University of Modena and Reggio Emilia
XXXII cycle of the International Doctorate School in
Information and Communication Technologies
Doctor of Philosophy dissertation in
Computer Engineering and Science

Identification of Anomalies in Driver Attention and People Behavior

Davide Abati

Supervisor: Prof. Rita Cucchiara
PhD school director: Prof. Sonia Bergamaschi

Modena, 2020

Review committee composed of:

Prof. Octavia Camps
Full Professor
Electrical and Computer Engineering, Northeastern University, Boston (MA)

Prof. Giovanni Maria Farinella
Assistant Professor
Department of Mathematics and Informatics, University of Catania

The progress of our modern age, and of the world of tomorrow, depends not only on technical expertise, but also on unobstructed curiosity and the benefits - and pleasures - of traveling far upstream, against the current of practical considerations.

Robbert Dijkgraaf

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As the world matures increasingly connected and digitized by the day, with sensors and computing devices becoming more and more pervasive, new opportunities appear for artificial intelligence. In particular, computer vision can forcefully prevail as one of the lead technologies delivering intelligent systems, with applications including (but not limited to) automotive, public monitoring and video surveillance.

In this thesis, we present computer vision solutions to tackle challenging research problems in different areas of operation.
We first focus on the automotive setting, where assistance technologies can significantly improve the safety of human drivers. In this frame, we develop of a computational model capable of predicting where a driver is likely to focus his attention on the outside scene. Such a system can enhance existing driver monitoring systems: indeed, information about the conventional attentional behavior can be useful whenever the actual gaze of the driver shifts from such an expectation, potentially indicating drowsiness or distraction. Despite such a problem appears utterly complex, as driving a car is a highly subjective task from an attentive perspective, we show how our deep learning solution profitably learns to reproduce human attention from raw videoclips.
Then, we move to video surveillance settings, and we tackle the problem of anomaly detection. Specifically, we devise a system capable of learning the traits that characterize normal (safe) situations (e.g., people walking or loitering) and of triggering alerts whenever unexpected events appear (e.g. thefts, brawls). Typ-

ically, anomaly detection (novelty detection) research requires such models to learn without utilizing any example of dangerous conditions, or prior knowledge about what type of anomalies can be expected in the test phase. Consequently, despite the importance of the topic and a plethora of prior work, performances of state-of-the-art systems are still affected by the unpredictable nature of novel events and their inaccessibility during the training procedure.
Finally, in the last part of the thesis we focus on online continual learning of deep models. This research is motivated by the fact that a desirable feature for anomaly detection models involves their capability, when trained in online settings and numerous normal scenarios are encountered sequentially, of including the new concepts as 'normal' without discarding the previously acquired ones. However, such a necessity is problematic, as neural networks are known to struggle in incremental learning settings, as they tend to catastrophically forget previously acquired knowledge. As part of a collaboration with Qualcomm AI Research, we propose a solution to overcome the forgetting problem in neural networks and, as a first effort, we demonstrate its suitability to benefit classification models.

Summary of contributions

As mentioned, this thesis will present several contributions in three research areas of focus of attention prediction, novelty detection and continual learning. For easiness of reading, we report in what follows a summary of each of them.

Driver attention prediction. Our research on attention prediction led to multiple contributions. First, in order to cope with the lack of public datasets, we collect and release `DR(eye)VE`, a dataset consisting of driver-centric and car-centric clips, along with driver's fixation points on the outer scene (Sec. 3.1). Furthermore, we present an analysis of the common drivers' attentional patterns, highlighting a clear attractor towards the vanishing point of the road, as well as strong correlations between eye fixations and car speed or scene semantics (Sec. 3.2). Guided by such evidence, we develop in Sec. 3.3 the first computational model (i.e. a multimodal neural network) capable of identifying, within a car-centric view of an urban scene, which regions are likely to capture the driver's attention. Finally, in Sec. 3.4 we report an extensive experimental analysis of the proposed model, assessing its performance against prior models for attention prediction both quantitatively and qualitatively. Moreover, we show how some peculiar human attentive behaviors are successfully reproduced by the model (Sec. 3.4.3.3), and finally report a user study concerning the perceived safety of ground-truth

and predicted attentional maps (Sec. 3.4.4).

Novelty detection. The common practice in novelty detection literature is to train encoder-decoder models (i.e. deep autoencoders) on the normal training distribution, and to assume that out-of-distribution samples will lead to higher residuals. In our research, we introduce a suitable latent-space regularizer enhancing the whole training process. Specifically, we pair a standard deep autoencoder with a neural density estimator, fitting its latent representations through an autoregressive procedure (Sec. 4.1). We provide an information theoretic justification for the estimator objective in Sec. 4.1.2, as well as intuitive interpretations and empirical evidences of its impact. Moreover, we stress the capability of the proposed model both in one-class classification (Sec. 4.2.1) and video anomaly detection (Sec. 4.2.2), where we show similar or superior performance with respect to prior methods, whilst not relying on any data-related assumption. Finally, we show our model is capable of automatically identifying peculiar attentive situations (e.g. distractions) within the `DR(eye)VE` dataset (Sec. 4.2.4).

Continual learning. Current models for continual learning concern settings in which a single neural network is trained on multiple classification problems sequentially. At test time, the network and is queried for the class of unlabeled examples, assuming knowledge about the task they belong to. In our research, we strive for a model capable of performing classification in absence of any task-related information during test. To this aim, we introduce the first conditional-computation model within continual learning (Sec. 5.1). First, we show how the employment of data-dependent gating schemes can solve the forgetting problem whenever task-label information is available (Sec. 5.1.2). Then, we illustrate how the adoption of a supervised task classifier can be used during test in lieu of a task oracle (Sec. 5.1.3). By experimental analysis, we validate our model on standard benchmarks, showing superior performance with respect to prior art both in presence and absence of task labels (Sec. 5.2.2, 5.2.3, 5.2.4).
This research has been carried out during a four month internship at Qualcomm AI Research in Amsterdam.

Other activities

Beside the research constituting the main corpus of the thesis and the ones briefly illustrated in the Appendix, I was involved in several supplemental activities, such as teaching and other services. In the remainder of the chapter I will report them.

