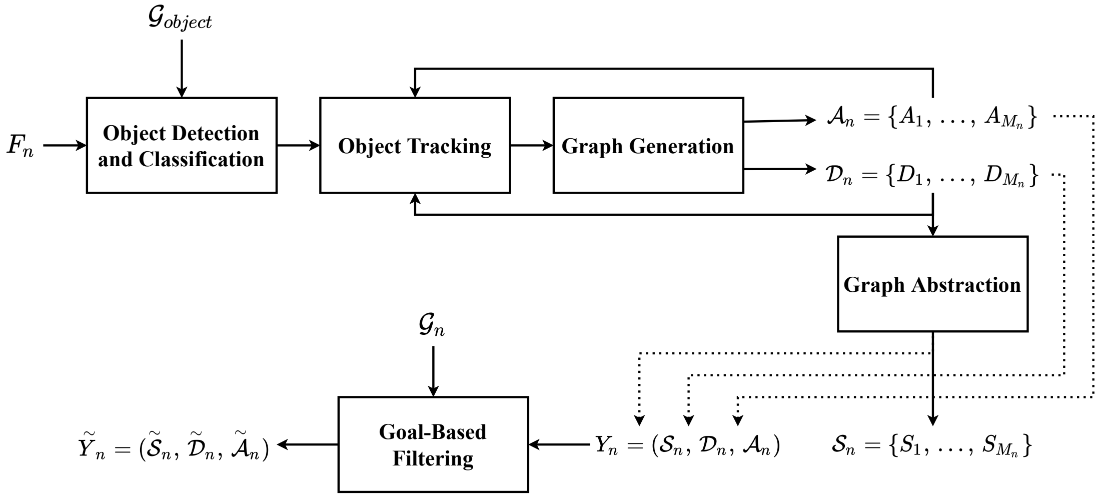
In this chapter, we provide concrete examples of the aforementioned goal-oriented semantic signal processing framework. First, we design a semantic extractor for a real-time computer vision application on a video-stream input. Next, we discuss applications of the framework with different signal modalities and provide a crude taxonomy. Then, we showcase the potential data compression capabilities of the designed semantic extractor and discuss efficient compression and coding strategies for transmission and storage. Finally, we demonstrate a way to identify points of significant innovation over extended periods of time using the embedded attributes at multiple-levels. It is important to note that these case studies are provided only as a proof-concept of the generality of the proposed framework, which has a great potential to be customized for many different applications outside of what is presented here.

4.1 Real-Time Semantic Processing of Video-Stream Data

In this section, we demonstrate the use of the proposed semantic signal processing framework on real-time video signals via the architecture shown in Fig. 4.1. An input sequence of frames denoted as F_n, with n as the discrete time index, is fed into a detection and classification block, and the detected entities in each frame are filtered out via a global goal $\mathcal{G}_{object}$. The remaining *globally-interesting* objects and extracted predicates among them are tracked across frames using their position and velocity information provided by the object tracking block. Based on the tracks, a *multi-graph instance representation* $\mathcal{D}_n = \{D_1, \ldots, D_{M_n}\}$ composed of M_n disconnected subgraphs and a corresponding attribute set $\mathcal{A}_n = \{A_1, \ldots, A_{M_n}\}$ are generated. The detailed semantic representation for each frame is updated in accordance with the tracking unit's updates. In this specific application, the *multi-graph class representation* $\mathcal{S}_n$ is constructed via a post-processing step as a higher-level abstraction and summary of $\mathcal{D}_n$ in the Graph Abstraction block. Finally, the complete semantic description $Y_n = (\mathcal{S}_n, \mathcal{D}_n, \mathcal{A}_n)$ is generated. The local goals $\mathcal{G}_n$ prune the semantic description Y_n to generate the goal-oriented semantic description $\widetilde{Y}_n$. Note that, in this example, we are considering fast algorithms that are available for real-time implementation. Alternatively, for delay-tolerant (offline) applications, scene graph generator models [109] can be used directly in place of the first three blocks. A detailed description of each block and the algorithms employed are discussed in the following.

4.1.1 Object Detection

Taking into account the real-time processing requirements, we use a pretrained YOLOv4-CSP [19] model on COCO dataset [196] for our object detection module since it achieves greater accuracy compared to the other models [18, 197] under the latency requirements of a typical video signal with 24 frames-per-second.



$\mathcal{G}_{object}$
F_n
Object Detection and Classification
Object Tracking
Graph Generation
$\mathcal{A}_n = \{A_1, \ldots, A_{M_n}\}$
$\mathcal{D}_n = \{D_1, \ldots, D_{M_n}\}$
Graph Abstraction
$\mathcal{G}_n$
Goal-Based Filtering
$\widetilde{Y}_n = (\widetilde{\mathcal{S}}_n, \widetilde{\mathcal{D}}_n, \widetilde{\mathcal{A}}_n)$
$Y_n = (\mathcal{S}_n, \mathcal{D}_n, \mathcal{A}_n)$
$\mathcal{S}_n = \{S_1, \ldots, S_{M_n}\}$

Figure 4.1: Semantic processing architecture for real-time video signals. Objects are detected and classified at each frame F_n by YOLOv4-CSP [19]. For temporal extension, detected objects are tracked across the frames using DeepSORT [195]. $\mathcal{S}_n$ is obtained by performing a summarization of $\mathcal{D}_n$, i.e, simply drawing a crude version of $\mathcal{S}_n$ by only keeping the class information of the nodes. Finally, the complete semantic description Y_n consists of the generated graph $\mathcal{D}_n$, its abstract version $\mathcal{S}_n$ and its attribute set $\mathcal{A}_n$. The resulting semantic description Y_n is then filtered via local goals $\mathcal{G}_n$.

Consequently, our semantic graph language contains $N_C = 80$ semantic component classes, i.e., $\mathcal{C} = \{c_1, \ldots, c_{80}\}$, corresponding to the object categories in the COCO dataset (c_1 : person, c_2 : bike, c_{18} : dog, etc.). Omitting the time index n for simplicity, YOLOv4-CSP provides each detected object in the form $o_i = (\boldsymbol{b}_i, \boldsymbol{y}_i)$, where $\boldsymbol{b}_i \in \mathbb{R}^4$ contains the position of the object according to its bounding box and $\boldsymbol{y}_i \in \mathbb{R}^{N_C}$ contains the category confidence scores. Here, we determine the classes of detected semantic instances via $\boldsymbol{y}_i$ and represent each object in the form $o_i \equiv (c_a, i)$. This representation simply indicates that the object o_i is an instance of component class c_a with a unique identifier index i where:

$$a = \arg\max_{c=1,\ldots,80} \boldsymbol{y}_i[c] \tag{4.1}$$

To put it another way, (c_a, i) is the pointer of object o_i so that $\mathcal{C}^D$ as defined in (3.9) stores the detected objects as

$$\mathcal{C}^D = \left\{ (c_a, i) \equiv o_i \,\middle|\, c_a \in \mathcal{C} \right\} \tag{4.2}$$

An example output of object detection/classification module is shown in Fig. 4.2. The three detected objects belong to the semantic component classes c_1 : person, c_{18} : dog, and c_{19} : horse, i.e., the set of detected objects is simply $\mathcal{C}^D = \{o_1 \equiv (c_1, 1),\ o_2 \equiv (c_{19}, 2),\ o_3 \equiv (c_{18}, 3)\}$.

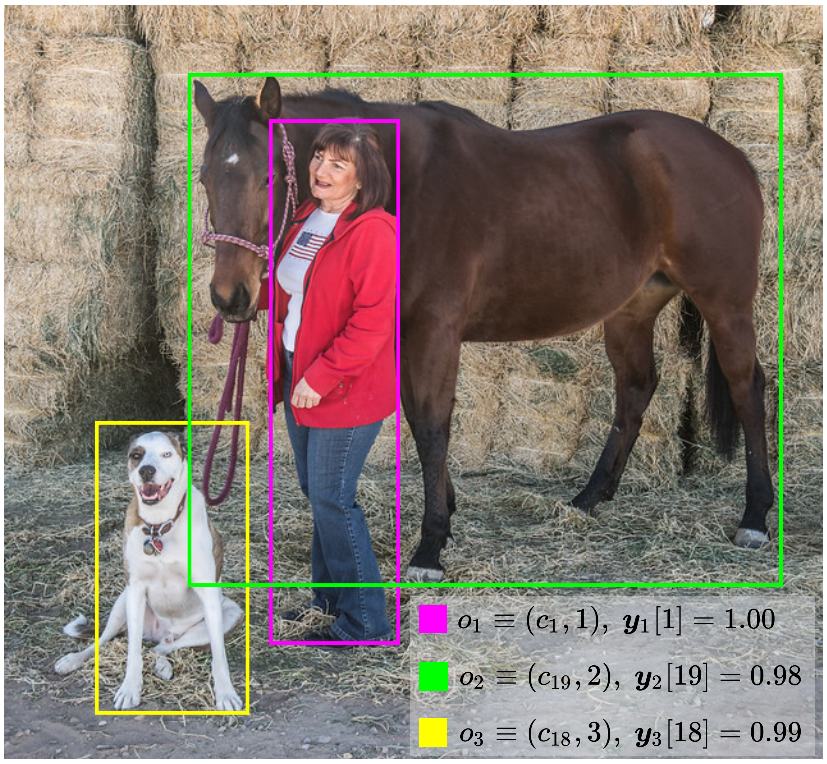

$o_1 \equiv (c_1, 1),\ \boldsymbol{y}_1[1] = 1.00$
$o_2 \equiv (c_{19}, 2),\ \boldsymbol{y}_2[19] = 0.98$
$o_3 \equiv (c_{18}, 3),\ \boldsymbol{y}_3[18] = 0.99$

Figure 4.2: An example of detected objects by YOLOv4-CSP [19] model. Predicted bounding boxes $\boldsymbol{b}_i$ are shown and maximum element of class confidence scores $\max\{\boldsymbol{y}_i[c]\}$ are listed.

In this example, we define our predicate set as $\mathcal{P} = \{p_0, p_1, p_2\}$ where p_0 : *null*, p_1 : *moving-together*, and p_2 : *conjunct*. The *null* predicate p_0 is connected to isolated objects by default. The predicate p_1 : *moving-together* is placed between adjacent objects moving towards to the same direction with similar velocities, whereas p_2 : *conjunct* predicate is placed between objects which have significantly overlapping bounding boxes and possibly possessive relationships, e.g., a person with their bike or bag. Note that all predicates are defined in such a way that all the edges start from the detected components and end at the predicates. This is due to the symmetric nature of the predicates in this specific application. Moreover, it should be noted that these predicates are only defined due to the simplicity of their inference, and the predicate set can be enriched further by extracting other relationships among the detected instances. For instance, if a *person* and a *bicycle* are moving towards the same direction with a significant overlap between their bounding boxes, then the predicate between these objects can be set to *riding*. With a similar reasoning for objects *person* and *car*, a *driving* predicate can be generated. Analogous to the objects, we denote the predicates as $e_k \equiv (p_w, k)$ with k being a unique identifier index. Predicates e_k are placed between some objects $o_i \equiv (c_a, i)$ and $o_j \equiv (c_b, j)$, where the set $\mathcal{P}^D$ as defined in (3.9) stores the detected predicates as

$$\mathcal{P}^D = \left\{ (p_w, k) \equiv e_k \,\middle|\, p_w \in \mathcal{P} \right\} \tag{4.3}$$

4.1.2 Object Tracking

To identify the temporal and spatial changes within frames we perform tracking on the detected objects. Just like the object detection module, we require a tracker with a fast inference capability. Therefore, we use the DeepSORT [195] algorithm for tracking individual objects in the video stream. Specifically, DeepSORT utilizes a Kalman filter [198] to recursively predict future positions of the objects. We denote the state vector of each object $o_i \equiv (c_a, i)$ with $\boldsymbol{m}_i$, which contains the parameters of its bounding box $\boldsymbol{b}_i$ and its respective velocities as

$$\boldsymbol{m}_i = \left[x_i^c, y_i^c, w_i, h_i, \dot{x}_i^c, \dot{y}_i^c, \dot{w}_i, \dot{h}_i \right] \tag{4.4}$$

where x_i^c, y_i^c, w_i, h_i are the center coordinates, the width, and the height of the bounding box $\boldsymbol{b}_i$, respectively. We utilize the following *linear constant velocity* state transition model for the internal Kalman filter (initialized for each detected object) of DeepSORT tracker where we drop the object identifier subscript i for simplicity and denote the discrete time with superscript t:

$$\begin{bmatrix} x^c \\ y^c \\ w \\ h \\ \dot{x}^c \\ \dot{y}^c \\ \dot{w} \\ \dot{h} \end{bmatrix}_{t+1} = \begin{bmatrix} 1 & 0 & 0 & 0 & d_t & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & d_t & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & d_t & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & d_t \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^c \\ y^c \\ w \\ h \\ \dot{x}^c \\ \dot{y}^c \\ \dot{w} \\ \dot{h} \end{bmatrix}_{t} + \begin{bmatrix} n_{x^c} \\ n_{y^c} \\ n_w \\ n_h \\ n_{\dot{x}^c} \\ n_{\dot{y}^c} \\ n_{\dot{w}} \\ n_{\dot{h}} \end{bmatrix}_{t} \tag{4.5}$$

$$\begin{bmatrix} z_{x^c} \\ z_{y^c} \\ z_w \\ z_h \end{bmatrix}_{t} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x^c \\ y^c \\ w \\ h \\ \dot{x}^c \\ \dot{y}^c \\ \dot{w} \\ \dot{h} \end{bmatrix}_{t} + \begin{bmatrix} v_{x^c} \\ v_{y^c} \\ v_w \\ v_h \end{bmatrix}_{t} \tag{4.6}$$

equivalently in matrix form as:

$$\boldsymbol{m}^{t+1} = \boldsymbol{K}\boldsymbol{m}^t + \boldsymbol{n}^t \tag{4.7}$$

$$\boldsymbol{z}^t = \boldsymbol{H}\boldsymbol{m}^t + \boldsymbol{v}^t \tag{4.8}$$

where d_t is velocity update coefficient that depends on operation FPS of underlying application, $\boldsymbol{K}$ is state transition matrix, $\boldsymbol{H}$ is observation/measurement matrix, $\boldsymbol{m}^t$ is state vector, $\boldsymbol{z}^t$ is measurement vector (i.e., noisy bounding box coordinates), $\boldsymbol{n}^t$ is process and $\boldsymbol{v}^t$ is measurement noise respectively. We assume $\boldsymbol{n}^t$ and $\boldsymbol{v}^t$ are zero-mean independent Gaussian random vectors:

$$\boldsymbol{n}^t \sim \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{Q}^t\right), \boldsymbol{v}^t \sim \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{R}^t\right) \tag{4.9}$$

where $\boldsymbol{Q}^t$ and $\boldsymbol{R}^t$ diagonal covariance matrices which their entries may change at each time step with respect to predictions as:

$$\begin{aligned}\boldsymbol{Q}^t = diag\Big(& (\sigma_{pos1} \cdot \hat{m}^{t-1|t-1}[2])^2,\ (\sigma_{pos1} \cdot \hat{m}^{t-1|t-1}[3])^2, \\ & (\sigma_{pos2} \cdot \hat{m}^{t-1|t-1}[2])^2,\ (\sigma_{pos2} \cdot \hat{m}^{t-1|t-1}[3])^2, \\ & (\sigma_{vel} \cdot \hat{m}^{t-1|t-1}[2])^2,\ (\sigma_{vel} \cdot \hat{m}^{t-1|t-1}[3])^2, \\ & (\sigma_{vel} \cdot \hat{m}^{t-1|t-1}[2])^2,\ (\sigma_{vel} \cdot \hat{m}^{t-1|t-1}[3])^2 \Big)\end{aligned} \tag{4.10}$$

$$\begin{aligned}\boldsymbol{R}^t = diag\Big(& (\sigma_{mea} \cdot \hat{m}^{t-1|t-1}[2])^2,\ (\sigma_{mea} \cdot \hat{m}^{t-1|t-1}[3])^2, \\ & (\sigma_{mea} \cdot \hat{m}^{t-1|t-1}[2])^2,\ (\sigma_{mea} \cdot \hat{m}^{t-1|t-1}[3])^2 \Big)\end{aligned} \tag{4.11}$$

We initialize the state estimate $\hat{\boldsymbol{m}}^0$ with bounding box coordinates provided by the YOLOv4-CSP model and set the unobserved velocities to 0. Furthermore, the Kalman filter process can be expressed with the following recursions:

$$\hat{\boldsymbol{m}}^{t|t-1} = \boldsymbol{K}\hat{\boldsymbol{m}}^{t-1|t-1} \tag{4.12}$$

$$\boldsymbol{A}^{t|t-1} = \boldsymbol{K}\boldsymbol{A}^{t-1|t-1}\boldsymbol{K}^\top + \boldsymbol{Q}^t \tag{4.13}$$

$$\boldsymbol{B}^t = \boldsymbol{A}^{t|t-1}\boldsymbol{H}^\top \left(\boldsymbol{H}\boldsymbol{A}^{t|t-1}\boldsymbol{H}^\top + \boldsymbol{R}^t\right)^{-1} \tag{4.14}$$

$$\hat{\boldsymbol{m}}^{t|t} = \hat{\boldsymbol{m}}^{t|t-1} + \boldsymbol{B}^t \left(\boldsymbol{z}^t - \boldsymbol{H}\hat{\boldsymbol{m}}^{t|t-1}\right) \tag{4.15}$$

$$\boldsymbol{A}^{t|t} = \left(\boldsymbol{I} - \boldsymbol{B}^t\boldsymbol{H}\right)\boldsymbol{A}^{t|t-1} \tag{4.16}$$

where we initialize the covariance matrix $\boldsymbol{A}^0$ again as a diagonal matrix (i.e., $2 \cdot \boldsymbol{Q}^0$). At each time step t, the Kalman filter first predicts the prior state vector estimate $\hat{\boldsymbol{m}}^{t|t-1}$ as in (4.12) and the prior covariance matrix $\boldsymbol{A}^{t|t-1}$ as in (4.13). Next, the Kalman filter calculates the Kalman gain $\boldsymbol{B}^t$ using (4.14). Finally, given the current observation $\boldsymbol{z}^t$ and the calculated Kalman gain $\boldsymbol{B}^t$, it updates the posterior state vector estimate $\hat{\boldsymbol{m}}^{t|t}$ and posterior covariance matrix $\boldsymbol{A}^{t|t}$ using (4.15) and (4.16) respectively. The tracking principle of DeepSORT algorithm is shown in Fig. 4.3.