Partecipation to national projects

- MIUR PRIN project *"PREVUE: PRediction of activities and Events by Vision in an Urban Environment"*, grant ID E94I19000650001.
- ECSEL JU project *"PRYSTINE: Programmable Systems for Intelligence in Automobiles"*, grant n.783190

Collaborations

- Internship at Qualcomm AI Research, Amsterdam - June to September, 2019

Teaching activities

- Lecturer on Reinforcement Learning - Pattern Recognition and Machine Learning - Prof. Simone Calderara (2018 - 2019)
- Laboratory lecturer - AI-DLDA International Summer School on Artificial Intelligence (2018)
- Laboratory lecturer - Pattern Recognition and Machine Learning - Prof. Simone Calderara (2017 - 2019)
- Laboratory lecturer - Computer Vision - Prof. Rita Cucchiara (2017)
- Laboratory lecturer - International Master of II level in Visual Computing and Multimedia Technologies (2017)

Master thesis supervision

- "Continual Learning via Logits Distillation" - Dott. Pietro Buzzega
- "Visual Reasoning: Learning and Composing Primitives in an Associative Memory" - Dott. Rosario Di Carlo

Journals Reviewing

- IEEE Transactions on Multimedia
- IEEE Transactions on Intelligent Transportation Systems

- ACM Transactions on Multimedia Computing, Communications, and Applications

Conferences, tutorials and schools

- International Conference on Computer Vision and Pattern Recognition - CVPR, 2019
- Regularization Methods for Machine Learning - RegML, Genova, 2018
- International Conference on Computer Vision - ICCV, Venice, 2017
- International Computer Vision Summer School - ICVSS, Sicily, 2017
- 13th Italian Research Conference on Digital Libraries - IRCDL, Modena, 2017

Seminars attended

- "Computational And Experimental Neuroscience Toward Artificial Intelligence" - Prof. Jonathan Mapelli (University of Modena and Reggio Emilia) - April 10, 2018
- "Coordination of autonomous vehicles" - Prof. Giacomo Cabri (University of Modena and Reggio Emilia) - April 5, 2018
- "Deep Learning for Fault Prediction" - Prof. Roberto Paredes (Universitat Politècnica de València, UPV) - February 12-15, 2018
- "Visual Appearance Acquisition of Real Objects" - Doc. Massimiliano Corsini (CNR) - February 8, 2018
- "Deep Learning Technologies: from hardware components to vertical frameworks" - Doc. Piero Altoè (NVIDIA) - November 29, 2017
- "Quantum technologies and security: potential use-cases and risks" - Prof. Enrico Prati (CNR) - November 21, 2017
- "The eye of the machine" - Prof. Simone Arcagni (University of Palermo) - September 29, 2017
- "Academic English Workshop" - Doc. Silvia Cavalieri - May 9-23, 2017

- "Internet privacy: towards more transparency" - Prof. Balachander Krishnamurthy (AT&T Labs Research) - November 21, 2016
- "Handwritten Text Recognition" - Prof. Moisés Pastor - PRHLT Research Center (Universitat Politècnica de València, UPV) - November 7-11, 2016
- "Hidden Markov Models and Selected Applications" - Prof. Jon Ander Gomez Adrian (Universitat Politècnica de València, UPV) - October 24-26, 2016
- "Syncronization problems in Computer Vision" - Prof. Andrea Fusiello (University of Udine) - October 21, 2016
- "Multi camera tracking: following people in large camera networks" - Doc. Ergys Ristani (Duke University) - October 18, 2016

Chapter 2

Literature Survey

In this section, we discuss the existing solutions and relevant literature for the research problems tackled in this thesis. Specifically, Sec. 2.1 describes prior art in attention prediction (also with a special focus on the driving setting), whereas Sec. 2.2 reports previous research in the topic of anomaly detection and the employment of autoregressive techniques for the purpose of learning deep probability density models. Finally, Sec. 2.3 discusses recent works in the topics of continual learning and conditional computation with neural networks.

2.1 Driver focus of attention prediction

The way humans favor some entities in the scene, along with key factors guiding eye fixations in presence of a given task (e.g. visual search) has been extensively studied for decades [182, 200]. The main difficulty that rises when approaching the subject is the variety of perspectives under which it can be cast. Indeed, visual attention has been approached by psychologists, neurobiologists and computer scientists, making the field highly interdisciplinary [48]. We are particularly interested in the computational perspective, in which predicting human attention is often formalized as an estimation task delivering the probability of each point in a given scene to attract the observer's gaze.

2.1.1 Attention in images and videos

Coherently with psychological literature, that identifies two distinct mechanisms guiding human eye fixations [177], computational models for FoA prediction branch into two families: top-down and bottom-up strategies. Former approaches aim at highlighting objects and cues that could be meaningful in the context of a given task. For this reason, such methods are also known as task-driven. Usually, top-down computer vision models are built to integrate semantic contextual information in the attention prediction process [180]. This can be achieved by either merging estimated maps at different levels of scale and abstraction [55], or including a-priori cues about relevant objects for the task at hand [201, 51, 42]. Human focus in complex interactive environments (*e.g.* while playing videogames) [140, 141, 20] follows task-driven behaviors as well.

Conversely, bottom-up models capture salient objects or events naturally popping out in the image, independently of the observer, the undergoing task and other external factors. This task is widely known in literature as *visual saliency prediction*. In this context, computational models focus on spotting visual discontinuities, either by clustering features or considering the rarity of image regions, locally [164, 114] or globally [1, 208, 31]. For a comprehensive review of visual attention prediction methods, we refer the reader to [18]. Recently, the success of deep networks involved both task-driven attention and saliency prediction, as models have become more powerful in both paradigms, achieving state-of-the-art results on public benchmarks [93, 104, 65, 36, 37].

In video, attention prediction and saliency estimation are more complex with respect to still images since motion heavily affects human gaze. Some models merge bottom-up saliency with motion maps, either by means of optical flow [212] or feature tracking [208]. Other methods enforce temporal dependencies between bottom-up features in successive frames. Both supervised [212, 171] and unsupervised [121, 196, 197] feature extraction can be employed, and temporal coherence can be achieved either by conditioning the current prediction on information from previous frames [155] or by capturing motion smoothness with optical flow [212, 171]. While deep video saliency models still lack, an interesting work is [13], which relies on a recurrent architecture fed with clip encodings to predict the fixation map by means of a Gaussian Mixture Model (GMM). Nevertheless, most methods limit to bottom-up features accounting for just visual discontinuities in terms of textures or contours. Our proposal, instead, is specifically tailored to the driving task and fuses the bottom-up information with semantics and motion elements that have emerged as attention factors from the

analysis of the DR(eye)VE dataset.

2.1.2 Attention and driving

Prior works addressed the task of detecting saliency and attention in the specific context of assisted driving. In such cases, however, gaze and attentive mechanisms have been mainly studied for some driving sub-tasks only, often acquiring gaze maps from on-screen images. Bremond *et al.* [169] presented a model that exploits visual saliency with a non-linear SVM classifier for the detection of traffic signs. The validation of this study was performed in a laboratory non-realistic setting, emulating an in-car driving session. A more realistic experiment [21] was then conducted with a larger set of targets, *e.g.* including pedestrians and bicycles.
Driver's gaze has also been studied in a pre-attention context, by means of intention prediction relying only on fixation maps [144]. The study in [185] inspects the driver's attention at T junctions, in particular towards pedestrians and motorbikes, and exploits object saliency to avoid the *looked-but-failed-to-see* effect. In absence of eye tracking systems and reliable gaze data, [47, 175, 193, 16] focus on drivers' head, detecting facial landmarks to predict head orientation. Such mechanisms are more robust to varying lighting conditions and occlusions, but there is no certainty about the adherence of predictions to the true gaze during the driving task.

2.1.3 Datasets

Many image saliency datasets have been released in the past few years, improving the understanding of the human visual attention and pushing computational models forward. Most of these datasets include no motion information, as saliency ground truth maps are built by aggregating fixations of several users within the same still image. Usually, a Gaussian filtering post-processing step is employed on recorded data, in order to smooth such fixations and integrate their spatial locations. Some datasets, such as the MIT saliency benchmark [23], were labeled through an eye tracking system, while others, like the SALICON dataset [75] relied on users clicking on salient image locations. We refer the reader to [19] for a comprehensive list of available datasets. On the contrary, datasets addressing human attention prediction in video still lack. Up to now, *Action in the Eye* [171] represents the most important contribution, since it consists in the largest video

dataset accompanied by gaze and fixation annotations. That information, however, is collected in the context of action recognition, so it is heavily task-driven. A few datasets address directly the study of attention mechanisms while driving, as summarized in Tab. 3.1. However, these are mostly restricted to limited settings and are not publicly available. In some of them [169, 185] fixation and saliency maps are acquired during an in-lab simulated driving experience. In-lab experiments enable several attention drifts that are influenced by external factors (*e.g.* monitor distance and others) rather than the primary task of driving [174]. A few in-car datasets exist [21, 144], but were precisely tailored to force the driver to fulfill some tasks, such as looking at people or traffic signs. Coarse gaze information is also available in [47], while the external road scene images are not acquired. We believe that the dataset presented in [144] is, among the others, the closer to our proposal. Yet, video sequences are collected from one driver only it is not publicly available. Conversely, our `DR(eye)VE` dataset is the first dataset addressing driver's focus of attention prediction that is made publicly available. Furthermore, it includes sequences from several different drivers and presents a high variety of landscapes (*i.e.* highway, downtown and countryside), lighting and weather conditions.

2.2 Anomaly detection

2.2.1 Reconstruction-based methods

On the one hand, many works lean toward learning a parametric projection and reconstruction of normal data, assuming outliers will yield higher residuals. Traditional sparse-coding algorithms [210, 34, 110] adhere to such framework, and represent normal patterns as a linear combination of a few basis components, under the hypotheses that novel examples would exhibit a non-sparse representation in the learned subspace. In recent works, the projection step is typically drawn from deep autoencoders [59]. In [112] the authors recover sparse coding principles by imposing a sparsity regularization over the learned representations, while a recurrent neural network enforces their smoothness along the time dimension. In [159], instead, the authors take advantage of an adversarial framework in which a discriminator network is employed as the actual novelty detector, spotting anomalies by performing a discrete in-distribution test. Oppositely, future frame prediction [105] maximizes the expectation of the next frame exploiting its knowledge of the past ones; at test time, observed deviations against the

predicted content advise for abnormality. Differently from the above-mentioned works, our proposal relies on modeling the prior distribution of latent representations. This choice is coherent with recent works from the density estimation community [179, 12]. However, to the best of our knowledge, our work is the first advocating for the importance of such a design choice for novelty detection.

2.2.2 Probabilistic methods

A complementary line of research investigates different strategies to approximate the density function of normal appearance and motion features. The primary issue raising in this field concerns how to estimate such densities in a high-dimensional and complex feature space. In this respect, prior works involve hand-crafted features such as optical flow or trajectory analysis and, on top of that, employ both non-parametric [2] and parametric [11, 116, 100] estimators, as well as graphical modeling [79, 95].
Modern approaches rely on deep representations (e.g., captured by autoencoders), as in Gaussian classifiers [158] and Gaussian Mixtures [213]. In [63] the authors involve a Kernel Density Estimator (KDE) modeling activations from an auxiliary object detection network.
A recent research trend considers training Generative Adversarial Networks (GANs) on normal samples. However, as such models approximate an implicit density function, they can be queried for new samples but not for likelihood values. Therefore, GAN-based models employ different heuristics for the evaluation of novelty. For instance, in [161] a guided latent space search is exploited to infer it, whereas [146] directly queries the discriminator for a normality score.