To improve the tracking performance under challenging scenarios such as occlusion, non-stationary camera and multiple viewpoints, DeepSORT further utilizes

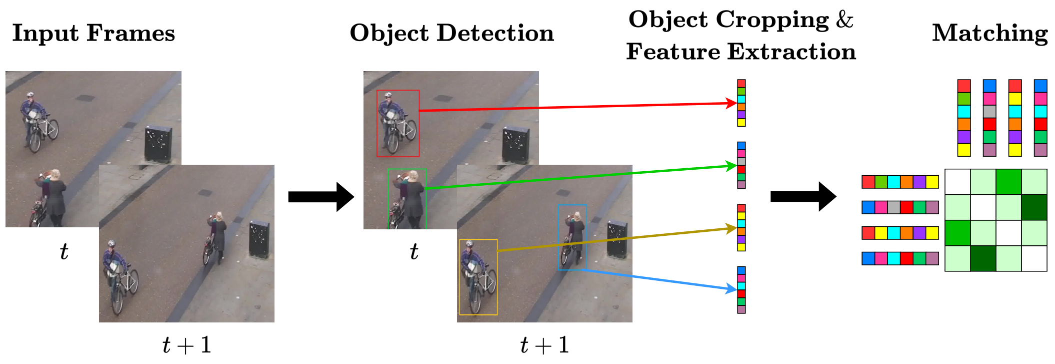


Figure 4.3: Tracking by detection paradigm adopted by DeepSORT. Input frames are processed by a object detection model YOLOv4-CSP, then, detected objects are cropped and fed into another small CNN trained on re-identification task for visual feature extraction. The algorithm compares feature vectors of detected objects and existing tracks. Next, it assigns a cost to each (detection, track) pair to fill the entries of a cost matrix. Finally, the matching is performed by an assignment linear programming solver using the generated cost matrix, and DeepSORT updates track attributes for the next frame.

convex combination of deep cosine association metric [199] and Mahalanobis distance between observed and predicted bounding boxes coordinates. Particularly, each detected object $o_i \equiv (c_a, i)$ is cropped from the original frame based on the bounding boxes $\boldsymbol{b}_i$ and passed through a CNN (trained on re-identification task) to be represented as unit-norm vectors $\boldsymbol{r}_i \in \mathbb{R}^{128}$, $\|\boldsymbol{r}_i\|_2 = 1$, as illustrated in Figs. 4.4 and 4.5. We refer to $\boldsymbol{r}_i$'s as the feature vectors. In fact, for each tracked object $o_i \equiv (c_a, i)$, we store multiple feature vectors across frames $\left\{\boldsymbol{r}_i^{(l)}\right\}_{l=1}^{T_i}$ where $\boldsymbol{r}_i^{(l)}$ stands for the l-th feature vector of the tracked object $o_i \equiv (c_a, i)$, and T_i is number of times the object o_i is observed. It is important to mention that, in certain settings we only store the most recent feature vector by utilizing exponentially moving average (EMA) to gradually update stored features as:

$$\tilde{\boldsymbol{r}}_i^{(t+1)} = \beta\tilde{\boldsymbol{r}}_i^{(t)} + (1-\beta)\boldsymbol{r}_i^{(t+1)} \tag{4.17}$$

Specifically, we utilize EMA for innovation detection at attribute level which introduced in Section 4.4 while speeding up the track association process. The parameters utilized in our simulation is listed in Table 4.1.

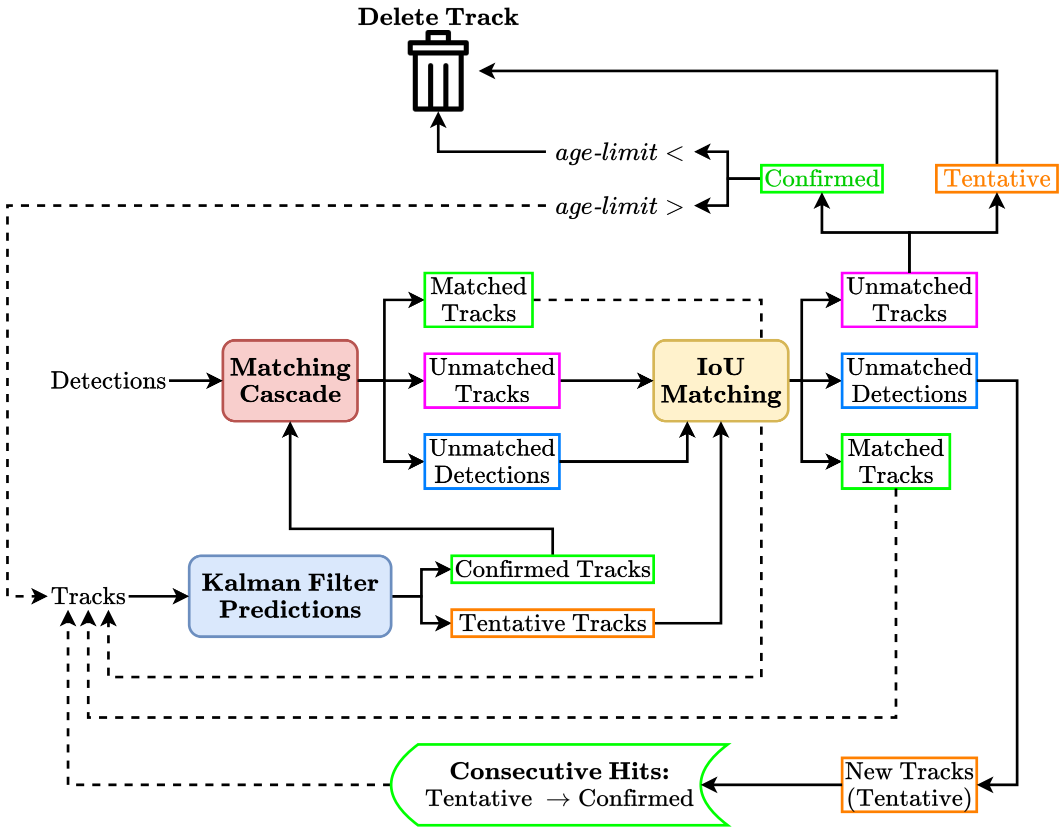


Figure 4.4: Schematic of DeepSORT tracking algorithm. At each frame, states of the tracked objects are updated via Kalman filter. The Matching Cascade (see Fig. 4.5 for details) performs first association of *detections* and existing *tracks* using both extracted visual features and state predictions. The second association between *unmatched tracks* and *unmatched detections* is performed with respect to IoU scores of the bounding boxes and the estimations. Then, remaining *unmatched tracks* are deleted if their status are confirmed but expired or tentative. Finally, *unmatched detections* are initialized as *new tentative tracks* which their status are changed to *confirmed* with observed consecutive hits and deleted otherwise.

Table 4.1: DeepSORT parameters used in our simulations.

DeepSORT Simulation Parameters	
Velocity coefficient d_t	1 (for 30 FPS)
Initial state estimate $\hat{\boldsymbol{m}}^0$	$[x^c, y^c, w, h, 0, 0, 0, 0]$
Initial covariance estimate $\boldsymbol{A}^0$	$2 \cdot \boldsymbol{Q}^0$
Noise std coefficient σ_{pos1}	0.1
Noise std coefficient σ_{pos2}	0.05
Noise std coefficient σ_{vel}	0.00625
Noise std coefficient σ_{mea}	0.05
Track deletion *age-limit*	150 (5s for 30 FPS)
Track confirmation *num-hits*	5
Cosine distance gating threshold τ^C	0.2
Mahalanobis distance gating threshold τ^C	6
Maximum feature dictionary size T_i	50 (or 1 for EMA)
Cost matrix coefficient λ	0.5 (or 0.7 for EMA)
EMA coefficient β	0.9

4.1.3 Graph Generation and Graph Abstraction

We employ the state $\boldsymbol{m}_i$, feature vectors $\left\{\boldsymbol{r}_i^{(l)}\right\}_{l=1}^{T_i}$ and the cropped RGB image $I_i \in \mathbb{R}^{w_i \times h_i \times 3}$ as the multi-level attributes of object $o_i \equiv (c_a, i)$ with $L = 3$ number of levels as defined in (3.16), i.e.:

$$\Theta(c_a, i) = \left\{\boldsymbol{\theta}^{(1)}(c_a, i),\, \boldsymbol{\theta}^{(2)}(c_a, i),\, \boldsymbol{\theta}^{(3)}(c_a, i)\right\} = \left\{\boldsymbol{m}_i,\, \{\boldsymbol{r}_i^{(l)}\}_{l=1}^{T_i},\, I_i\right\} \tag{4.18}$$

Note that, it is possible that different categories can have a different number of attribute levels L_a; however, for our experiments, we set $L_a = 3$, $\forall a$, for simplicity. It is important to note that, even though in our experiments, $\boldsymbol{\theta}^{(3)}(c_a, i)$ is set to the raw data I_i itself, it can also be chosen as any multi-scale representation of I_i such as its wavelet decomposition or discrete cosine transform. As an example, attributes for a detected *person* object is illustrated in Fig. 4.6.

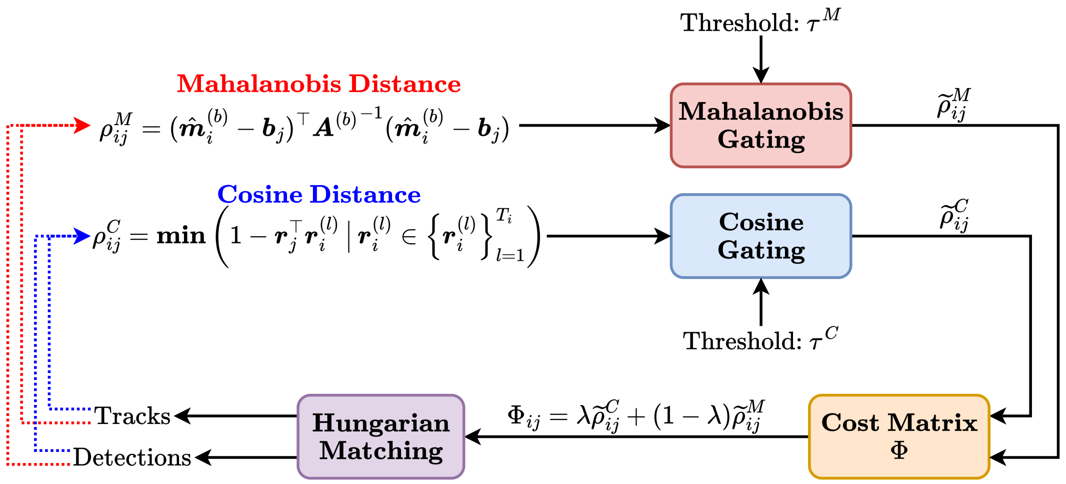

Threshold: τ^M
Mahalanobis Distance
$\rho^M_{ij} = (\hat{\boldsymbol{m}}^{(b)}_i - \boldsymbol{b}_j)^\top \boldsymbol{A}^{(b)^{-1}} (\hat{\boldsymbol{m}}^{(b)}_i - \boldsymbol{b}_j)$
Mahalanobis Gating
$\widetilde{\rho}^M_{ij}$
Cosine Distance
$\rho^C_{ij} = \mathbf{min}\left(1 - \boldsymbol{r}_j^\top \boldsymbol{r}_i^{(l)} \,|\, \boldsymbol{r}_i^{(l)} \in \left\{\boldsymbol{r}_i^{(l)}\right\}_{l=1}^{T_i}\right)$
Cosine Gating
$\widetilde{\rho}^C_{ij}$
Threshold: τ^C
Tracks
Detections
Hungarian Matching
$\Phi_{ij} = \lambda\widetilde{\rho}^C_{ij} + (1-\lambda)\widetilde{\rho}^M_{ij}$
Cost Matrix Φ

Figure 4.5: First association process of DeepSORT via Matching Cascade. For each track$_i$ and detection$_j$ pair, two distance metrics are computed, namely, Mahalanobis distance ρ^M_{ij} of bounding box coordinates and cosine distance ρ^C_{ij} of feature vectors. We denote the entries that corresponds to bounding box coordinates of estimates with (b) superscript as in state estimate $\hat{\boldsymbol{m}}^{(b)}_i$ and covariance matrix $\boldsymbol{A}^{(b)^{-1}}$, and use $\boldsymbol{b}_j$ for bounding box coordinates of detection$_j$. For feature dictionary configuration, we use the smallest cosine distance among all extracted feature vectors in $\left\{\boldsymbol{r}_i^{(l)}\right\}_{l=1}^{T_i}$ of track$_i$. If the computed distance values exceed the predefined gating thresholds τ^M or τ^C, we replace them with infinity to prevent unfavourable matching outcomes. The entries of cost matrix consist of convex combination of gated Mahalanobis $\widetilde{\rho}^M_{ij}$ and gated cosine $\widetilde{\rho}^C_{ij}$ distances. Finally, the resulting cost matrix is used by Hungarian assignment algorithm [200] to match tracks and detections.

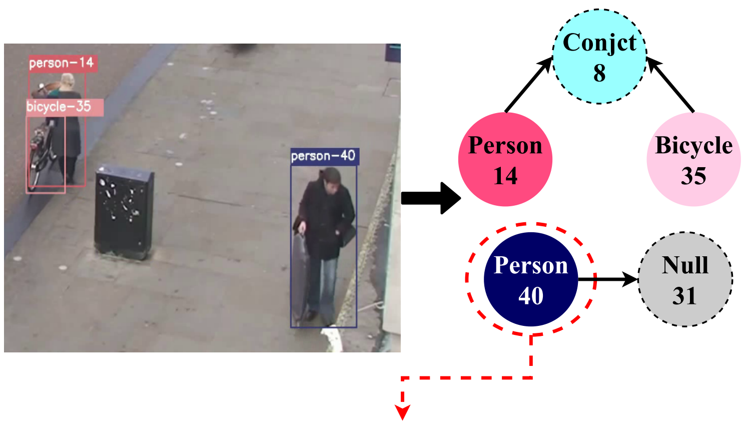
The exploitation of object positions and velocities enables us to determine types of predicates between different object pairs. We identify the type of a predicate between objects $o_i \equiv (c_a, i)$ and $o_j \equiv (c_b, j)$ by using their states $\boldsymbol{m}_i$ and $\boldsymbol{m}_j$. The following three metrics are defined for predicate detection:

$$d^{(1)}(i,j) = \frac{\left\|(x^c_i, y^c_i) - (x^c_j, y^c_j)\right\|_2^2}{\sqrt{w_i \cdot h_i \cdot w_j \cdot h_j}} \tag{4.19}$$

$$d^{(2)}(i,j) = \frac{(\dot{x}^c_i, \dot{y}^c_i)^\top (\dot{x}^c_j, \dot{y}^c_j)}{||(\dot{x}^c_i, \dot{y}^c_i)||_2 \, ||(\dot{x}^c_j, \dot{y}^c_j)||_2} = cos\left((\dot{x}^c_i, \dot{y}^c_i),\, (\dot{x}^c_j, \dot{y}^c_j)\right) \tag{4.20}$$

$$d^{(3)}(i,j) = IoU\left(\boldsymbol{b}_i, \boldsymbol{b}_j\right) \tag{4.21}$$

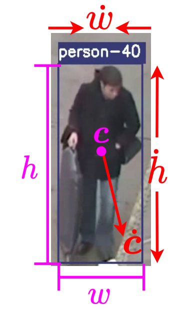
Here, $d^{(1)}(i,j)$ corresponds to scaled Euclidean (2-norm) distance between center



$$\Theta(c_1, 40) = \{\theta^1(c_1, 40),\, \theta^2(c_1, 40),\, \theta^3(c_1, 40)\}$$

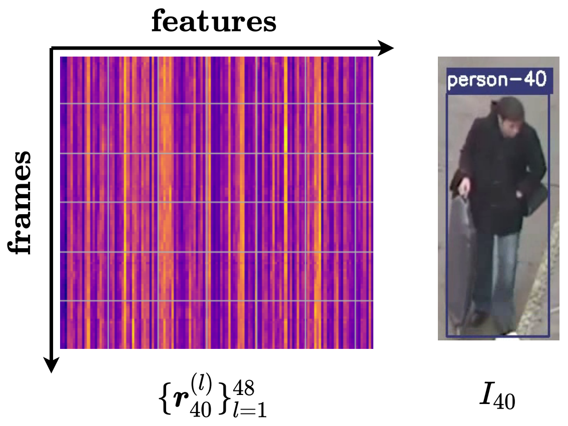
$$= \{\boldsymbol{m}_{40},\, \{\boldsymbol{r}_{40}^{(l)}\}_{l=1}^{48},\, I_{40}\}$$

(a) Frame under process, generated instance graph, and corresponding attribute set for Person-40.



$$\boldsymbol{m}_{40} = \Big[\overbrace{1795,\ 561.7}^{\boldsymbol{c}=(x_c, y_c)},\ \overbrace{98.94}^{w},\ \overbrace{243.8}^{h},\ \underbrace{-0.777,\ 2.098}_{\dot{\boldsymbol{c}}=(\dot{x}_c, \dot{y}_c)},\ \underbrace{-0.336}_{\dot{w}},\ \underbrace{0.441}_{\dot{h}}\Big]$$

(b) Lowest level attribute set corresponding to the state vector $\boldsymbol{m}$ of Person-40. Units are in pixels for position and pixels-per-frame for velocities.