2.2.3 Autoregressive density estimation

The first deep autoregressive models were introduced to overcome the issues affecting Restricted Boltzmann Machines [170, 46] in modeling the density of binary inputs. NADE [96] and its real-valued version RNADE [186] employ a deep network to estimate conditional probabilities of unobserved variables. The works in [187, 53] introduce further improvements, by introducing a training strategy copying with all possible autoregressive orderings among variables. When employed directly on color space, autoregression techniques lead to powerful generative models [190, 189] such as Pixel-CNN, capable of hallucinating new images sequentially and conditioning the probability distribution of an unknown pixel on the observed ones. Autoregressive models [170, 96, 186] constitute a powerful

tool to factorize a joint distribution, thus avoid to define a priori knowledge of its landscape explicitly. Despite the chain rule is true in general, estimating CPDs can be easier for some peculiar variable ordering and harder for others, depending on the nature of the joint variable space. Some prior models obey handcrafted orderings [190, 189], whereas others rely on order agnostic training [187, 53]. Nevertheless, it is still not clear how to estimate the proper order for a given set of variables. Differently, in our model, this issue is directly tackled by optimization, aimed at model density underlying the latent space representations. A recent line of research merges autoregression with attention and meta-learning techniques for few-shot density estimation [148], as well as the parallel framework of normalizing flows in [82, 137]. Finally, other unsupervised approaches involve crowd trajectory analysis [123, 25], video volumes clustering [211, 154], foreground explainability [6] and discriminability of normal frames with respect to context [39, 70]. For a comprehensive review of traditional anomaly detection techniques in computer vision, we refer the reader to [142].

2.3 Channel gated network for continual learning

2.3.1 Continual learning

Catastrophic forgetting has been a well-known problem of neural networks [122]. Early approaches to alleviate the issue involved orthogonal representation learning and replay of prior samples [45]. The recent advent in deep learning has led to the widespread use of deep neural networks in the continual learning field. First attempts, such as Progressive Neural Networks [157] tackle the forgetting problem by introducing a new set of parameters for each new task at the expense of limited scalability. Another popular solution is to apply knowledge distillation by using the past parametrizations of the model as a reference when learning new tasks [101].

Consolidation approaches emerged recently with the focus of identifying the weights that are critically important for prior tasks and preventing significant updates to them during the learning of new tasks. The relevance/importance estimation for each parameter can be carried out through the Fisher Information Matrix [85], the path integral of loss gradients [207], gradient magnitude [3] and a posteriori uncertainty estimation in a Bayesian Neural Network [130].

Other popular consolidation strategies rely on the estimation of binary masks that directly map each task to the set of parameters responsible for it. Such masks

can be estimated either by random assignment [120], pruning [118] or gradient descent [117, 165]. However, existing mask-based approaches can only operate in the presence of an oracle providing the task label. Our work is akin to the above-mentioned models, with two fundamental differences: i) our binary masks (gates) are dynamically generated and depend on the network input, and ii) we promote mask-based approaches to class-incremental learning settings, by relying on a novel architecture comprising a task classifier.
Several models allow access to a finite-capacity memory buffer (*episodic* memory), holding examples from prior tasks. A popular approach is iCaRL [147], which computes class prototypes as the mean feature representation of stored memories, and classifies test examples in a nearest-neighbor fashion. Alternatively, other approaches intervene in the training algorithm, proposing to adjust the gradient computed on the current batch towards an update direction that guarantees non-destructive effects on the stored examples [108, 27, 151]. Such an objective can imply the formalization of constrained optimization problems [108, 27] or the employment of meta-learning algorithms [151]. Differently, *generative* memories do not rely on the replay of any real example whatsoever, in favor of generative models from which fake examples of past tasks can be efficiently sampled [168, 202, 134]. In this work, we also rely on either episodic or generative memories to deal with the class-incremental learning setting. However, we carry out replay only to prevent forgetting of the task predictor, thus avoiding to update task-specific classification heads.

2.3.2 Conditional computation

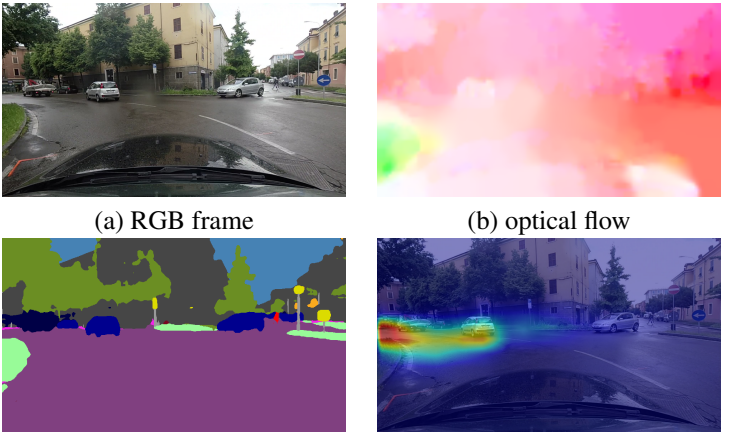
Conditional computation research focuses on deep neural networks that adapt their architecture to the given input. Although the first work has been applied to language modeling [167], several follow-up works explored conditional computation in computer vision problems. In this respect, prior works introduce binary gates deciding whether a computational block has to be executed or skipped. Such a decision impacts the network at different levels, either by dropping entire residual blocks [191, 198] or units in a specific layer [30, 14]. In our work, we rely on the latter strategy, learning a set of task-specific gating modules that select which convolutional kernels to apply on the given input. To our knowledge, this is the first application of data-dependent channel-gating in the framework of continual learning.

Chapter 3

Predicting the Driver's Focus of Attention: the DR(eye)VE Project

According to the J3016 SAE international Standard, which defined the five levels of autonomous driving [57], cars will provide a fully autonomous journey only at the fifth level. At lower levels of autonomy, computer vision and other sensing systems will still support humans in the driving task. Human-centric Advanced Driver Assistance Systems (ADAS) have significantly improved safety and comfort in driving (*e.g.* collision avoidance systems, blind spot control, lane change assistance etc.). Among ADAS solutions, the most ambitious examples are related to monitoring systems [73, 49, 92, 125]: they parse the attention behavior of the driver together with the road scene to predict potentially unsafe manoeuvres and act on the car in order to avoid them – either by signaling the driver or braking. However, all these approaches suffer from the complexity of capturing the true driver's attention and rely on a limited set of fixed safety-inspired rules. In this chapter, we shift the attention prediction problem from a personal level (*what the driver is looking at*) to a task-driven level (*what most drivers would look at*) introducing a computer vision model able to to replicate the human attentional behavior during the driving task.

We achieve this result in two stages: First, we conduct a data-driven study on drivers' gaze fixations under different circumstances and scenarios. The study



(a) RGB frame (b) optical flow

(c) semantic segmentation (d) predicted map

Figure 3.1: An example of visual attention while driving (d), estimated from our deep model using (a) raw video, (b) optical flow and (c) semantic segmentation.

concludes that the semantic of the scene, the speed and bottom-up features all influence the driver's gaze. Second, we advocate for the existence of common gaze patterns that are shared among different drivers. We empirically demonstrate the existence of such patterns by developing a deep learning model that can profitably learn to predict where a driver would be looking at in a specific situation. To this aim we recorded and annotated 555,000 frames (approx. 6 hours) of driving sequences in different traffic and weather conditions: the DR(eye)VE dataset. For every frame we acquired the driver's gaze through an accurate eye tracking device and registered such data to the external view recorded from a roof-mounted camera. The DR(eye)VE data richness enables us to train an end-to-end deep network that predicts salient regions in car-centric driving videos. The network we propose is based on three branches which estimate attentional maps from a) visual information of the scene, b) motion cues (in terms of optical flow) and c) semantic segmentation (Fig. 3.1). In contrast to the majority of experiments, which are conducted in controlled laboratory settings or employ sequences of unrelated images [185, 23, 75], we train our model on data acquired on the field. Final results demonstrate the ability of the network to generalize across different day times, different weather conditions, different landscapes and different drivers.

Eventually, we believe our work can be complementary to the current semantic segmentation and object detection literature [203, 194, 131, 28, 126] by providing a diverse set of information. According to [174], the act of driving combines complex attention mechanisms guided by the driver's past experience, short reactive times and strong contextual constraints. Thus, very little information is needed to drive if guided by a strong focus of attention (FoA) on a limited set of targets: our model aims at predicting them.
The remainder of the chapter is organized as follows. In Sec. 2.1, related works about computer vision and gaze prediction are provided to frame our work in the current state-of-the-art scenario. Sec. 3.1 describes the DR(eye)VE dataset and some insights about several attention patterns that human drivers exhibit. Sec. 3.3 illustrates the proposed deep network to replicate such human behavior, and Sec. 3.4 reports the performed experiments.

3.1 The DR(eye)VE project

In this section we present the DR(eye)VE dataset (Fig. 3.2), the protocol adopted for video registration and annotation, the automatic processing of eye-tracker data and the analysis of the driver's behavior in different conditions.

3.1.1 The dataset

The DR(eye)VE dataset consists of 555,000 frames divided in 74 sequences, each of which is 5 minutes long. Eight different drivers of varying age from 20 to 40, including 7 men and a woman, took part to the driving experiment, that lasted more than two months. Videos were recorded in different contexts, both in terms of landscape (downtown, countryside, highway) and traffic condition, ranging from traffic-free to highly cluttered scenarios. They were recorded in diverse weather conditions (sunny, rainy, cloudy) and at different hours of the day (both daytime and night). Tab. 3.1 recaps the dataset features and Tab. 3.2 compares it with other related proposals. DR(eye)VE is currently the largest publicly available dataset including gaze and driving behavior in automotive settings.

3.1.2 The Acquisition System

The driver's gaze information was captured using the commercial *SMI ETG 2w* Eye Tracking Glasses (ETG). ETG capture attention dynamics also in presence

Figure 3.2: Examples taken from a random sequence of DR(eye)VE. From left to right: frames from the eye tracking glasses with gaze data, from the roof-mounted camera, temporal aggregated fixation maps (as defined in Sec. 3.1) and overlays between frames and fixation maps.

of head pose changes, which occur very often during the task of driving. While a frontal camera acquires the scene at 720p/30fps, users pupils are tracked at 60Hz. Gaze information are provided in terms of eye fixations and saccade movements. ETG was manually calibrated before each sequence for every driver.

Simultaneously, videos from the car perspective were acquired using the *GARMIN VirbX* camera mounted on the car roof (RMC, Roof-Mounted Camera). Such sensor captures frames at 1080p/25fps, and includes further information such as GPS data, accelerometer and gyroscope measurements.

3.1.3 Video-gaze registration

The dataset has been processed to move the acquired gaze from the egocentric (ETG) view to the car (RMC) view. The latter features a much wider field of view (FoV), and can contain fixations that are out of the egocentric view. For instance, this can occur whenever the driver takes a peek at something at the border of this FoV, but doesn't move his head. For every sequence, the two videos were manually aligned to cope with the difference in sensors framerate. Videos were then registered frame-by-frame through a homographic transformation that projects fixation points across views. More formally, at each timestep t the RMC frame I^t_{RMC} and the ETG frame I^t_{ETG} are registered by means of a homography matrix

Table 3.1: Summary of the DR(eye)VE dataset characteristics. The dataset was designed to embody the most possible diversity in the combination of different features. The reader is referred to the dataset presentation [5] for details on each sequence.

# Videos	# Frames	Drivers	Weather conditions
74	555,000	8	sunny
			cloudy
			rainy
Lighting	**Gaze Info**	**Metadata**	**Camera Viewpoint**
day	raw fixations	GPS	driver (720p)
evening	gaze map	car speed	car (1080p)
night	pupil dilation	car course	

Table 3.2: A comparison between DR(eye)VE and other datasets.

Dataset	Frames	Drivers	Scenarios	Annotations	Real-world	Public
Pugeault *et al.* [144]	158,668	–	Countryside, Highway, Downtown	Gaze Maps Driver's Actions	Yes	No
Simon *et al.* [169]	40	30	Downtown	Gaze Maps	No	No
Underwood *et al.* [185]	120	77	Urban Motorway	–	No	No
Fridman *et al.* [47]	1,860,761	50	Highway	Gaze Classes	Yes	No
DR(eye)VE [5]	555,000	8	Countryside, Highway, Downtown	Gaze Maps GPS Speed Course	Yes	Yes

$H_{ETG \rightarrow RMC}$, computed by matching SIFT descriptors [109] from one view to the other (see Fig. 3.3). A further RANSAC [44] procedure ensures robustness to outliers. While homographic mapping is theoretically sound only across planar views - which is not the case of outdoor environments - we empirically found that projecting an object from one image to another always recovered the correct position. This makes sense if the distance between the projected object and the camera is far greater than the distance between the object and the projective plane. In Sec. 3.4.5.4, we derive formal bounds to explain this phenomena.