(c) Second and third level attributes of Person-40. The second level includes a set of collected feature vectors across 48 frames with $T_{40} = 48$. The third level includes the cropped RGB image I_{40}, although alternative encoded versions of I_{40} can also be used.

Figure 4.6: Illustration of attributes for a *person* component instance.

coordinates of the positions, $d^{(2)}(i,j)$ is the cosine similarity between the velocities, while $d^{(3)}(i,j)$ is the intersection-over-union of bounding boxes of objects o_i and o_j. Specifically, for a predicate $e_k \equiv (p_1, k)$, i.e., *moving-together* between objects $o_i \equiv (c_a, i)$ and $o_j \equiv (c_b, j)$, across multiple frames we require:

$$\mathbb{1}\left(d^{(1)}(i,j) \le \tau^{(1)}\right) \cdot \mathbb{1}\left(d^{(2)}(i,j) \ge \tau^{(2)}\right) \tag{4.22}$$

and similarly, for $e_k \equiv (p_2, k)$, i.e., *conjunct*, we require:

$$\mathbb{1}\left(d^{(1)}(i,j) \le \tau^{(1)}\right) \cdot \mathbb{1}\left(d^{(2)}(i,j) \ge \tau^{(2)}\right) \cdot \mathbb{1}\left(d^{(3)}(i,j) \ge \tau^{(3)}\right) \tag{4.23}$$

for predetermined thresholds $\tau^{(1)}$, $\tau^{(2)}$ and $\tau^{(3)}$, where $\mathbb{1}(\cdot)$ denotes the indicator function. Furthermore, even though it is not crucial for our application, we define attributes for a predicate $e_k \equiv (p_w, k)$, $p_w \in \mathcal{P} \setminus \{p_0\}$ as:

$$\Theta(p_w, k) = \left\{\boldsymbol{\theta}^{(1)}(p_w, k)\right\} = \left\{\left[d^{(1)}(i,j),\, d^{(2)}(i,j),\, d^{(3)}(i,j)\right]\right\} \tag{4.24}$$

with $L_w = 1$, $\forall w$. Here, $\Theta(p_w, k)$ consists of a single-level attribute, i.e., a vector that contains metric evaluations between objects o_i and o_j. Note that unlike the other predicates belonging to classes in $\mathcal{P} \setminus \{p_0\}$ which are connected to two objects $o_i \equiv (c_a, i)$ and $o_j \equiv (c_b, j)$, a null predicate $e_l \equiv (p_0, l)$ is only connected to $o_u \equiv (c_f, u)$, for an isolated object o_u. For null predicates, the attribute set is simply defined as the the empty set, i.e., $\Theta(p_0, l) = \varnothing$.

Given the set of detected objects $\mathcal{C}^D$ and predicates $\mathcal{P}^D$, we denote our detected connection set $\mathcal{E}^D$ with triplets and pairs as:

$$\begin{aligned}\mathcal{E}^D = &\left\{(o_i,\, e_k,\, o_j) \,\middle|\, o_i, o_j \in \mathcal{C}^D,\, e_k \in \mathcal{P}^D,\, p_w \in \mathcal{P} \setminus \{p_0\}\right\} \\ &\bigcup \left\{(o_u,\, e_l) \,\middle|\, o_u \in \mathcal{P}^D,\, e_l \equiv (p_0, l)\right\}\end{aligned} \tag{4.25}$$

In other words, the detected connection set $\mathcal{E}^D$ simply consists of *(object, predicate, object)* triplets and (*unaccompanied object, null predicate*) pairs. Consequently, the graph D is generated by the detected object, predicate, and edge sets as:

$$D = \left(\mathcal{C}^D,\, \mathcal{P}^D,\, \mathcal{E}^D\right) \tag{4.26}$$

We then define the companion attribute set A of graph D as:

$$A = \left\{ \Theta(c_a, i) \;\middle|\; o_i \equiv (c_a, i) \in \mathcal{C}^D \right\} \bigcup \left\{ \Theta(p_w, k) \;\middle|\; e_k \equiv (p_w, k) \in \mathcal{P}^D \right\} \tag{4.27}$$

which stores attributes for both object and predicate instances. Furthermore, we generate the class-level representation of instance graph D as defined in (3.4). The class description graph S is defined as:

$$S = (\mathcal{C}^S, \mathcal{P}^S, \mathcal{E}^S), \tag{4.28}$$

where

$$\mathcal{C}^S = \left\{ c_a \;\middle|\; o_i \equiv (c_a, i) \in \mathcal{C}^D \right\} \tag{4.29}$$

$$\mathcal{P}^S = \left\{ p_w \;\middle|\; e_k \equiv (p_w, k) \in \mathcal{P}^D \right\} \tag{4.30}$$

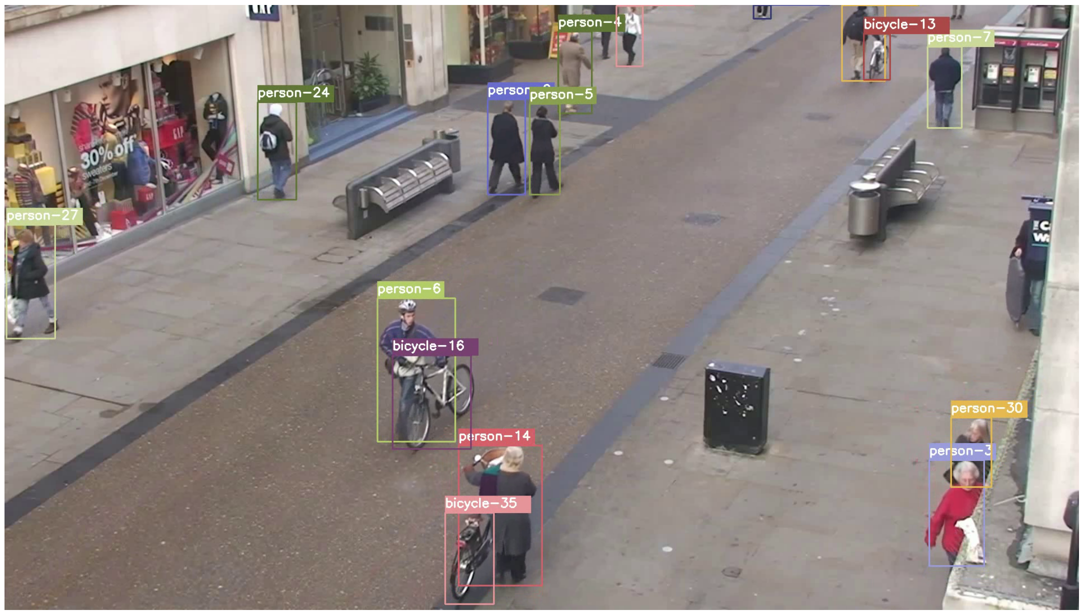
$$\mathcal{E}^S = \begin{array}{l} \left\{ (c_a,\, p_w,\, c_b) \;\middle|\; (o_i,\, e_k,\, o_j) \equiv ((c_a, i),\, (p_w, k)\, (c_b, j)) \in \mathcal{E}^D \right\} \\ \quad \bigcup \left\{ (c_f, p_0) \;\middle|\; (o_u,\, e_l) \equiv ((c_f, u),\, (p_0, l)) \in \mathcal{E}^D \right\} \end{array} \tag{4.31}$$

To facilitate a better understanding of the high-level abstraction provided by (4.28), we provide an illustrative example in Fig. 4.7. The class level graph generation is a summarization of the components and predicates in the instance level graph, i.e., it is a surjective function mapping D to S.

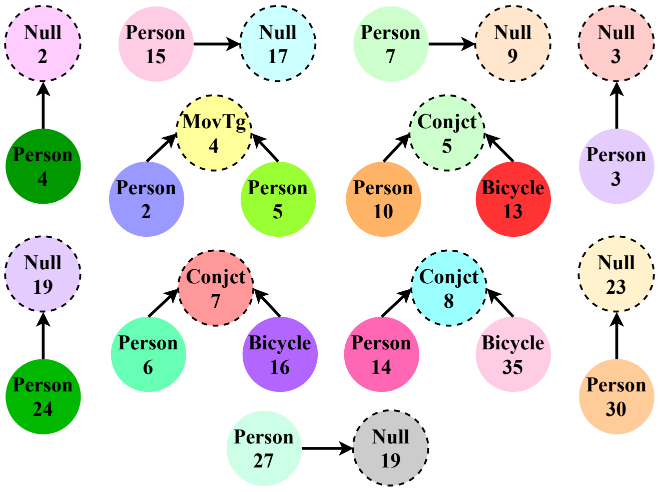
Finally, we define the semantic description Y as a triplet consisting of the class-level graph, the instance-level graph, and attribute supersets as

$$Y = (S, D, A). \tag{4.32}$$

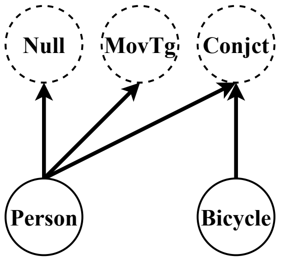
It should be noted that, while in Chapter 3, an instance level graph $\mathcal{D}$ is defined as union of disjoint subgraphs $\mathcal{D} = \{D_1, \ldots, D_m, \ldots D_M\}$, we use a single combined graph D to represent the whole scene. It is desirable to split $\mathcal{D}$ into *atomic graphs* for an easy processing of goals, as discussed in Chapter 3. That is, splitting of $\mathcal{D}$ into its disjoint subgraphs D_m (also called atomic graphs), can be performed during post-processing for this specific scenario. A similar splitting operation can also be applied to the attribute set $\mathcal{A} = \{A_1, \ldots, A_m, \ldots, A_M\}$ and class-level graph $\mathcal{S} = \{S_1, \ldots, S_m, \ldots, S_M\}$ where S_m denotes the abstraction of D_m. The splitting operation on $\mathcal{D}$ and its class level counterpart is illustrated in Fig. 4.8.



(a) Frame under process with detections.

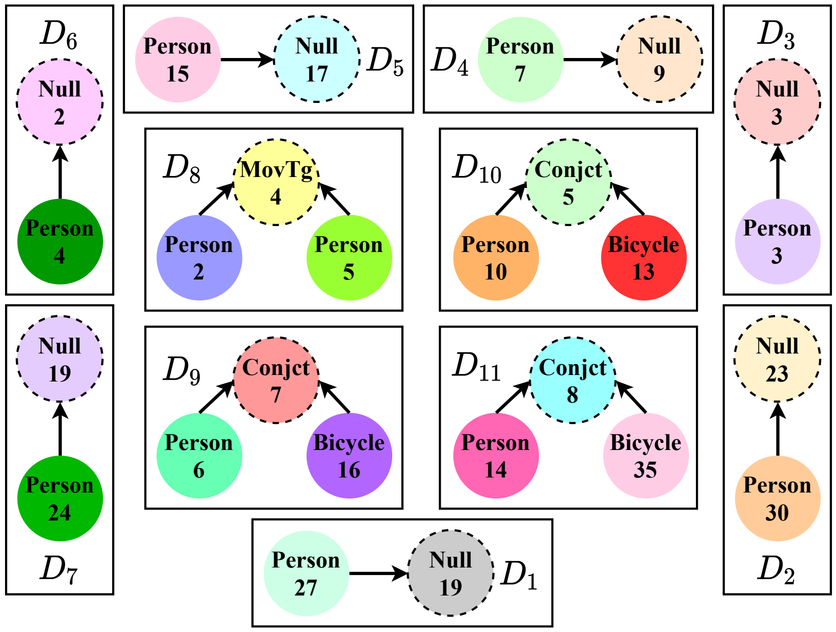


(b) Instance-level graph D. Object nodes are denoted with solid circles, predicates are denoted with dashed circles. Isolated objects are connected to the *null* predicate by default.

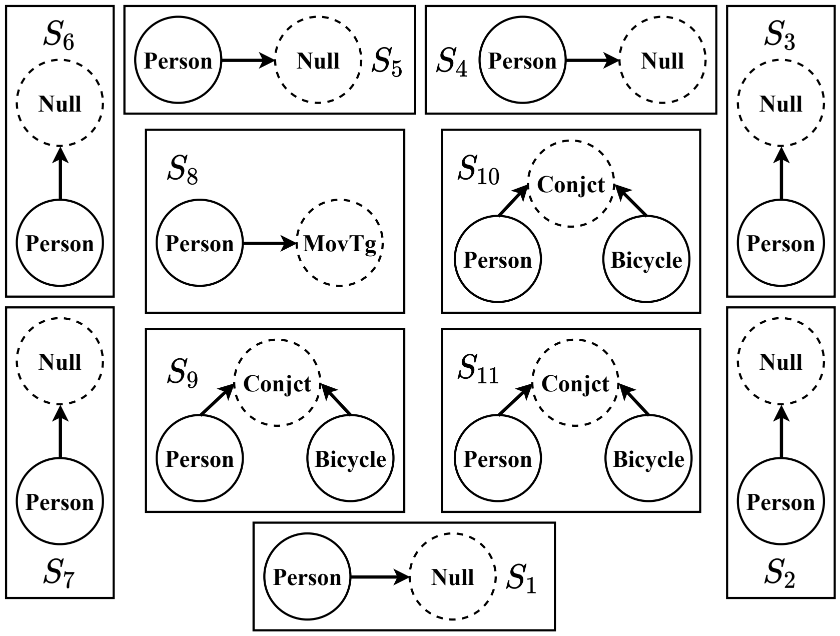


(c) Class-level graph S.

Figure 4.7: Illustration of the instance level and class level graph representations.



(a) Instance-level graph splitting at post-processing. Graph $\mathcal{D}$ in Fig. 4.7b is split into its atomic components D_m such that $\mathcal{D} = \{D_1, \ldots, D_{11}\}$. Corresponding attributes are split to atoms $\mathcal{A} = \{A_1, \ldots, A_{11}\}$ as well.



(b) Class-level abstractions are performed on disconnected subgraphs similar to the one in Fig. 4.7c so that we obtain $\mathcal{S} = \{S_1, \ldots, S_{11}\}$. Together with original graph $\mathcal{D}$ and attribute set $\mathcal{A}$, abstraction $\mathcal{S}$ forms the complete semantic description Y. Global goal-based filtering is performed using these high-level graphs.

Figure 4.8: Decomposing the instance-level and class-level graphs into their atomic components.

4.1.4 Goal-Based Filtering

We can also illustrate the goal-based filtering defined in Chapter 3 using the same scenario. Goal-based filtering operation allows for further distillation of the semantic information with respect to a specified goal and it enables reasoning over the semantic graph. In a sense, the filtering operation acts as a semantic parsing operator. The goal filtering operation for a small semantic graph with only two subgraphs is illustrated in Fig. 4.9. It should be observed that the queried graph patterns G^S and G^D as defined in (3.19), both include simple motifs with a single graph. Consequently, graph complexity and attribute complexity vectors reduce to scalars.

As illustrated in this part, the proposed goal-oriented semantic graph language and the signal processing framework can be incorporated into video signals for real-time applications. The semantic language allows for a structured and complete representation of the *meaningful* and *interesting* information embedded within the signal and allows for easy parsing of the information according to the desired goal.

4.1.5 Discussion on the Designed Semantic Extractor and its Alternatives

As stated in Section 4.1, for this case study on video stream data, we only considered causal semantic extractors with real-time processing capabilities. The utilization of object detection model allows us to identify signal components (i.e. objects) at each frame. The following object tracker ensures temporal continuity of the semantic signal components across frames in real-time. Furthermore, with the achieved temporal extension, we can identify our elementary predicates and fill the entries in our companion attributes for additional semantic information which allows us to detect significant changes in time called innovations (see Section 4.4). It is important to note that, to expand the predicate set one can utilize a knowledge base and exploit classes of detected components. For example, if there

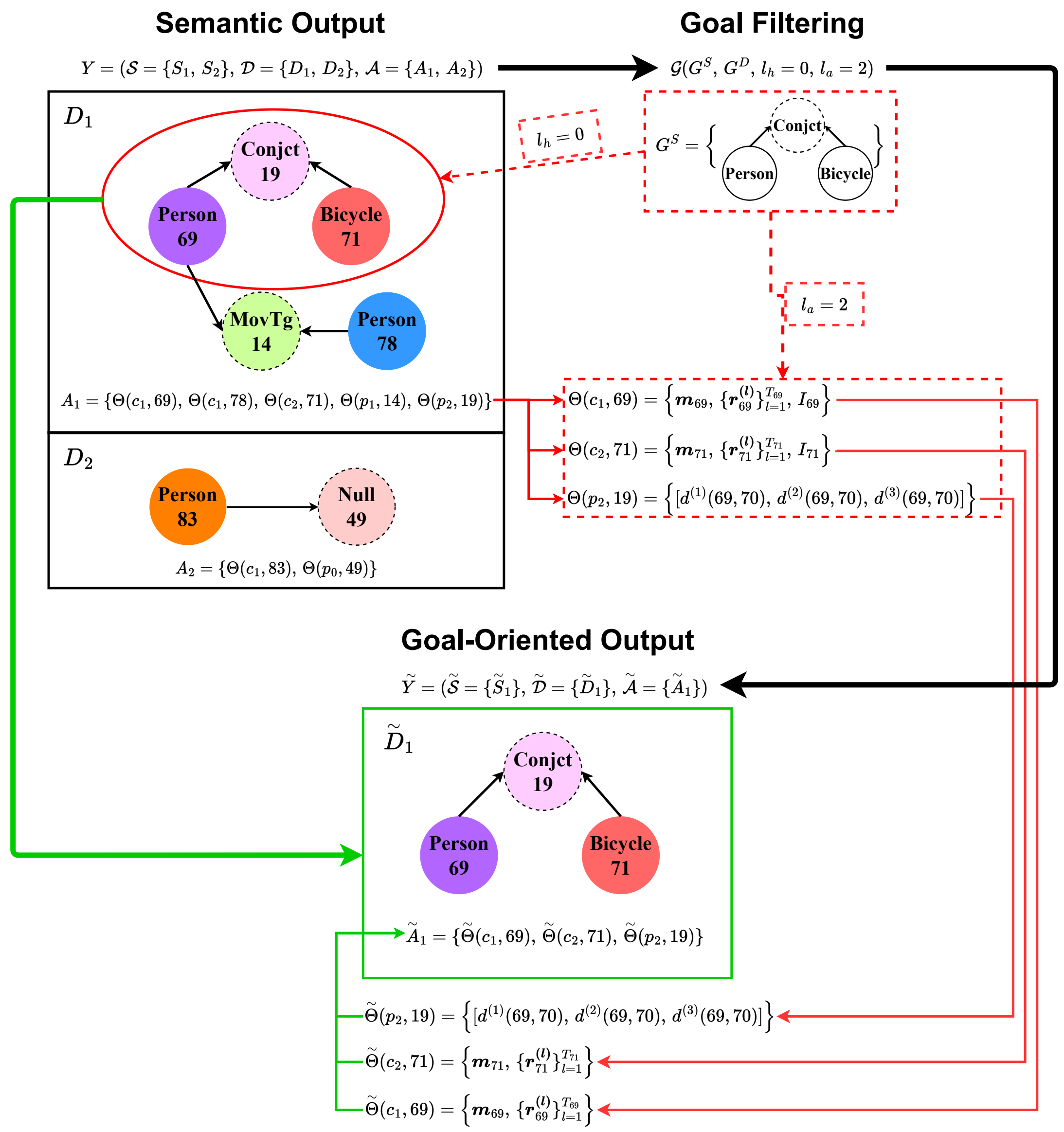

Semantic Output
Goal Filtering
$Y = (\mathcal{S} = \{S_1, S_2\}, \mathcal{D} = \{D_1, D_2\}, \mathcal{A} = \{A_1, A_2\})$
$\mathcal{G}(G^S, G^D, l_h = 0, l_a = 2)$
D_1
Conjct 19
Person 69
Bicycle 71
MovTg 14
Person 78
$l_h = 0$
$G^S = \{ \}$
Conjct
Person
Bicycle
$l_a = 2$
$A_1 = \{\Theta(c_1, 69), \Theta(c_1, 78), \Theta(c_2, 71), \Theta(p_1, 14), \Theta(p_2, 19)\}$
$\Theta(c_1, 69) = \left\{\boldsymbol{m}_{69}, \{\boldsymbol{r}_{69}^{(l)}\}_{l=1}^{T_{69}}, I_{69}\right\}$
$\Theta(c_2, 71) = \left\{\boldsymbol{m}_{71}, \{\boldsymbol{r}_{71}^{(l)}\}_{l=1}^{T_{71}}, I_{71}\right\}$
$\Theta(p_2, 19) = \left\{[d^{(1)}(69, 70), d^{(2)}(69, 70), d^{(3)}(69, 70)]\right\}$
D_2
Person 83
Null 49
$A_2 = \{\Theta(c_1, 83), \Theta(p_0, 49)\}$
Goal-Oriented Output
$\widetilde{Y} = (\widetilde{\mathcal{S}} = \{\widetilde{S}_1\}, \widetilde{\mathcal{D}} = \{\widetilde{D}_1\}, \widetilde{\mathcal{A}} = \{\widetilde{A}_1\})$
$\widetilde{D}_1$
Conjct 19
Person 69
Bicycle 71
$\widetilde{A}_1 = \{\widetilde{\Theta}(c_1, 69), \widetilde{\Theta}(c_2, 71), \widetilde{\Theta}(p_2, 19)\}$
$\widetilde{\Theta}(p_2, 19) = \left\{[d^{(1)}(69, 70), d^{(2)}(69, 70), d^{(3)}(69, 70)]\right\}$
$\widetilde{\Theta}(c_2, 71) = \left\{\boldsymbol{m}_{71}, \{\boldsymbol{r}_{71}^{(l)}\}_{l=1}^{T_{71}}\right\}$
$\widetilde{\Theta}(c_1, 69) = \left\{\boldsymbol{m}_{69}, \{\boldsymbol{r}_{69}^{(l)}\}_{l=1}^{T_{69}}\right\}$

Figure 4.9: A goal-based filtering example. The filtering operation yields goal-oriented semantic description $\widetilde{Y}$. Dashed red lines denote the *queries/goals* and solid green lines denote the goal-oriented returns. For brevity, we do not visualize class-level graphs S. The goal pattern G^S is searched in S_1 and D_1 to provide $\widetilde{S}_1$, and $\widetilde{D}_1$. Graph complexity is chosen as $l_h = 0$ so that only the exact matchings are returned. The corresponding attributes A_1 are distilled into $\widetilde{A}_1$ according to the given attribute complexity parameter $l_a = 2$.

exist significant overlap between bounding boxes of $person_1$ and $bike_2$ components and have similar velocities, the knowledge base might suggest to utilize $ride_1$ predicate instance to express semantic information $person_1 \rightarrow ride_1 \rightarrow bike_2$. Here, we briefly discuss other alternatives to the designed semantic extractor which can transform the video signals to the semantic graph structure (see Table 4.2).

As discussed in Chapter 2, scene graph generators are able to provide detailed descriptions of scenes based on natural languages which can enrich the predicate set. However, image based scene graph generators lack temporal continuity and requires additional graph level tracker to ensure a temporal extension. On the other hand, video based scene graph generators can process video-stream data as batches in a non-causal fashion without the need of additional graph level tracker. Nevertheless, both image and video based scene graph generators are not suitable for applications that requires real-time processing. For delay tolerant applications, these models can be used directly in place of first three block in Fig. 4.1.

Another alternative semantic extractor can be formed by the combination of captioning and text to graph aligner modules. Specifically, the captioning module outputs a textual description at each frame, and the aligner converts the textual description into a graph format. Note that, unlike scene graph generators, this semantic extractor can be suitable for real-time applications. However, it also lacks temporal continuity and requires a tracker either at the text or graph level. Moreover, inherit utilization of natural languages by the captioning module may introduce ambiguities to the underlying application. It should be pointed out that, the video captioning variants can also be utilized in delay tolerant applications for temporal extension provided by batch processing of frames, again in a non-causal fashion.

Finally, for real-time applications, segmentation methods can be employed to generate an arbitrary semantic description, possibly at a slightly finer resolution then object detection methods with the temporal extension. However, it is more challenging to convert the segmentation mask into the adopted semantic graph structure compared to list of detected objects.

Table 4.2: Comparison of designed semantic extractor and its alternatives.

Semantic Extractor	**Real-Time Processing**	**Temporal Continuity**	**Causal**
Object Detection + Object Tracker (Ours)	Yes (25∼35 FPS)	Yes (by the Object Tracker)	Yes
Scene Graph Generator (Image)	No (0.5∼10 FPS)	No (requires graph level tracker)	Yes
Scene Graph Generator (Video)	No (0.1∼10 FPS)	Yes (processing batch of frames)	No
Image Captioning + Text2Graph Aligner	Yes (20∼30 FPS)	No (requires text or graph level tracker)	Yes
Video Captioning + Text2Graph Aligner	Yes (10∼30 FPS with small batch sizes)	Yes (processing batch of frames)	No

4.2 Other Signal Modalities and Application Areas

So far, we have presented a detailed application example using the proposed semantic signal processing framework: namely, the real-time computer vision on video streams. However, there is a great potential of the proposed framework for use in many other applications as well, including heterogeneous sensor networks that work on a dedicated task. Some application domains and corresponding examples are that can benefit from the proposed semantic signal processing framework listed in Table 4.3. Moreover, we provide the respective input data modalities of these applications in Table 4.4.

A good example of the many fields that can benefit from semantic signal processing is agriculture. For intelligent crop and plantation monitoring applications, a heterogeneous network of sensors (cameras, temperature/humidity sensors, etc.) provides information on critical events such as the crop yield and flowering status, temperature and humidity of the soil, existence of pests [201, 202]. In this application, a global goal regarding the ultimate objectives of the farm can be defined for a common customized language, and then it can be mapped to the respective capabilities of each sensor in the network. Based on goal-oriented returns from the sensors, appropriate actions can be taken. For example, an appropriate pesticide can be recommended and applied automatically to eliminate insects, or in the case of plant growth monitoring, ripe plants can be harvested or infected plants can be exterminated.

Table 4.3: Some examplar smart applications and their domains that can benefit from the proposed semantic signal processing framework.

Domain	**Applications**
Smart City	Road traffic prediction, Electricity consumption prediction, Air quality prediction, Empty parking spot detection, Social network device ownership identification
Smart Building	Plant disease/pest detection, Energy consumption prediction, Human activity/emotion recognition, Human motion tracking, Shop object localization
Healthcare	Smart wearables, Physiological measurements, Pathological voice detection
Industry	Machinery fault diagnosis, 6D pose estimation, Texture and shape detection on production lines
Sports	Real-time sport analytics, Physical activity recognition
Transportation	Mobility prediction, Traffic road classification, Traffic sign detection, Driver Behaviour Detection
Security	Intrusion detection, Crypto currency forecasting, Malware network traffic detection

There are countless many other applications such as elderly fall detection, robot navigation, event detection in sports, animal monitoring, farm automation, traffic condition analysis, etc., that can employ the proposed framework or semantic approaches in general. We strongly believe that future research on signal processing should include a focus on different adaptations of a semantic framework for these applications. In order to give insights on how to adopt the

Table 4.4: Rough categorization of data modalities and their examples.

Data Modality	**Examples**
Time Series Data	Electricity consumption, Natural gas measurements, Humidity, Light intensity, Accelerometer, GPS trajectories, Sound, Bluetooth, Wi-Fi sensor measurements, Air quality, Vibration, Temperature, Seismic measurements
Image Data	IoT camera, Parking lot cameras, Medical images, RGB-D cameras on production lines, Smart fridge/smart markets cameras
Video Data	Human activity annotations, Basketball/soccer game records, Aerial video streams, Security cameras
N-arry Vector Data	Agricultural measurements, generally any sensor measurements that do not require temporal coherency
Graph Data	Telecommunication networks, Transportation/traffic networks, Social connection networks, Connected vehicles (V2V or V2X), Malware/IoT network traffic, Array of things, Crypto currencies, Semantic web for public data streams

proposed semantic signal processing framework for other applications we provide Table 4.5 where input modalities, possible semantic extractor and language component settings are listed.

4.3 Data Compression using Semantics and Goal-Filtering

The proposed semantic signal processing framework represents signals in a very organized and easy-to-parse structure, which enables goal-oriented filtering of the data to achieve very high compression rates. This is especially desirable in the next generation of machine-type communications where a huge amount of raw information will be generated by a plethora of IoT devices that needs to be transmitted throughout massive networks. In this section, we showcase the

Table 4.5: Proposed semantic signal processing framework on other applications.

Application	**Description**	**Input Modality**	**Semantic Extractor**	**Language Components**
Traffic Road Classification	Predict occupancy of roads from traffic states	Graph Data	GSP algorithms, GNNs	$\mathcal{C}$: roads $\mathcal{P}$: road conditions
Real-Time Sport Analytics	Localize players on the court, detect game events	Video Stream Data	CNN+RNN, GNN	$\mathcal{C}$: players, ball $\mathcal{P}$: player interactions, events
Sound Event Detection	Predict occurring objects in the given frame	Time-Series Data	CNN+RNN, Conv-RNN	$\mathcal{C}$: occurring objects $\mathcal{P}$: object states, sound levels
Smart Agriculture	Plant diseases detection and pest control to boost crop yield	Image, N-arry Vector Data	CNN, DNN, DT, SVM	$\mathcal{C}$: crops, plants, bugs, vermins $\mathcal{P}$: crop/plant health states
Driver Behaviour Detection	Identify driver behaviour by front facing vehicle camera	Image, Video Stream Data	CNN, CNN+RNN, Conv-RNN	$\mathcal{C}$: driver, phone, cigarette $\mathcal{P}$: driver states and actions
Voice Pathology Detection	Identify patients' health conditions from their speeches	Time-Series Data	RNN, CNN	$\mathcal{C}$: patients $\mathcal{P}$: patient health conditions

potential compression rates that can be achieved by semantic representation and goal-filtering through a simple example.

Depending on the application and the nature of the signals of interest, the amount of data to be stored or transmitted can be greatly reduced using the proposed semantic signal processing framework. To illustrate this point via a simple example, we have simulated an object detection problem using YOLOv4 [19] on a 240-frame-long prerecorded video. The object set of YOLOv4 is defined as our component set C, while the predicate set is defined only with a p_0 : *exists* predicate as described in Chapter 3. Therefore, in this example, each detection from a frame can be represented as a single component-predicate connection (i.e., $c_i \rightarrow p_0$). The detected class instances throughout the 240-frame video are illustrated in Fig. 4.10.

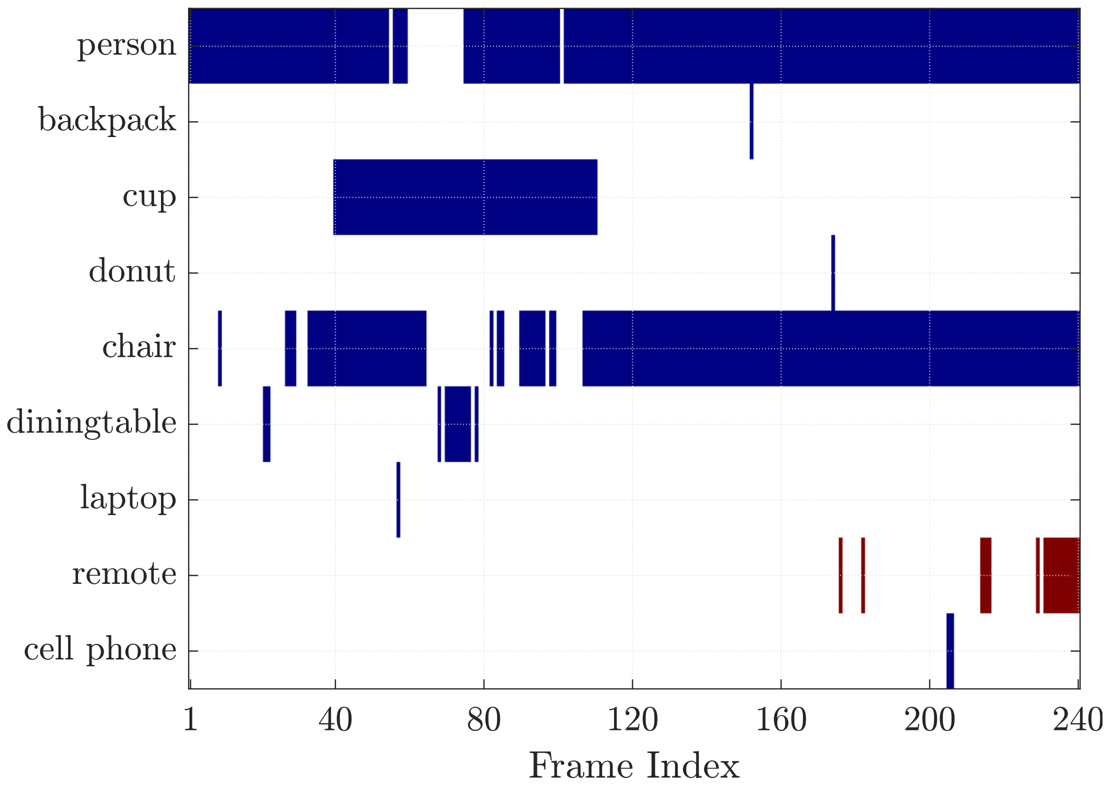


Figure 4.10: Detected component classes using YOLOv4 on a 240-frame long video. The *interesting* detections from class *remote* are shown in red.

In accordance with the semantic language definition in Chapter 3, the detector outputs in Fig. 4.10 correspond to a *multi-graph instance representation* $\mathcal{D}$.

The corresponding attribute set $\mathcal{A}$ for this experiment is chosen as the cropped bounding box images of each detection, with a single level of complexity (i.e., $L = 1$). The class information is then encoded using Huffman coding [203] with a predefined historical occurrence rate. The bounding box images and the full frames are encoded using JPEG compression [204] with a constant compression rate of 10:1. Note that the specific coding schemes discussed here are selected only to give a general idea of the possible data throughputs, and alternative encoding schemes can be used for different applications.

A typical application for this object detection problem could be the transmission of *interesting* objects to an external agent. For the demonstration of the goal-filtering capabilities of the proposed framework, we assume that an external agent is interested in the detections of *remote* class, and may or may not require the bounding box images of pertinent detections. With this configuration of the experiment and the input video stream given in Fig. 4.10, the following transmission strategies are investigated:

- Full-Image Transmission: each full frame is sent,
- $(\mathcal{D}, \mathcal{A})$: the semantic graph outputs and the bounding box images are sent,
- $\mathcal{D}$: only the semantic graph outputs are sent,
- $(\widetilde{\mathcal{D}}, \widetilde{\mathcal{A}})$: the goal-filtered semantic graph outputs and the corresponding bounding box images are sent,
- $\widetilde{\mathcal{D}}$: only the goal-filtered semantic graph outputs are sent.

The data throughput per video frame using the above transmission strategies is given in Fig. 4.11. As seen in Fig. 4.11, the amount of data generated and transmitted can be reduced dramatically by organizing the data in a semantic framework (see $\mathcal{D}$, $\mathcal{A}$). Even further reductions are possible by introducing goals, and filtering out unwanted signal components (see $\widetilde{\mathcal{D}}$, $\widetilde{\mathcal{A}}$). With a goal-oriented approach, transmissions are only sent when events of interest happen. Therefore, the amount of data reduction is dependent on the rate of innovation of *interesting* patterns in the signal. A summary of the results in Fig. 4.11 is given in Table 4.6, where the total throughput of the 240-frame video and the corresponding data compression rates are listed. As reinforced by Table 4.6, extremely high reductions in data rates are possible with the proposed goal-oriented approach.