3.1.4 Fixation map computation.

The pipeline discussed above provides a frame-level annotation of the driver's fixations. In contrast to image saliency experiments [23], there is no clear and indisputable protocol for obtaining continuous maps from raw fixations when acquired in task-driven real-life scenarios. This is even more evident when fixations are collected in task-driven real-life scenarios. The main motivation resides in the fact that observer's subjectivity cannot be removed by averaging different observers' fixations. Indeed two different observers cannot experience the same scene at the same time (*e.g.* two drivers cannot be at the same time in the same point of the street). The only chance to average among different observers would be the adoption of a simulation environment, but it has been proved that the cognitive load in controlled experiments is lower than in real test scenarios and it effects the true attention mechanism of the observer [156]. In our preliminary `DR(eye)VE` release [5], fixation points were aggregated and smoothed by means of a temporal sliding window. In such a way, temporal filtering discarded momentary glimpses that contain precious information about the driver's attention. Following the psychological protocol in [119] and [56], this limitation was overcome in the current release where the new fixation maps were computed without temporal smoothing. Both [119] and [56] highlight the high degree of subjectivity of scene scanpaths in short temporal windows (< 1 sec) and suggest to neglect the fixations pop-out order within such windows. This mechanism also ameliorates the *inhibition of return* phenomenon that may prevent interesting objects to be observed twice in short temporal intervals [143, 61], leading to the underestimation of their importance.
More formally, the *fixation map* F_t for a frame at time t is built by accumulating projected gaze points in a temporal sliding window of $k = 25$ frames, centered in

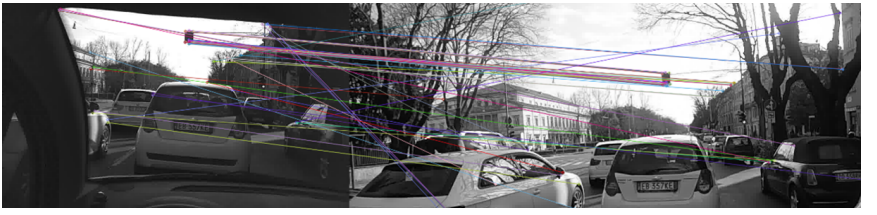


Figure 3.3: Registration between the egocentric and roof-mounted camera views by means of SIFT descriptor matching.

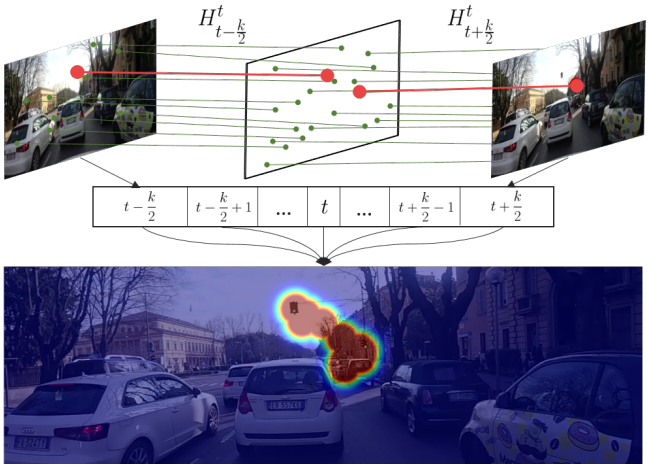


Figure 3.4: Resulting fixation map from a 1 second integration (25 frames). The adoption of the *max* aggregation of equation 3.1 allows to account in the final map two brief glances towards traffic lights.

t. For each time step $t+i$ in the window, where $i \in \{-\frac{k}{2}, -\frac{k}{2}+1, \ldots, \frac{k}{2}-1, \frac{k}{2}\}$, gaze points projections on F_t are estimated through the homography transformation H^t_{t+i} that projects points from the image plane at frame $t+i$, namely p_{t+i}, to the image plane in F_t. A continuous fixation map is obtained from the projected fixations by centering on each of them a multivariate Gaussian having a diagonal covariance matrix Σ (the spatial variance of each variable is set to $\sigma_s^2 = 200$ pixels) and taking the *max* value along the time axis:

$$F_t(x,y) = \max_{i \in (-\frac{k}{2}, \ldots, \frac{k}{2})} \mathcal{N}((x,y) \mid H^t_{t+i} \cdot p_{t+i}, \Sigma) \tag{3.1}$$

The Gaussian variance has been computed by averaging the ETG spatial acquisition errors on 20 observers looking at calibration patterns at different distances from 5 to 15 meters. The described process can be appreciated in Fig. 3.4. Eventually, each map F_t is normalized to sum to 1, so that it can be considered a probability distribution of fixation points.

3.1.5 Dataset structure

The following table (Tab.3.3) reports the design the DR(eye)VE dataset. The 74 sequences, of 5 minutes each, were recorded under a variety of driving conditions. Experimental design played a crucial role in preparing the dataset to rule out spurious correlation between driver, weather, traffic, daytime and scenario. Here we report the details for each sequence.

Table 3.3: Characteristics of each sequence in the DR(eye)VE dataset.

Sequence	Daytime	Weather	Landscape	Driver	Set
01	Evening	Sunny	Countryside	D8	Train Set
02	Morning	Cloudy	Highway	D2	Train Set
03	Evening	Sunny	Highway	D3	Train Set
04	Night	Sunny	Downtown	D2	Train Set
05	Morning	Cloudy	Countryside	D7	Train Set
06	Morning	Sunny	Downtown	D7	Train Set
07	Evening	Rainy	Downtown	D3	Train Set
08	Evening	Sunny	Countryside	D1	Train Set
09	Night	Sunny	Highway	D1	Train Set
10	Evening	Rainy	Downtown	D2	Train Set
11	Evening	Cloudy	Downtown	D5	Train Set
12	Evening	Rainy	Downtown	D1	Train Set
13	Night	Rainy	Downtown	D4	Train Set
14	Morning	Rainy	Highway	D6	Train Set
15	Evening	Sunny	Countryside	D5	Train Set
16	Night	Cloudy	Downtown	D7	Train Set
17	Evening	Rainy	Countryside	D4	Train Set
18	Night	Sunny	Downtown	D1	Train Set
19	Night	Sunny	Downtown	D6	Train Set
20	Evening	Sunny	Countryside	D2	Train Set
21	Night	Cloudy	Countryside	D3	Train Set
22	Morning	Rainy	Countryside	D7	Train Set
23	Morning	Sunny	Countryside	D5	Train Set
24	Night	Rainy	Countryside	D6	Train Set
25	Morning	Sunny	Highway	D4	Train Set
26	Morning	Rainy	Downtown	D5	Train Set
27	Evening	Rainy	Downtown	D6	Train Set
28	Night	Cloudy	Highway	D5	Train Set