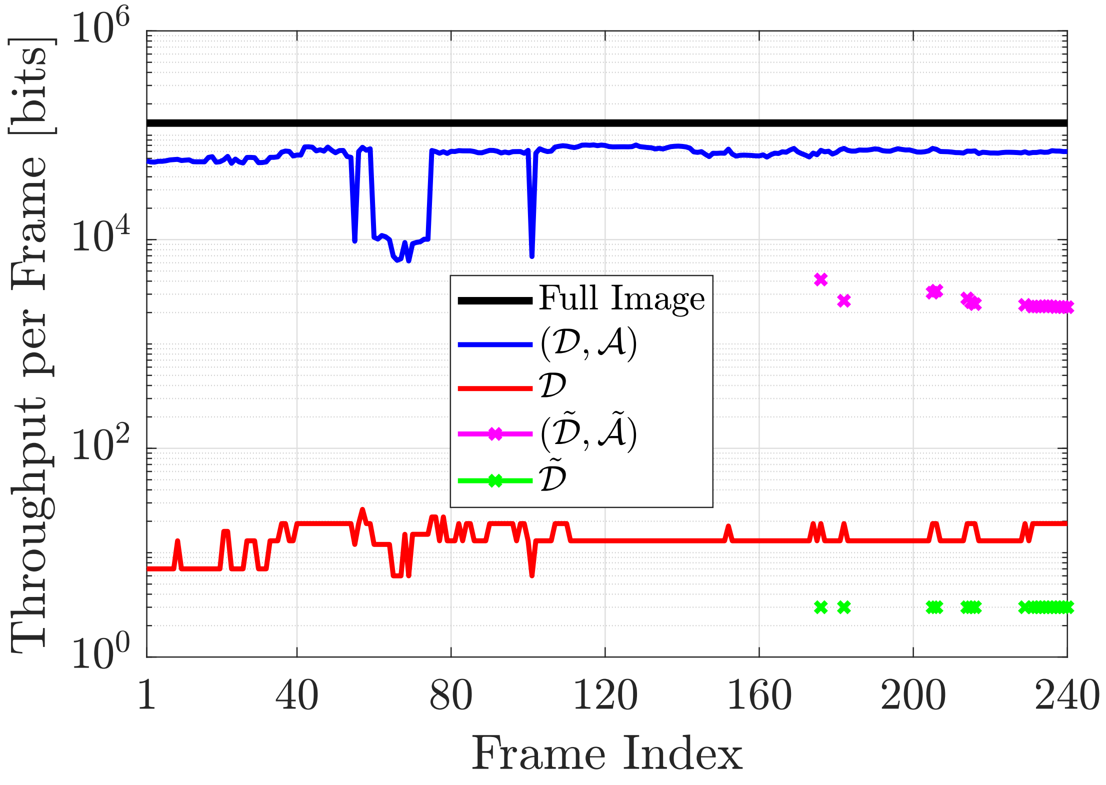


Figure 4.11: Data throughput using different transmission strategies.

Table 4.6: Total throughput and compression rates using the semantic representation of a video stream. Compression rates are given with respect to JPEG frame throughput.

	JPEG Frames	$(\mathcal{D}, \mathcal{A})$	$\mathcal{D}$	$(\tilde{\mathcal{D}}, \tilde{\mathcal{A}})$	$\tilde{\mathcal{D}}$
Total Throughput [bits]	3.1×10^7	1.6×10^7	3316	4.6×10^4	54
Compression Rate		2 : 1	9468 : 1	684 : 1	580000 : 1

In many semantic signal processing applications, it is possible to further increase the compression rates. As an example, we consider the case study described in Section 4.1 where it is possible to track the individual objects by comparing their features as illustrated in Fig. 4.6c using the similarity of the feature vectors for consecutive frames. These similarities can also be exploited to create a differential data storage and transmit system. That is, the first feature vector obtained for a specific individual is to be compressed as previously described. When a new feature vector arrives, the difference with the previous one can be computed, which represents the fresh information in the new feature vector.

Furthermore, one may expect to have nearly *sparse* difference vectors due to similarities, and consequently, it may be possible to further compress the information to be stored or transmitted via innovative compressed sensing approaches [205].

In addition, the averages of feature vectors for each class can be determined and shared with the receiver offline to enable *class-aware compression.* In most scenarios, the attribute set (3.16) contains the class information and its associated features. Hence, instead of compressing and transmitting each feature vector, sensors may use the average feature information which is available at both the sensors and the base station. Similar to a differential data transmission setup, the fresh information in the feature vector can be extracted by taking the difference between the new feature vector and the average feature vector of the corresponding class. Furthermore, vector quantization or compressed sensing can be employed to compress the differential feature vector. An interesting line of research in the context of *class-aware compression* is not only to use the class feature averages but also design a compressor for each class separately to exploit the shared knowledge of the sensors and the base station.

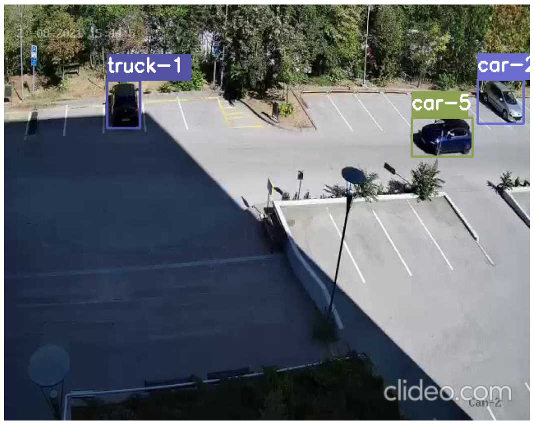
4.4 Innovation Detection on Attributes

In this section, we present a way to identify points of significant innovation over extended periods of time using the companion attributes embedded in semantic graph output. An example of an innovation event at the graph signal level can mean a graph pattern of interest (that is predefined by the internally or externally defined goals) has emerged (or receded) in the semantic description of the raw signal. Once a graph pattern of interest is identified and is being tracked, the numerical attributes such as position and subfeature vectors can be tracked across time to detect innovation events at the attribute level (again, based on the interests defined by the goals). As stated in Section 4.1.3, we employ state vector $\boldsymbol{m}_i(t)$, visual feature vector $\boldsymbol{r}_i(t)$ and cropped RGB image $I_i(t)$ as the multi-level attributes for each signal component node for our computer vision application. Here, we utilize exponential moving average (EMA) formulated in (4.17) with $\beta = 0.9$ and track the evolution of these features while reducing the number of stored feature vectors and speeding up object tracking submodule which explained in Section 4.1.2. To identify the innovation, we increment t and estimate the contribution of recently acquired attribute vector. Specifically, we choose ℓ_1-norm of the contribution and compare it with a predefined threshold.

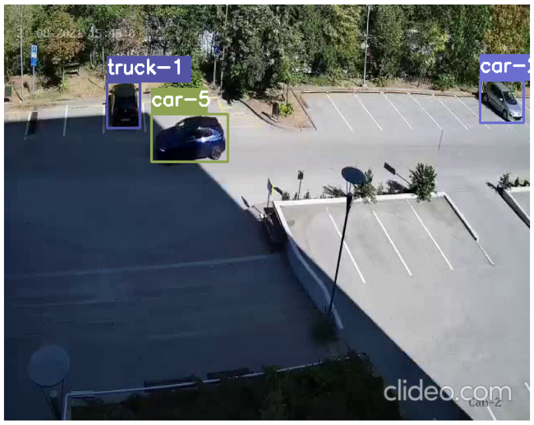
To illustrate the observed innovation across consecutive frames, we show the case in Fig. 4.12 where a car transients from a sunny to shaded region, abruptly changes its trajectory and lefts the scene backward. Note that in the scenario given in Fig. 4.12, the semantic components and predicates stay the same (i.e., the same objects and relationships are present throughout the clip), while the attributes of the *car-5* component change due to its movement, image brightness and contrast etc.

First, we utilize second-level attributes $\boldsymbol{r}(t)$ and its EMA evolved counterpart $\tilde{\boldsymbol{r}}(t)$, illustrated in Fig. 4.13, to identify the innovation occurred when the lightning conditions change for *car-5* component during transition from the sunny to the shaded region of parking lot (see Fig. 4.12a and Fig. 4.12b). We define the innovation as ℓ_1-norm of contribution vector $\boldsymbol{r}_c(t) = \boldsymbol{r}(t) - \tilde{\boldsymbol{r}}(t)$. As illustrated in Fig. 4.14, $||\boldsymbol{r}_c(t)||_1$ achieves its maximum around $t = 105$ where we identify the innovation due to transition of *car-5* component from the sunny to shaded region.

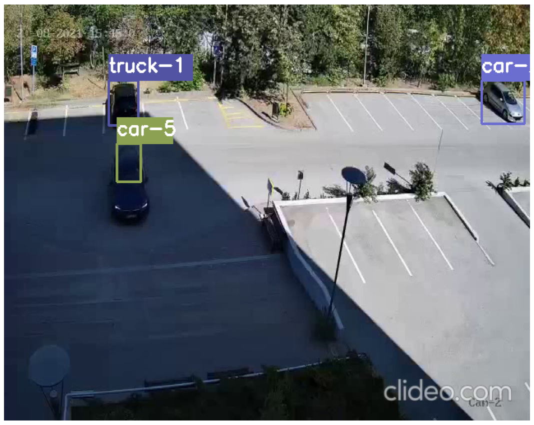
Next, we utilize a fragment of the first-level attributes, i.e., the vector corresponding to velocity entries $[\dot{x}^c, \dot{y}^c, \dot{w}, \dot{h}]$ in $\boldsymbol{m}(t)$, denoted by $\boldsymbol{m}^v(t)$ and its EMA evolved counterpart $\tilde{\boldsymbol{m}}^v(t)$, illustrated in Fig. 4.15, to identify the innovation occurred when *car-5* component abruptly changes its trajectory to left the scene backward (see Fig. 4.12c to Fig. 4.12f). Again, we define the innovation as ℓ_1-norm of contribution vector $\boldsymbol{m}_c^v(t) = \boldsymbol{m}^v(t) - \tilde{\boldsymbol{n}}^v(t)$. As illustrated in Fig. 4.16, $||\boldsymbol{m}_c^v(t)||_1$ achieves its maximum around $t = 242$ where we identify the innovation due to changes in *car-5* component's trajectory due to abrupt stopping followed by backward movement.



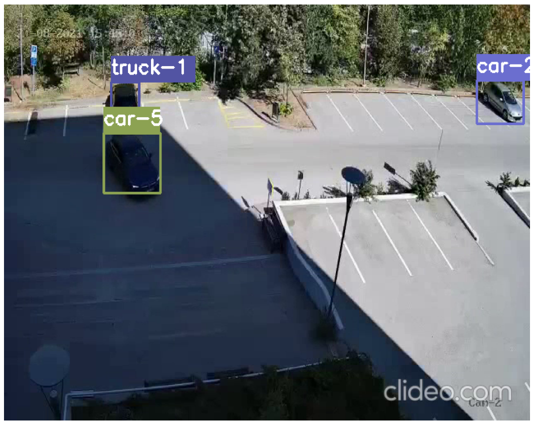
(a) Frame 16



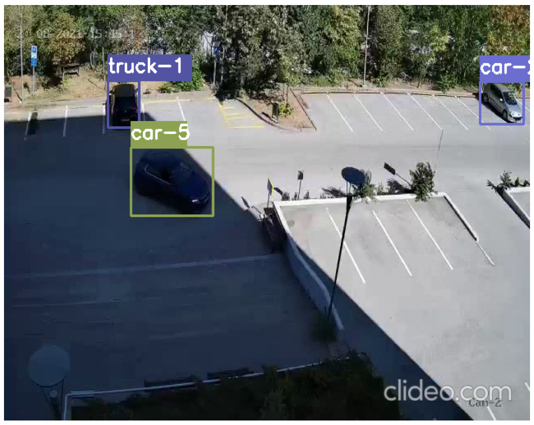
(b) Frame 100



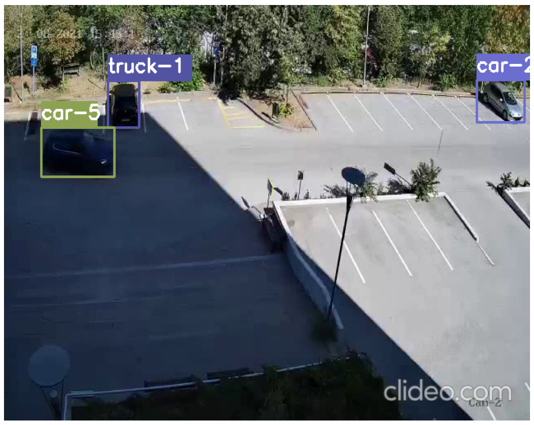
(c) Frame 240



(d) Frame 303

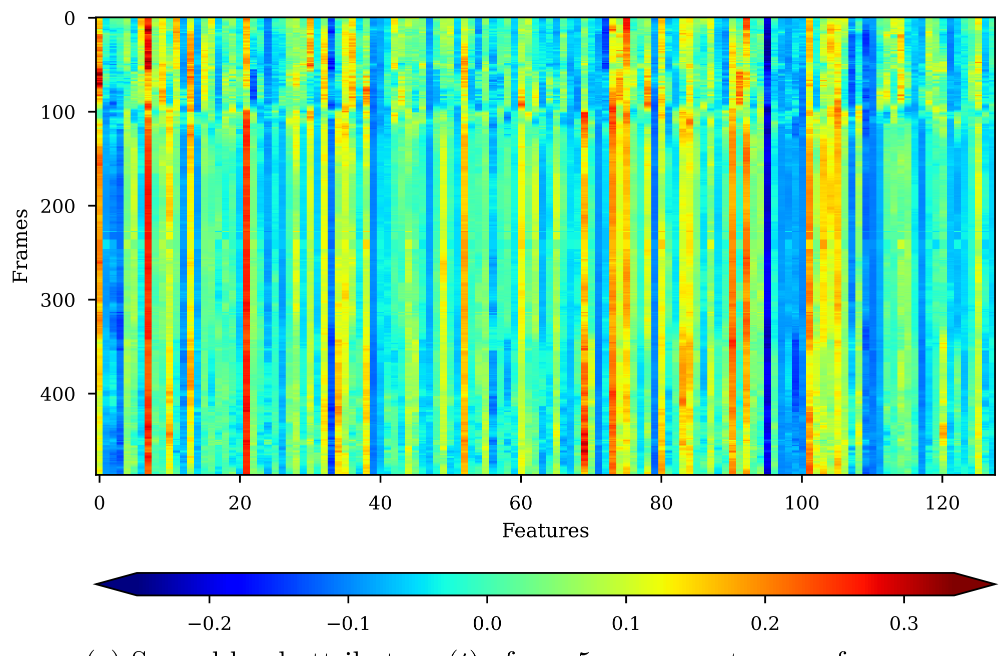


(e) Frame 378

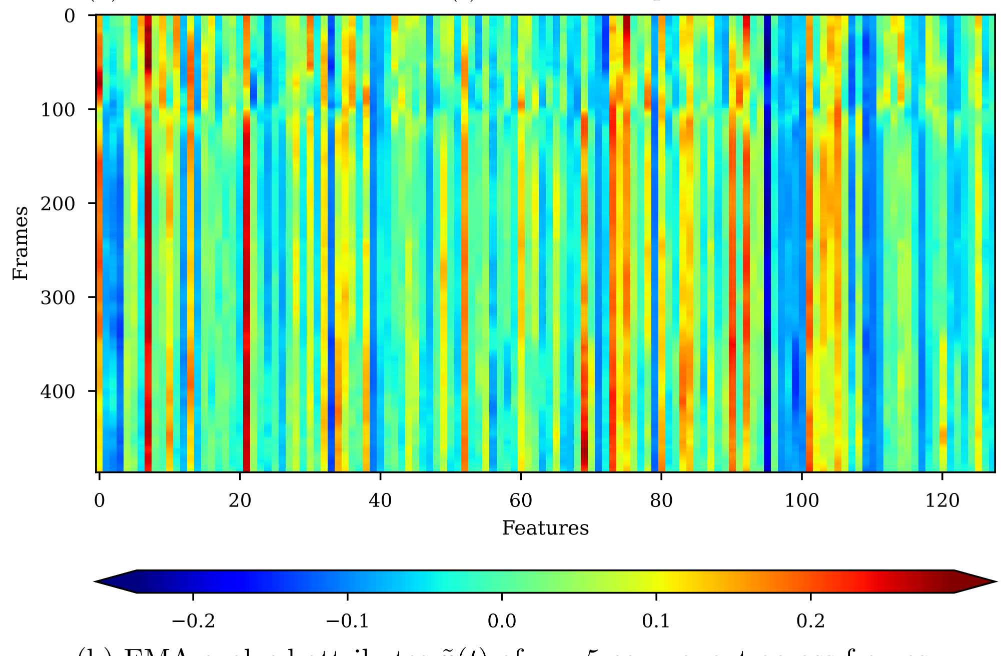


(f) Frame 453

Figure 4.12: A car transients from a sunny to shaded region, abruptly changes its trajectory and lefts the scene backward.