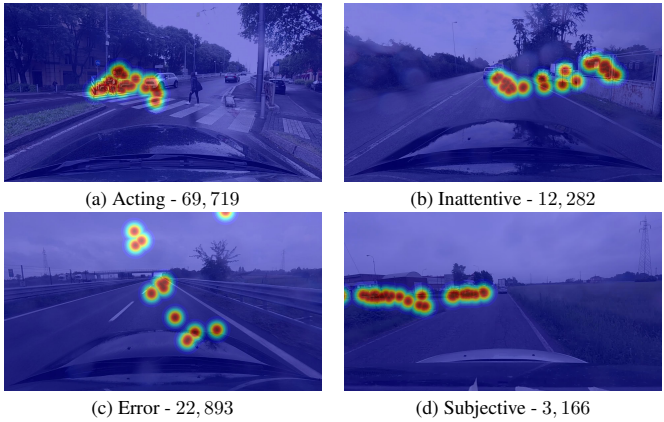
29	Night	Cloudy	Countryside	D8	Train Set
30	Evening	Cloudy	Highway	D7	Train Set
31	Morning	Rainy	Highway	D8	Train Set
32	Morning	Rainy	Highway	D1	Train Set
33	Evening	Cloudy	Highway	D4	Train Set
34	Morning	Sunny	Countryside	D3	Train Set
35	Morning	Cloudy	Downtown	D3	Train Set
36	Evening	Cloudy	Countryside	D1	Train Set
37	Morning	Rainy	Highway	D8	Train Set
38	Night	Sunny	Downtown	D8	Test Set
39	Night	Rainy	Downtown	D4	Test Set
40	Morning	Sunny	Downtown	D1	Test Set
41	Night	Sunny	Highway	D1	Test Set
42	Evening	Cloudy	Highway	D1	Test Set
43	Night	Cloudy	Countryside	D2	Test Set
44	Morning	Rainy	Countryside	D1	Test Set
45	Evening	Sunny	Countryside	D4	Test Set
46	Evening	Rainy	Countryside	D5	Test Set
47	Morning	Rainy	Downtown	D7	Test Set
48	Morning	Cloudy	Countryside	D8	Test Set
49	Morning	Cloudy	Highway	D3	Test Set
50	Morning	Rainy	Highway	D2	Test Set
51	Night	Sunny	Downtown	D3	Test Set
52	Evening	Sunny	Highway	D7	Test Set
53	Evening	Cloudy	Downtown	D7	Test Set
54	Night	Cloudy	Highway	D8	Test Set
55	Morning	Sunny	Countryside	D6	Test Set
56	Night	Rainy	Countryside	D6	Test Set
57	Evening	Sunny	Highway	D5	Test Set
58	Night	Cloudy	Downtown	D4	Test Set
59	Morning	Cloudy	Highway	D7	Test Set
60	Morning	Cloudy	Downtown	D5	Test Set
61	Night	Sunny	Downtown	D5	Test Set
62	Night	Cloudy	Countryside	D6	Test Set
63	Morning	Rainy	Countryside	D8	Test Set
64	Evening	Cloudy	Downtown	D8	Test Set
65	Morning	Sunny	Downtown	D2	Test Set
66	Evening	Sunny	Highway	D6	Test Set

67	Evening	Cloudy	Countryside	D3	Test Set
68	Morning	Cloudy	Countryside	D4	Test Set
69	Evening	Rainy	Highway	D2	Test Set
70	Morning	Rainy	Downtown	D3	Test Set
71	Night	Cloudy	Highway	D6	Test Set
72	Evening	Cloudy	Downtown	D2	Test Set
73	Night	Sunny	Countryside	D7	Test Set
74	Morning	Rainy	Downtown	D4	Test Set

3.1.6 Labeling attention drifts

Fixation maps exhibit a very strong central bias. This is common in saliency annotations [173] and even more in the context of driving. For these reasons, there is a strong unbalance between lots of easy-to-predict scenarios and unfrequent but interesting hard-to-predict events.
To enable the evaluation of computational models under such circumstances, the `DR(eye)VE` dataset has been extended with a set of further annotations. For each video, subsequences whose ground truth poorly correlates with the average ground truth of that sequence are selected. We employ Pearson's Correlation Coefficient (CC) and select subsequences with CC < 0.3. This happens when the attention of the driver focuses far from the vanishing point of the road. Examples of such subsequences are depicted in Fig. 3.5. Several human annotators inspected the selected frames and manually split them into (a) acting, (b) inattentive, (c) errors and (d) subjective events:

- *errors* can happen either due to failures in the measuring tool (*e.g.* in extreme lighting conditions) or in the successive data processing phase (*e.g.* SIFT matching);
- *inattentive* subsequences occur when the driver focuses his gaze on objects unrelated to the driving task (*e.g.* looking at an advertisement);
- *subjective* subsequences describe situations in which the attention is closely related to the individual experience of the driver, *e.g.* a road sign on the side might be an interesting element to focus for someone that has never been on that road before but might be safely ignored by someone who drives that road every day.
- *acting* subsequences include all the remaining ones.



(a) Acting - 69, 719 (b) Inattentive - 12, 282

(c) Error - 22, 893 (d) Subjective - 3, 166

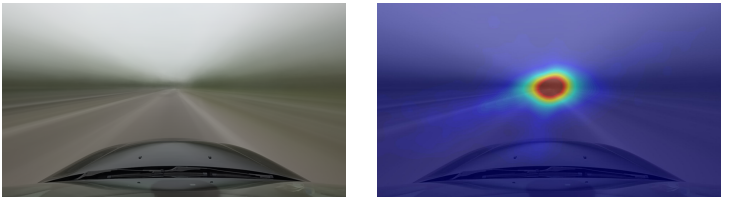
Figure 3.5: Examples of the categorization of frames where gaze is far from the mean. Overall, 108, 060 frames (∼20% of DR(eye)VE) were extended with this type of information.