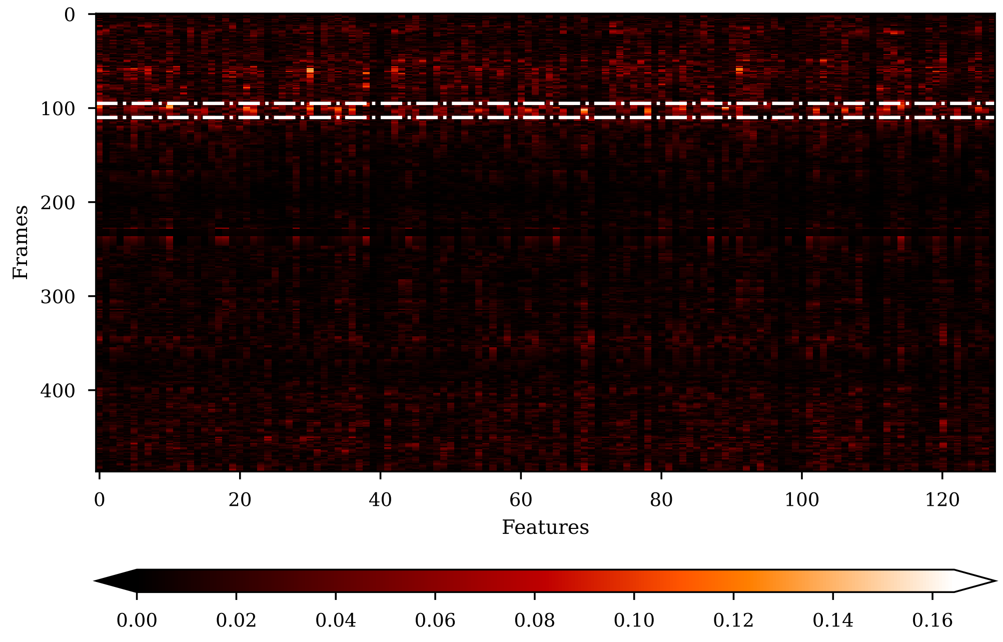


(a) Second-level attributes $\boldsymbol{r}(t)$ of *car-5* component across frames.

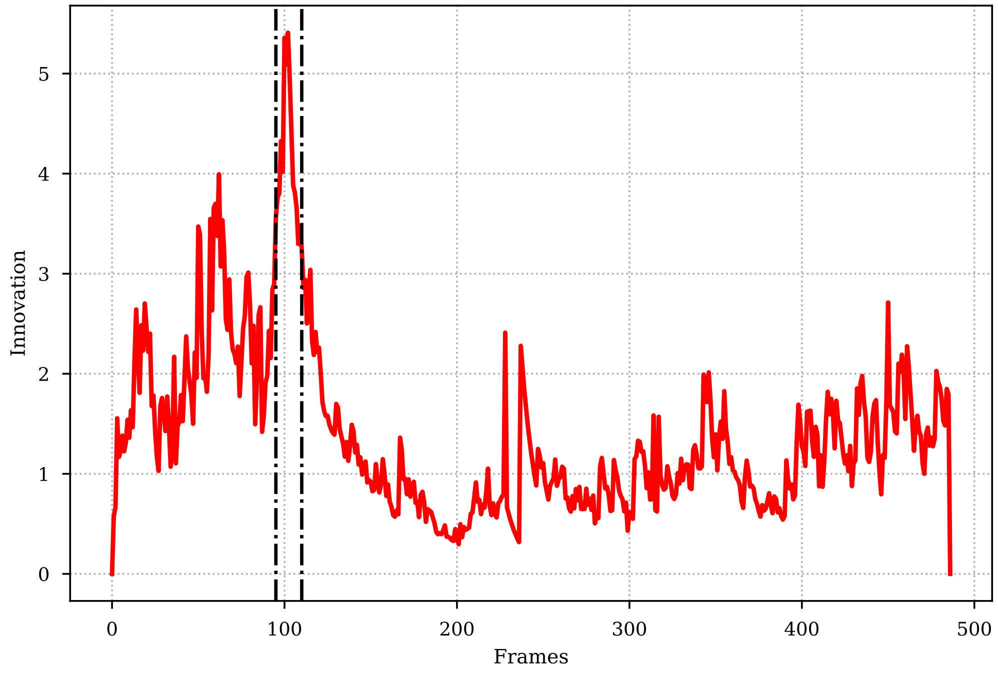


(b) EMA evolved attributes $\tilde{\boldsymbol{r}}(t)$ of *car-5* component across frames.

Figure 4.13: Original and EMA evolved second-level attributes of *car-5* component across frames.

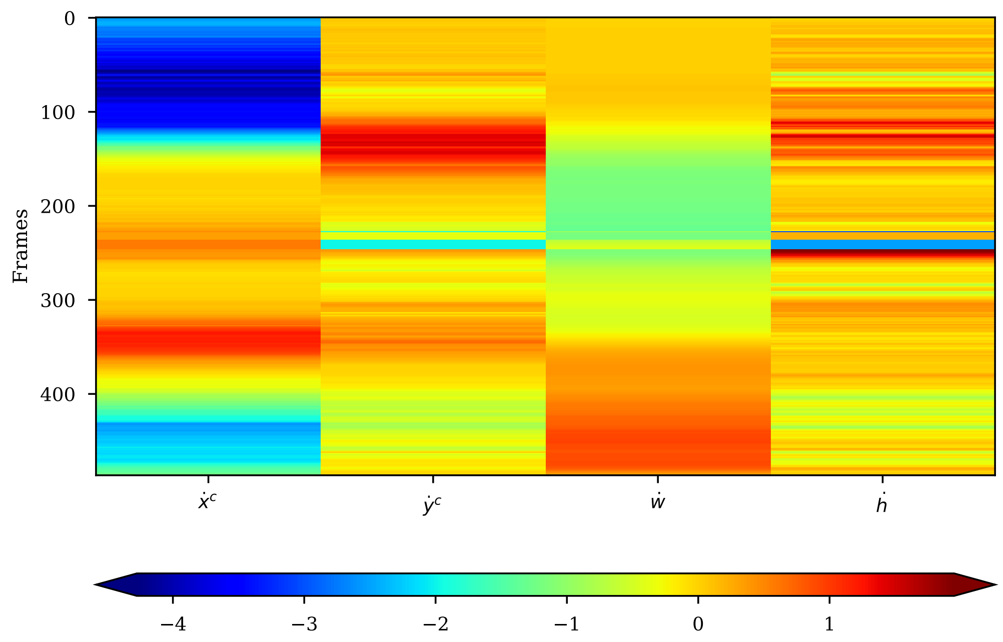


(a) Absolute value of contribution vector $\boldsymbol{r}_c(t) = \boldsymbol{r}(t) - \tilde{\boldsymbol{r}}(t)$ for *car-5* component across frames. The innovation region is denoted with white dashdotted lines.

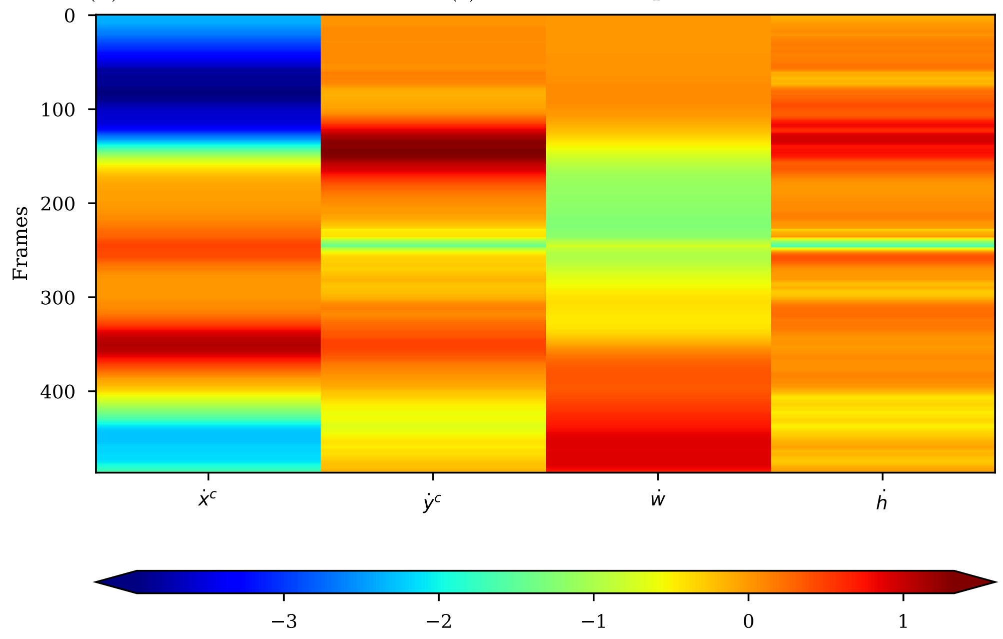


(b) Innovation of second-level attributes $\boldsymbol{r}(t)$ across frames as ℓ_1-norm of contribution vector $\boldsymbol{r}_c(t)$. The innovation region is denoted with black dashdotted lines.

Figure 4.14: Innovation analysis of second-level attributes for *car-5* component. The innovation curve achieves its peak value around frame 105 where *car-5* component transients from sunny to shaded region as illustrated in Fig. 4.12b.

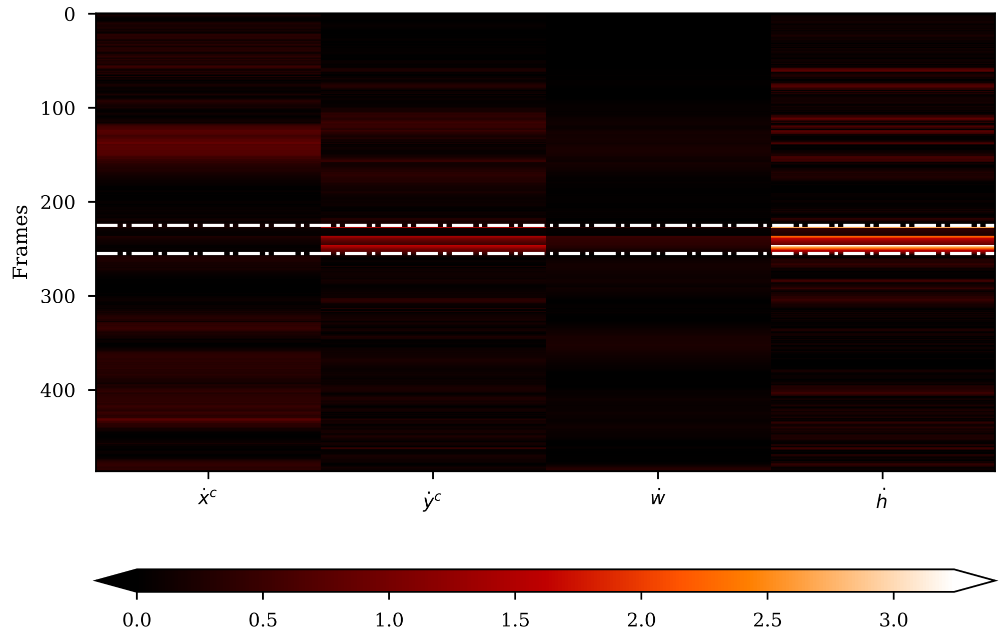


(a) First-level attributes $\boldsymbol{m}^v(t)$ of *car-5* component across frames.

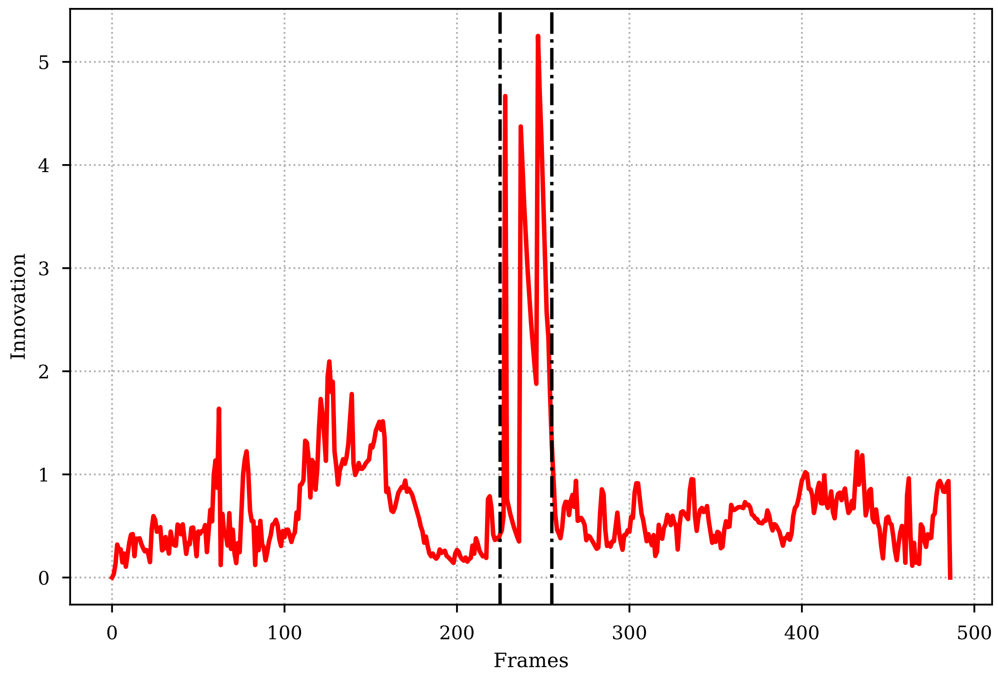


(b) EMA evolved attributes $\tilde{\boldsymbol{m}}^v(t)$ of *car-5* component across frames.

Figure 4.15: Original and EMA first-level attributes evolved attributes of *car-5* component across frames.



(a) Absolute value of contribution vector $\boldsymbol{m}_c^v(t) = \boldsymbol{m}^v(t) - \tilde{\boldsymbol{m}}^v(t)$ for *car-5* component across frames. The innovation region is denoted with white dashdotted lines.



(b) Innovation of first-level attributes $\boldsymbol{m}^v(t)$ across frames as ℓ_1-norm of contribution vector $\boldsymbol{m}_c^v(t)$. The innovation region is denoted with black dashdotted lines.

Figure 4.16: Innovation analysis of first-level attributes for *car-5* component. The innovation curve achieves its peak value around frame 242 where *car-5* component changes its trajectory with abrupt stopping followed by backward movement as illustrated in Fig. 4.12c.

Chapter 5

Conclusion and Future Work

In this thesis, we first reviewed definitions of semantic information, various semantic extraction and semantic communication techniques in the literature. Then, we re-introduced the formal semantic signal processing framework proposed in our former work [25] where the semantic information in the signals represented by a graph-based structure. The adopted semantic graph-language enables structured and universal representation and efficient processing of semantic information in the signals. In the framework, preprocessing of input signals is followed by a semantic information extractor block which identifies components from a set of predefined application specific classes where the states, actions and relations among the identified components are described by another application specific predefined set called predicates. Moreover, to provide additional information regarding the input signal in an organized way, each node in the resulting semantic graph is embedded with a hierarchical set of attributes. The adopted signal processing framework also incorporates internally or externally defined time varying goals that enables grouping of graphs that will proceed in the process chain and that are not essential to the underlying task. The post-processing block may contain wide range of signal processing techniques on filtered semantic graph outputs such as spatio-temporal graph tracking and innovation identification. At any point in the processing chain, the desired level of semantic information of those graphs which are of interest can be locally stored or shared with another processor through a communication protocol.

Here, we focused on semantic extractor block of the adopted semantic signal processing framework. To demonstrate the potential of the re-introduced framework, we presented a real-time computer vision application on video-stream data and designed an appropriate semantic extractor. Specifically, we adopted tracking by detection paradigm to identify semantic signal components in the stream and exploited object positions and velocities to identify predicates. Furthermore, we employed position and velocity indicating state vector, visual feature vector and cropped RGB image as the multi-level attributes. Then, we showed that, with the proposed semantic representation and goal-filtering, it is possible to achieve extremely high reductions in data rates compared to traditional approaches. Finally, we demonstrated a way to identify points of significant innovation over extended periods of time by tracking the evolution of multi-level attributes while reducing the number of stored features.

The adopted semantic signal processing framework opens up multiple research directions, in both theory and practice. First, available machine learning techniques should be assessed for their applicability in this framework for real-time, offline, and batch processing applications. For real-time applications where sensor data is semantically processed to observe and control the state of a system, semantic extraction should be completed within a certain deadline, depending on signal bandwidth and processing capabilities. For applications that may allow offline processing such as semantic medical imaging, the semantic extraction and processing techniques must prioritize increasing the reliability of extracted meaning over the computational complexity. Based on the results of the assessment on both existing real-time and offline semantic extractors, desired features of new machine learning techniques should be identified for improved semantic extraction in the proposed framework.

Along with the new ML techniques for semantic information extraction, new goal-oriented signal processing techniques should be developed to take advantage of the available semantic information on the identified signal components. By first identifying the components of interest and prioritizing processing their attributes, the re-introduced framework may improve the time complexity over traditional signal processing techniques. Note that, these algorithms should be able to set the goals of semantic filtering and semantic post-processing based on the estimated states of the components of interest. Further research on goal generation and dissemination of goals should also be performed.

It is also important to note that, the process of semantic information extraction relies on AI-enabled techniques followed by traditional methods as post-processes to meet real-time requirements. However, the development of a single unified AI-enabled model for the transformation of video stream into the proposed graph structure is another research direction we plan to investigate. Furthermore, in this thesis, first, we established a connection between innovation and component node attributes through observations, then, presented a way to characterize the innovation with traditional ways. Different characterizations of innovation may pave the way for new AI-enabled methods for innovation detection and tracking. A promising approach is to treat innovations as anomalies and develop AI-enabled anomaly detection models [206] for attribute-level innovation tracking with a unified architecture. Moreover, the same reasoning can be also applied to graph-level innovations and graph anomaly detection techniques [207] may be preferred. As future work, we plan to delve further into characterization of the innovation problem at both attribute and graph-level.

We finally note that, depending on the type of sensor/device and its computational capabilities, the adopted framework can be used collectively or independently. As the signal processing and communications paradigms move towards semantic signal processing and transmission, we believe the proposed semantic extraction framework will be an essential building block in developing the next generation of sensor devices and networks.

Bibliography

[1] M. Kaveh, "Signal processing everywhere [president's message]," *IEEE Signal Processing Magazine*, vol. 28, no. 2, p. 6, 2011.

[2] Y. Bar-Hillel and R. Carnap, "Semantic information," *The British Journal for the Philosophy of Science*, vol. 4, no. 14, pp. 147–157, 1953.

[3] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[4] D. Smith and S. Singh, "Approaches to multisensor data fusion in target tracking: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 12, pp. 1696–1710, 2006.

[5] B. Khaleghi, A. Khamis, F. O. Karray, and S. N. Razavi, "Multisensor data fusion: A review of the state-of-the-art," *Information Fusion*, vol. 14, no. 1, pp. 28–44, 2013.

[6] E. I. Zacharaki, S. Wang, S. Chawla, D. Soo Yoo, R. Wolf, E. R. Melhem, and C. Davatzikos, "Classification of brain tumor type and grade using MRI texture and shape in a machine learning scheme," *Magnetic Resonance in Medicine*, vol. 62, no. 6, pp. 1609–1618, 2009.

[7] J. Wiens and E. S. Shenoy, "Machine learning for healthcare: On the verge of a major shift in healthcare epidemiology," *Clinical Infectious Diseases*, vol. 66, pp. 149–153, 08 2017.

[8] B. J. Erickson, P. Korfiatis, Z. Akkus, and T. L. Kline, "Machine learning for medical imaging," *RadioGraphics*, vol. 37, no. 2, pp. 505–515, 2017.

[9] B. Krollner, B. Vanstone, and G. Finnie, “Financial time series forecasting with machine learning techniques: A survey,” in *Proceedings of the 18th European Symposium on Artificial Neural Networks (ESANN 2010)*, pp. 25–30, 2010.

[10] A. Dingli and K. S. Fournier, “Financial time series forecasting – a deep learning approach,” *International Journal of Machine Learning and Computing*, vol. 7, pp. 118–122, 2017.

[11] M. F. Dixon, I. Halperin, and P. Bilokon, *Machine Learning in Finance: From Theory to Practice*. Springer, Cham, 1 ed., 2020.

[12] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan, “Large-scale parallel collaborative filtering for the Netflix prize,” in *Algorithmic Aspects in Information and Management*, (Berlin, Heidelberg), pp. 337–348, Springer Berlin Heidelberg, 2008.

[13] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, *et al.*, “Grandmaster level in StarCraft II using multi-agent reinforcement learning,” *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[14] N. Farsad, M. Rao, and A. Goldsmith, “Deep learning for joint source-channel coding of text,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2326–2330, 2018.

[15] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, “In-edge AI: intelligentizing mobile edge computing, caching and communication by federated learning,” *IEEE Network*, vol. 33, no. 5, pp. 156–165, 2019.

[16] J. Metcalf, S. D. Blunt, and B. Himed, “A machine learning approach to cognitive radar detection,” in *2015 IEEE Radar Conference (RadarCon)*, pp. 1405–1411, 2015.

[17] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *Advances in neural information processing systems*, vol. 28, 2015.

[18] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10781–10790, 2020.

[19] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Scaled-yolov4: Scaling cross stage partial network," in *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, pp. 13029–13038, 2021.

[20] K. B. Letaief, W. Chen, Y. Shi, J. Zhang, and Y.-J. A. Zhang, "The roadmap to 6G: AI empowered wireless networks," *IEEE Communications Magazine*, vol. 57, no. 8, pp. 84–90, 2019.

[21] E. C. Strinati and S. Barbarossa, "6G networks: Beyond Shannon towards semantic and goal-oriented communications," *Computer Networks*, vol. 190, p. 107930, 2021.

[22] I. F. Akyildiz, A. Kak, and S. Nie, "6G and beyond: The future of wireless communications systems," *IEEE Access*, vol. 8, pp. 133995–134030, 2020.

[23] S. Dang, O. Amin, B. Shihada, and M.-S. Alouini, "What should 6G be?," *Nature Electronics*, vol. 3, no. 1, pp. 20–29, 2020.

[24] Y. Chen, P. Zhu, G. He, X. Yan, H. Baligh, and J. Wu, "From connected people, connected things, to connected intelligence," in *2020 2nd 6G wireless summit (6G SUMMIT)*, pp. 1–7, IEEE, 2020.

[25] M. Kalfa, M. Gok, A. Atalik, B. Tegin, T. M. Duman, and O. Arikan, "Towards goal-oriented semantic signal processing: Applications and future challenges," *Digital Signal Processing*, vol. 119, p. 103134, 2021.

[26] C. E. Shannon and W. Weaver, "The mathematical theory of information," *Urbana: University of Illinois Press*, vol. 97, no. 6, pp. 128–164, 1949.

[27] B. Güler, A. Yener, and A. Swami, "The semantic communication game," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 4, pp. 787–802, 2018.

[28] H. Xie, Z. Qin, G. Y. Li, and B.-H. Juang, “Deep learning enabled semantic communication systems,” *IEEE Transactions on Signal Processing*, vol. 69, pp. 2663–2675, 2021.

[29] H. Xie and Z. Qin, “A lite distributed semantic communication system for internet of things,” *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 142–153, 2020.

[30] E. Bourtsoulatze, D. B. Kurka, and D. Gündüz, “Deep joint source-channel coding for wireless image transmission,” *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 3, pp. 567–579, 2019.

[31] J. Bao, P. Basu, M. Dean, C. Partridge, A. Swami, W. Leland, and J. A. Hendler, “Towards a theory of semantic communication,” in *2011 IEEE Network Science Workshop*, pp. 110–117, IEEE, 2011.

[32] X. Liu, W. Jia, W. Liu, and W. Pedrycz, “Afsse: An interpretable classifier with axiomatic fuzzy set and semantic entropy,” *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 11, pp. 2825–2840, 2019.

[33] Z. Qin, X. Tao, J. Lu, and G. Y. Li, “Semantic communications: Principles and challenges,” *arXiv preprint arXiv:2201.01389*, 2021.

[34] X. Chang, P. Ren, P. Xu, Z. Li, X. Chen, and A. Hauptmann, “Scene graphs: A survey of generations and applications,” *arXiv preprint arXiv:2104.01111*, 2021.

[35] Y. Li, W. Ouyang, B. Zhou, K. Wang, and X. Wang, “Scene graph generation from objects, phrases and region captions,” in *Proceedings of the IEEE international conference on computer vision*, pp. 1261–1270, 2017.

[36] D. Xu, Y. Zhu, C. B. Choy, and L. Fei-Fei, “Scene graph generation by iterative message passing,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5410–5419, 2017.

[37] Y. Li, W. Ouyang, X. Wang, and X. Tang, “Vip-cnn: Visual phrase guided convolutional neural network,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1347–1356, 2017.

[38] Y. Li, W. Ouyang, B. Zhou, J. Shi, C. Zhang, and X. Wang, "Factorizable net: an efficient subgraph-based framework for scene graph generation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 335–351, 2018.

[39] J. Yang, J. Lu, S. Lee, D. Batra, and D. Parikh, "Graph r-cnn for scene graph generation," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 670–685, 2018.

[40] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[41] K. Tang, H. Zhang, B. Wu, W. Luo, and W. Liu, "Learning to compose dynamic tree structures for visual contexts," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6619–6628, 2019.

[42] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, pp. 1112–1119, 2014.

[43] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 12, pp. 2724–2743, 2017.

[44] L. Zou, R. Huang, H. Wang, J. X. Yu, W. He, and D. Zhao, "Natural language question answering over RDF: a graph data driven approach," in *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, pp. 313–324, 2014.

[45] W. Zheng, H. Cheng, J. X. Yu, L. Zou, and K. Zhao, "Interactive natural language question answering over knowledge graphs," *Information Sciences*, vol. 481, pp. 141–159, 2019.

[46] J. J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne, and K. Wilkinson, "Jena: implementing the semantic web recommendations," in *Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers & Posters*, pp. 74–83, 2004.

[47] U. Bellur and R. Kulkarni, "Improved matchmaking algorithm for semantic web services based on bipartite graph matching," in *IEEE International Conference on Web Services (ICWS 2007)*, pp. 86–93, IEEE, 2007.

[48] T. Segaran, C. Evans, and J. Taylor, *Programming the Semantic Web: Build Flexible Applications with Graph Data.* " O'Reilly Media, Inc.", 2009.

[49] W. Fan, J. Li, S. Ma, N. Tang, Y. Wu, and Y. Wu, "Graph pattern matching: From intractable to polynomial time," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 264–275, 2010.

[50] W. Fan, X. Wang, and Y. Wu, "Incremental graph pattern matching," *ACM Transactions on Database Systems (TODS)*, vol. 38, no. 3, pp. 1–47, 2013.

[51] F. Serratosa, "Fast computation of bipartite graph matching," *Pattern Recognition Letters*, vol. 45, pp. 244–250, 2014.

[52] J. Sun, H. Qu, D. Chakrabarti, and C. Faloutsos, "Neighborhood formation and anomaly detection in bipartite graphs," in *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pp. 8–pp, IEEE, 2005.

[53] Š. Čebirić, F. Goasdoué, H. Kondylakis, D. Kotzinos, I. Manolescu, G. Troullinou, and M. Zneika, "Summarizing semantic graphs: a survey," *The VLDB Journal*, vol. 28, no. 3, pp. 295–327, 2019.

[54] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[55] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation [c] 2014 ieee conference on computer vision and pattern recognition (cvpr)," *IEEE Computer Society Go to reference in article*, 2014.

[56] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

[57] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[58] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, 2017.

[59] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[60] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[61] W. Van Gansbeke, S. Vandenhende, S. Georgoulis, and L. Van Gool, "Unsupervised semantic segmentation by contrasting object mask proposals," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10052–10062, 2021.

[62] X. Xia and B. Kulis, "W-net: A deep model for fully unsupervised image segmentation," *arXiv preprint arXiv:1711.08506*, 2017.

[63] X. Ji, J. F. Henriques, and A. Vedaldi, "Invariant information distillation for unsupervised image segmentation and clustering," *arXiv preprint arXiv:1807.06653*, vol. 2, no. 3, p. 8, 2018.

[64] J. Shotton, M. Johnson, and R. Cipolla, "Semantic texton forests for image categorization and segmentation," in *2008 IEEE conference on computer vision and pattern recognition*, pp. 1–8, IEEE, 2008.

[65] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *CVPR 2011*, pp. 1297–1304, Ieee, 2011.

[66] J. Tighe, M. Niethammer, and S. Lazebnik, "Scene parsing with object instances and occlusion ordering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3748–3755, 2014.

[67] S. Hao, Y. Zhou, and Y. Guo, "A brief survey on semantic segmentation with deep learning," *Neurocomputing*, vol. 406, pp. 302–321, 2020.

[68] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[69] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015.

[70] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.

[71] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proceedings of the IEEE international conference on computer vision*, pp. 1520–1528, 2015.

[72] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[73] R. P. Poudel, P. Lamata, and G. Montana, "Recurrent fully convolutional neural networks for multi-slice mri cardiac segmentation," in *Reconstruction, segmentation, and analysis of medical images*, pp. 83–94, Springer, 2016.

[74] P. Luc, C. Couprie, S. Chintala, and J. Verbeek, "Semantic segmentation using adversarial networks," *arXiv preprint arXiv:1611.08408*, 2016.

[75] Y. Xue, T. Xu, H. Zhang, L. R. Long, and X. Huang, "Segan: adversarial network with multi-scale l1 loss for medical image segmentation," *Neuroinformatics*, vol. 16, no. 3, pp. 383–392, 2018.

[76] Y. Luo, Z. Zheng, L. Zheng, T. Guan, J. Yu, and Y. Yang, "Macro-micro adversarial network for human parsing," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 418–434, 2018.

[77] R. P. Poudel, S. Liwicki, and R. Cipolla, "Fast-scnn: Fast semantic segmentation network," *arXiv preprint arXiv:1902.04502*, 2019.

[78] H. Park, L. L. Sjösund, Y. Yoo, J. Bang, and N. Kwak, "Extremec3net: Extreme lightweight portrait segmentation networks using advanced c3-modules," *arXiv preprint arXiv:1908.03093*, 2019.

[79] H. Li, P. Xiong, H. Fan, and J. Sun, "Dfanet: Deep feature aggregation for real-time semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9522–9531, 2019.

[80] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár, "Panoptic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9404–9413, 2019.

[81] F. Lateef and Y. Ruichek, "Survey on semantic segmentation using deep learning techniques," *Neurocomputing*, vol. 338, pp. 321–348, 2019.

[82] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, P. Martinez-Gonzalez, and J. Garcia-Rodriguez, "A survey on deep learning techniques for image and video semantic segmentation," *Applied Soft Computing*, vol. 70, pp. 41–65, 2018.

[83] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2625–2634, 2015.

[84] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3128–3137, 2015.

[85] J. Mao, W. Xu, Y. Yang, J. Wang, and A. L. Yuille, "Explain images with multimodal recurrent neural networks," *arXiv preprint arXiv:1410.1090*, 2014.

[86] X. Chen and C. Lawrence Zitnick, "Mind's eye: A recurrent visual representation for image caption generation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2422–2431, 2015.

[87] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International conference on machine learning*, pp. 2048–2057, PMLR, 2015.

[88] J. Johnson, A. Karpathy, and L. Fei-Fei, "Densecap: Fully convolutional localization networks for dense captioning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4565–4574, 2016.

[89] L. Yang, K. Tang, J. Yang, and L.-J. Li, "Dense captioning with joint inference and visual context," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2193–2202, 2017.

[90] D.-J. Kim, T.-H. Oh, J. Choi, and I. S. Kweon, "Dense relational image captioning via multi-task triple-stream networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[91] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, *et al.*, "Visual genome: Connecting language and vision using crowdsourced dense image annotations," *International journal of computer vision*, vol. 123, no. 1, pp. 32–73, 2017.

[92] N. Aafaq, A. Mian, W. Liu, S. Z. Gilani, and M. Shah, "Video description: A survey of methods, datasets, and evaluation metrics," *ACM Computing Surveys (CSUR)*, vol. 52, no. 6, pp. 1–37, 2019.

[93] S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko, "Translating videos to natural language using deep recurrent neural networks," *arXiv preprint arXiv:1412.4729*, 2014.

[94] H. Xu, S. Venugopalan, V. Ramanishka, M. Rohrbach, and K. Saenko, "A multi-scale multiple instance video description network," *arXiv preprint arXiv:1505.05914*, 2015.

[95] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko, "Sequence to sequence-video to text," in *Proceedings of the IEEE international conference on computer vision*, pp. 4534–4542, 2015.

[96] H. Yu, J. Wang, Z. Huang, Y. Yang, and W. Xu, "Video paragraph captioning using hierarchical recurrent neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4584–4593, 2016.

[97] J. Krause, J. Johnson, R. Krishna, and L. Fei-Fei, "A hierarchical approach for generating descriptive image paragraphs," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 317–325, 2017.

[98] R. Krishna, K. Hata, F. Ren, L. Fei-Fei, and J. Carlos Niebles, "Dense-captioning events in videos," in *Proceedings of the IEEE international conference on computer vision*, pp. 706–715, 2017.

[99] H. Xu, B. Li, V. Ramanishka, L. Sigal, and K. Saenko, "Joint event detection and description in continuous video streams," in *2019 IEEE winter conference on applications of computer vision (WACV)*, pp. 396–405, IEEE, 2019.

[100] M. Z. Hossain, F. Sohel, M. F. Shiratuddin, and H. Laga, "A comprehensive survey of deep learning for image captioning," *ACM Computing Surveys (CsUR)*, vol. 51, no. 6, pp. 1–36, 2019.

[101] J. Johnson, R. Krishna, M. Stark, L.-J. Li, D. Shamma, M. Bernstein, and L. Fei-Fei, "Image retrieval using scene graphs," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3668–3678, 2015.

[102] R. Zellers, M. Yatskar, S. Thomson, and Y. Choi, "Neural motifs: Scene graph parsing with global context," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5831–5840, 2018.

[103] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3, pp. 229–256, 1992.

[104] C. Lu, R. Krishna, M. Bernstein, and L. Fei-Fei, "Visual relationship detection with language priors," in *European conference on computer vision*, pp. 852–869, Springer, 2016.

[105] R. Yu, A. Li, V. I. Morariu, and L. S. Davis, "Visual relationship detection with internal and external linguistic knowledge distillation," in *Proceedings of the IEEE international conference on computer vision*, pp. 1974–1982, 2017.

[106] A. Zareian, S. Karaman, and S.-F. Chang, "Bridging knowledge graphs to generate scene graphs," in *European Conference on Computer Vision*, pp. 606–623, Springer, 2020.

[107] R. Wang, Z. Wei, P. Li, Q. Zhang, and X. Huang, "Storytelling from an image stream using scene graphs," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 9185–9192, 2020.

[108] H. Qi, Y. Xu, T. Yuan, T. Wu, and S.-C. Zhu, "Scene-centric joint parsing of cross-view videos," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[109] A. Agarwal, A. Mangal, *et al.*, "Visual relationship detection using scene graphs: A survey," *arXiv preprint arXiv:2005.08045*, 2020.

[110] M. Malik, M. K. Malik, K. Mehmood, and I. Makhdoom, "Automatic speech recognition: a survey," *Multimedia Tools and Applications*, vol. 80, no. 6, pp. 9411–9457, 2021.

[111] K. H. Davis, R. Biddulph, and S. Balashek, "Automatic recognition of spoken digits," *The Journal of the Acoustical Society of America*, vol. 24, no. 6, pp. 637–642, 1952.

[112] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina, *et al.*, "State-of-the-art speech

recognition with sequence-to-sequence models," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4774–4778, IEEE, 2018.

[113] M. C. A. Korba, D. Messadeg, R. Djemili, and H. Bourouba, "Robust speech recognition using perceptual wavelet denoising and mel-frequency product spectrum cepstral coefficient features," *Informatica*, vol. 32, no. 3, 2008.

[114] R. Collobert, C. Puhrsch, and G. Synnaeve, "Wav2letter: an end-to-end convnet-based speech recognition system," *arXiv preprint arXiv:1609.03193*, 2016.

[115] R. Polikar *et al.*, "The wavelet tutorial," 1996.

[116] M. Anusuya and S. Katti, "Comparison of different speech feature extraction techniques with and without wavelet transform to kannada speech recognition," *International Journal of Computer Applications*, vol. 26, no. 4, pp. 19–24, 2011.

[117] S. Ranjan, "A discrete wavelet transform based approach to hindi speech recognition," in *2010 international conference on signal acquisition and processing*, pp. 345–348, IEEE, 2010.

[118] D. O'Shaughnessy, "Linear predictive coding," *IEEE potentials*, vol. 7, no. 1, pp. 29–32, 1988.

[119] S. Wijoyo and S. Wijoyo, "Speech recognition using linear predictive coding and artificial neural network for controlling movement of mobile robot," in *proceedings of 2011 international conference on information and electronics engineering (ICIEE 2011)*, pp. 28–29, 2011.

[120] K. Gupta and D. Gupta, "An analysis on lpc, rasta and mfcc techniques in automatic speech recognition system," in *2016 6th international conference-cloud system and big data engineering (confluence)*, pp. 493–497, IEEE, 2016.

[121] L. Grama and C. Rusu, "Audio signal classification using linear predictive coding and random forests," in *2017 International conference on speech technology and human-computer dialogue (SpeD)*, pp. 1–9, IEEE, 2017.

[122] B. H. Juang and L. R. Rabiner, "Hidden markov models for speech recognition," *Technometrics*, vol. 33, no. 3, pp. 251–272, 1991.

[123] Ø. Birkenes, T. Matsui, K. Tanabe, S. M. Siniscalchi, T. A. Myrvoll, and M. H. Johnsen, "Penalized logistic regression with hmm log-likelihood regressors for speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 6, pp. 1440–1454, 2009.

[124] H. Tang, C.-H. Meng, and L.-S. Lee, "An initial attempt for phoneme recognition using structured support vector machine (svm)," in *2010 IEEE international conference on acoustics, speech and signal processing*, pp. 4926–4929, IEEE, 2010.

[125] R. Solera-Ureña, J. Padrell-Sendra, D. Martín-Iglesias, A. Gallardo-Antolín, C. Peláez-Moreno, and F. Díaz-de María, "Svms for automatic speech recognition: a survey," in *Progress in nonlinear speech processing*, pp. 190–216, Springer, 2007.

[126] S. E. Krüger, M. Schafföner, M. Katz, E. Andelic, and A. Wendemuth, "Speech recognition with support vector machines in a hybrid system.," in *Interspeech*, pp. 993–996, Citeseer, 2005.

[127] B. Wang, Y. Yin, and H. Lin, "Attention-based transducer for online speech recognition," *arXiv preprint arXiv:2005.08497*, 2020.

[128] J. Islam, M. Mubassira, M. R. Islam, and A. K. Das, "A speech recognition system for bengali language using recurrent neural network," in *2019 IEEE 4th international conference on computer and communication systems (ICCCS)*, pp. 73–76, IEEE, 2019.

[129] A. Shewalkar, "Performance evaluation of deep neural networks applied to speech recognition: Rnn, lstm and gru," *Journal of Artificial Intelligence and Soft Computing Research*, vol. 9, no. 4, pp. 235–245, 2019.

[130] T. Makino, H. Liao, Y. Assael, B. Shillingford, B. Garcia, O. Braga, and O. Siohan, "Recurrent neural network transducer for audio-visual speech recognition," in *2019 IEEE automatic speech recognition and understanding workshop (ASRU)*, pp. 905–912, IEEE, 2019.

[131] D. Palaz, R. Collobert, *et al.*, "Analysis of cnn-based speech recognition system using raw speech as input," tech. rep., Idiap, 2015.

[132] S. Park, Y. Jeong, and H. S. Kim, "Multiresolution cnn for reverberant speech recognition," in *2017 20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA)*, pp. 1–4, IEEE, 2017.

[133] S. Ganapathy and V. Peddinti, "3-d cnn models for far-field multi-channel speech recognition," in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5499–5503, IEEE, 2018.

[134] L. Besacier, E. Barnard, A. Karpov, and T. Schultz, "Automatic speech recognition for under-resourced languages: A survey," *Speech communication*, vol. 56, pp. 85–100, 2014.

[135] Z. Geng, H. Yan, J. Zhang, and D. Zhu, "Deep-learning for radar: A survey," *IEEE Access*, vol. 9, pp. 141800–141818, 2021.

[136] P. Lang, X. Fu, M. Martorella, J. Dong, R. Qin, X. Meng, and M. Xie, "A comprehensive survey of machine learning applied to radar signal processing," *arXiv preprint arXiv:2009.13702*, 2020.

[137] J. Lombacher, K. Laudt, M. Hahn, J. Dickmann, and C. Wöhler, "Semantic radar grids," in *2017 IEEE intelligent vehicles symposium (IV)*, pp. 1170–1175, IEEE, 2017.

[138] C. X. Lu, S. Rosa, P. Zhao, B. Wang, C. Chen, J. A. Stankovic, N. Trigoni, and A. Markham, "See through smoke: robust indoor mapping with low-cost mmwave radar," in *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*, pp. 14–27.

[139] A. Ouaknine, A. Newson, P. Pérez, F. Tupin, and J. Rebut, "Multi-view radar semantic segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15671–15680, 2021.

[140] S. Solonska, V. Zhyrnov, and O. Holovin, "Semantic processing of radar spectral information for air object recognition," in *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T)*, pp. 751–754, IEEE, 2019.

[141] I. Shubin, S. Solonska, S. Snisar, V. Slavhorodskyi, and V. Skovorodnikova, "Semantic radar technology for detecting and recognizing low-visible air objects," in *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T)*, pp. 665–670, IEEE, 2019.

[142] S. Solonska and V. Zhyrnov, "Adaptive semantic analysis of radar data using fuzzy transform," in *Data-Centric Business and Applications*, pp. 157–179, Springer, 2021.

[143] I. Shubin, S. Solonska, S. Snisar, V. Zhyrnov, V. Slavhorodskyi, and V. Skovorodnikova, "Efficiency evaluation for radar signal processing on the basis of spectral-semantic model," in *2020 IEEE 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET)*, pp. 171–174, IEEE, 2020.

[144] G. Shi, D. Gao, X. Song, J. Chai, M. Yang, X. Xie, L. Li, and X. Li, "A new communication paradigm: from bit accuracy to semantic fidelity," *arXiv preprint arXiv:2101.12649*, 2021.

[145] E. Uysal, O. Kaya, A. Ephremides, J. Gross, M. Codreanu, P. Popovski, M. Assaad, G. Liva, A. Munari, T. Soleymani, *et al.*, "Semantic communications in networked systems: A data significance perspective," *arXiv preprint arXiv:2103.05391*, 2021.

[146] M. Kountouris and N. Pappas, "Semantics-empowered communication for networked intelligent systems," *IEEE Communications Magazine*, vol. 59, no. 6, pp. 96–102, 2021.

[147] Q. Lan, D. Wen, Z. Zhang, Q. Zeng, X. Chen, P. Popovski, and K. Huang, "What is semantic communication? a view on conveying meaning in the era of machine intelligence," *Journal of Communications and Information Networks*, vol. 6, no. 4, pp. 336–371, 2021.

[148] Q. Zhou, R. Li, Z. Zhao, C. Peng, and H. Zhang, "Semantic communication with adaptive universal transformer," *IEEE Wireless Communications Letters*, vol. 11, no. 3, pp. 453–457, 2021.

[149] A. Graves, "Adaptive computation time for recurrent neural networks," *arXiv preprint arXiv:1603.08983*, 2016.

[150] S. Guo and Y. Wang, "Signal shaping for semantic communications," *arXiv preprint arXiv:2202.02072*, 2022.

[151] P. Jiang, C.-K. Wen, S. Jin, and G. Y. Li, "Deep source-channel coding for sentence semantic transmission with harq," *IEEE Transactions on Communications*, 2022.

[152] Z. Weng and Z. Qin, "Semantic communication systems for speech transmission," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2434–2444, 2021.

[153] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.

[154] Y. Yang, C. Guo, F. Liu, C. Liu, L. Sun, Q. Sun, and J. Chen, "Semantic communications with artificial intelligence tasks: Reducing bandwidth requirements and improving artificial intelligence task performance," *IEEE Industrial Electronics Magazine*, 2022.

[155] Z. Zhang, Q. Yang, S. He, M. Sun, and J. Chen, "Wireless transmission of images with the assistance of multi-level semantic information," *arXiv preprint arXiv:2202.04754*, 2022.

[156] K. Lu, Q. Zhou, R. Li, Z. Zhao, X. Chen, J. Wu, and H. Zhang, "Rethinking modern communication from semantic coding to semantic communication," *IEEE Wireless Communications*, 2022.

[157] H. Zhang, S. Shao, M. Tao, X. Bi, and K. B. Letaief, "Deep learning-enabled semantic communication systems with task-unaware transmitter and dynamic data," *arXiv preprint arXiv:2205.00271*, 2022.

[158] Q. Hu, G. Zhang, Z. Qin, Y. Cai, and G. Yu, "Robust semantic communications against semantic noise," *arXiv preprint arXiv:2202.03338*, 2022.

[159] H. Yoo, T. Jung, L. Dai, S. Kim, and C.-B. Chae, "Real-time semantic communications with a vision transformer," *arXiv preprint arXiv:2205.03886*, 2022.

[160] T.-Y. Tung, D. B. Kurka, M. Jankowski, and D. Gunduz, "Deepjscc-q: Constellation constrained deep joint source-channel coding," *arXiv preprint arXiv:2206.08100*, 2022.

[161] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[162] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

[163] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[164] A. Van Den Oord, O. Vinyals, *et al.*, "Neural discrete representation learning," *Advances in neural information processing systems*, vol. 30, 2017.

[165] H. Xie, Z. Qin, and G. Y. Li, "Task-oriented multi-user semantic communications for vqa," *IEEE Wireless Communications Letters*, vol. 11, no. 3, pp. 553–557, 2021.

[166] D. A. Hudson and C. D. Manning, “Compositional attention networks for machine reasoning,” *arXiv preprint arXiv:1803.03067*, 2018.

[167] T.-Y. Tung and D. Gündüz, “Deepwive: Deep-learning-aided wireless video transmission,” *arXiv preprint arXiv:2111.13034*, 2021.

[168] F. Zhou, Y. Li, X. Zhang, Q. Wu, X. Lei, and R. Q. Hu, “Cognitive semantic communication systems driven by knowledge graph,” *arXiv preprint arXiv:2202.11958*, 2022.

[169] S. Jiang, Y. Liu, Y. Zhang, P. Luo, K. Cao, J. Xiong, H. Zhao, and J. Wei, “Reliable semantic communication system enabled by knowledge graph,” *Entropy*, vol. 24, no. 6, p. 846, 2022.

[170] T.-Y. Tung, S. Kobus, J. P. Roig, and D. Gündüz, “Effective communications: A joint learning and communication framework for multi-agent reinforcement learning over noisy channels,” *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2590–2603, 2021.

[171] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, “Human-level control through deep reinforcement learning,” *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[172] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning,” *arXiv preprint arXiv:1509.02971*, 2015.

[173] J. Liang, Y. Xiao, Y. Li, G. Shi, and M. Bennis, “Life-long learning for reasoning-based semantic communication,” *arXiv preprint arXiv:2202.01952*, 2022.

[174] H. Seo, J. Park, M. Bennis, and M. Debbah, “Semantics-native communication with contextual reasoning,” *arXiv preprint arXiv:2108.05681*, 2021.

[175] M. K. Farshbafan, W. Saad, and M. Debbah, “Curriculum learning for goal-oriented semantic communications with a common language,” *arXiv preprint arXiv:2204.10429*, 2022.

[176] L. Cohen, *Time-frequency analysis.* PTR Prentice Hall, 2012.

[177] Y. Chan, *Wavelet basics.* Springer Science & Business Media, 1994.

[178] C. Torrence and G. P. Compo, "A practical guide to wavelet analysis," *Bulletin of the American Meteorological society*, vol. 79, no. 1, pp. 61–78, 1998.

[179] A. K. Özdemir, S. Karakaş, E. D. Çakmak, D. İlhan Tüfekçi, and O. Arıkan, "Time–frequency component analyser and its application to brain oscillatory activity," *Journal of Neuroscience Methods*, vol. 145, no. 1, pp. 107–125, 2005.

[180] M. Anandarajan, C. Hill, and T. Nolan, "Text preprocessing," in *Practical Text Analytics*, pp. 45–59, Springer, 2019.

[181] D. W. Otter, J. R. Medina, and J. K. Kalita, "A survey of the usages of deep learning for natural language processing," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 2, pp. 604–624, 2020.

[182] V. Carletti, P. Foggia, A. Saggese, and M. Vento, "Introducing vf3: A new algorithm for subgraph isomorphism," in *International Workshop on Graph-Based Representations in Pattern Recognition*, pp. 128–139, Springer, 2017.

[183] W.-S. Han, J. Lee, and J.-H. Lee, "Turboiso: towards ultrafast and robust subgraph isomorphism search in large graph databases," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pp. 337–348, 2013.

[184] H. Shang, Y. Zhang, X. Lin, and J. X. Yu, "Taming verification hardness: an efficient algorithm for testing subgraph isomorphism," *Proceedings of the VLDB Endowment*, vol. 1, no. 1, pp. 364–375, 2008.

[185] Z. Sun, H. Wang, H. Wang, B. Shao, and J. Li, "Efficient subgraph matching on billion node graphs," *arXiv preprint arXiv:1205.6691*, 2012.

[186] P. Zhao and J. Han, "On graph query optimization in large networks," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 340–351, 2010.

[187] H. Ren, W. Hu, and J. Leskovec, "Query2box: Reasoning over knowledge graphs in vector space using box embeddings," *arXiv preprint arXiv:2002.05969*, 2020.

[188] H. Ren, H. Dai, B. Dai, X. Chen, D. Zhou, J. Leskovec, and D. Schuurmans, "Smore: Knowledge graph completion and multi-hop reasoning in massive knowledge graphs," *arXiv preprint arXiv:2110.14890*, 2021.

[189] X. Zhang, A. Bosselut, M. Yasunaga, H. Ren, P. Liang, C. D. Manning, and J. Leskovec, "Greaselm: Graph reasoning enhanced language models," in *International Conference on Learning Representations*, 2021.

[190] H. Ren, H. Dai, B. Dai, X. Chen, M. Yasunaga, H. Sun, D. Schuurmans, J. Leskovec, and D. Zhou, "Lego: Latent execution-guided reasoning for multi-hop question answering on knowledge graphs," in *International Conference on Machine Learning*, pp. 8959–8970, PMLR, 2021.

[191] Y. Wang, M. Yasunaga, H. Ren, S. Wada, and J. Leskovec, "Vqa-gnn: Reasoning with multimodal semantic graph for visual question answering," *arXiv preprint arXiv:2205.11501*, 2022.

[192] S. Ji, S. Pan, E. Cambria, P. Marttinen, and S. Y. Philip, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 2, pp. 494–514, 2021.

[193] R. E. Hall, B. Bowerman, J. Braverman, J. Taylor, H. Todosow, and U. Von Wimmersperg, "The vision of a smart city," tech. rep., Brookhaven National Lab., Upton, NY (US), 2000.

[194] K. Su, J. Li, and H. Fu, "Smart city and the applications," in *International Conference on Electronics, Communications and Control (ICECC)*, pp. 1028–1031, IEEE, 2011.

[195] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *2017 IEEE international conference on image processing (ICIP)*, pp. 3645–3649, IEEE, 2017.

[196] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft COCO: common objects in context," 2015.

[197] X. Du, T.-Y. Lin, P. Jin, G. Ghiasi, M. Tan, Y. Cui, Q. V. Le, and X. Song, "Spinenet: Learning scale-permuted backbone for recognition and localization," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11592–11601, 2020.

[198] G. Welch, G. Bishop, *et al.*, *An introduction to the Kalman filter.* Chapel Hill, NC, USA, 1995.

[199] N. Wojke and A. Bewley, "Deep cosine metric learning for person reidentification," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 748–756, IEEE, 2018.

[200] R. Jonker and T. Volgenant, "Improving the hungarian assignment algorithm," *Operations Research Letters*, vol. 5, no. 4, pp. 171–175, 1986.

[201] A. Baggio, "Wireless sensor networks in precision agriculture," in *ACM Workshop on Real-World Wireless Sensor Networks (REALWSN 2005), Stockholm, Sweden*, vol. 20, pp. 1567–1576, Citeseer, 2005.

[202] M. Srbinovska, C. Gavrovski, V. Dimcev, A. Krkoleva, and V. Borozan, "Environmental parameters monitoring in precision agriculture using wireless sensor networks," *Journal of Cleaner Production*, vol. 88, pp. 297–307, 2015.

[203] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, 1952.

[204] W. B. Pennebaker and J. L. Mitchell, *JPEG: Still image data compression standard.* Springer Science & Business Media, 1992.

[205] R. G. Baraniuk, "Compressive sensing [lecture notes]," *IEEE Signal Processing Magazine*, vol. 24, no. 4, pp. 118–121, 2007.

[206] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

[207] X. Ma, J. Wu, S. Xue, J. Yang, C. Zhou, Q. Z. Sheng, H. Xiong, and L. Akoglu, “A comprehensive survey on graph anomaly detection with deep learning,” *IEEE Transactions on Knowledge and Data Engineering*, 2021.