Acting subsequences are particularly interesting as the deviation of driver's attention from the common central pattern denotes an intention linked to task-specific actions (*e.g.* turning, changing lanes, overtaking ...). For these reasons, subsequences of this kind will have a central role in the evaluation of predictive models in Sec. 3.4.

3.2 Dataset analysis

By analyzing the dataset frames, the very first insight is the presence of a strong attraction of driver's focus towards the vanishing point of the road, that can be appreciated in Fig. 3.6. The same phenomenon was observed in previous studies [184, 17] in the context of visual search tasks. We observed indeed that drivers often tend to disregard road signals, cars coming from the opposite direction and pedestrians on sidewalks. This is an effect of human peripheral vision [160], that allows observers to still perceive and interpret stimuli out of - but sufficiently close to - their focus of attention (FoA). A driver can therefore achieve a larger area of attention by focusing on the road's vanishing point: due to the geometry of the road environment, many of the objects worth of attention are coming from

there and have already been perceived when distant.
Moreover, the gaze location tends to drift from this central attractor when the context changes in terms of car speed and landscape. Indeed [153] suggests that our brain is able to compensate spatially or temporally dense information by reducing the visual field size. In particular, as the car travels at higher speed the temporal density of information (*i.e.* the amount of information that the driver needs to elaborate per unit of time) increases: this causes the useful visual field of the driver to shrink [153]. We also observe this phenomenon in our experiments, as shown in Fig. 3.7.
DR(eye)VE data also highlight that the driver's gaze is attracted towards specific semantic categories. To reach the above conclusion, the dataset is analysed by means of the semantic segmentation model in [203] and the distribution of semantic classes within the fixation map evaluated. More precisely, given a segmented frame and the corresponding fixation map, the probability for each semantic class to fall within the area of attention is computed as follows: First, the fixation map (which is continuous in $[0, 1]$) is normalized such that the maximum value equals 1. Then, nine binary maps are constructed by thresholding such continuous values linearly in the interval $[0, 1]$. As the threshold moves towards 1 (the maximum value), the area of interest shrinks around the real fixation points (since the continuous map is modeled by means of several Gaussians centered in fixation points, see previous section). For every threshold, a histogram over semantic labels within the area of interest is built, by summing up occurrences collected from all DR(eye)VE frames. Fig. 3.8 displays the result: for each class, the probability of a pixel to fall within the region of interest is reported for each threshold value. The figure provides insight about which categories represent the



(a) (b)

Figure 3.6: Mean frame (a) and fixation map (b) averaged across the whole sequence 02, highlighting the link between driver's focus and the vanishing point of the road.

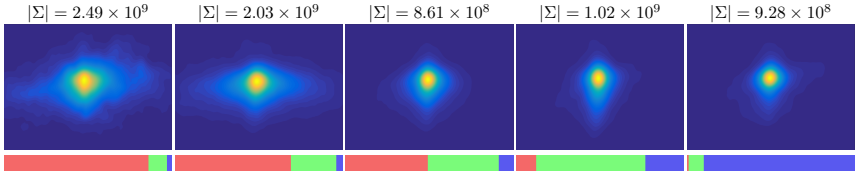


Figure 3.7: As speed gradually increases, driver's attention converges towards the vanishing point of the road. (a) When the car is approximately stationary, the driver is distracted by many objects in the scene. (b-e) As the speed increases, the driver's gaze deviates less and less from the vanishing point of the road. To measure this effect quantitatively, a two-dimensional Gaussian is fitted to approximate the mean map for each speed range, and the determinant of the covariance matrix Σ is reported as an indication of its spread (the determinant equals the product of eigenvalues, each of which measures the spread along a different data dimension). The bar plots illustrate the amount of downtown (red), countryside (green) and highway (blue) frames that concurred to generate the average gaze position for a specific speed range. Best viewed on screen.

real focus of attention and which ones tend to fall inside the attention region just by proximity with the formers. Object classes that exhibit a positive trend, such as road, vehicles and people, are the real focus of the gaze, since the ratio of pixels classified accordingly increases when the observed area shrinks around the fixation point. In a broader sense, the figure suggests that despite while driving our focus is dominated by road and vehicles, we often observe specific objects categories even if they contain little information useful to drive.

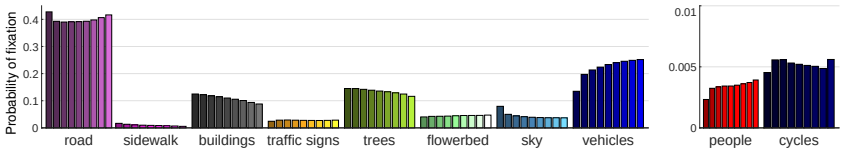


Figure 3.8: Proportion of semantic categories that fall within the driver's fixation map when thresholded at increasing values (from left to right). Categories exhibiting a positive trend (*e.g.* road and vehicles) suggest a real attention focus, while a negative trend advocates for an awareness of the object which is only circumstantial. See Sec. 3.2 for details.

3.3 Multi-Branch deep architecture for focus of attention prediction

The DR(eye)VE dataset is sufficiently large to allow the construction of a deep architecture to model common attentional patterns. Here, we describe our neural network model to predict human FoA while driving.

3.3.1 Architecture design

In the context of high level video analysis (*e.g.* action recognition and video classification), it has been shown that a method leveraging single frames can be outperformed if a sequence of frames is used as input instead [181, 77]. Temporal dependencies are usually modeled either by 3D convolutional layers [181], tailored to capture short range correlations, or by recurrent architectures (*e.g.* LSTM, GRU), that can model longer term dependencies [9, 136]. Our model follows the former approach, relying on the assumption that a small time window (*e.g.* half a second) holds sufficient contextual information for predicting where the driver would focus in that moment. Indeed, human drivers can take even less time to react to an unexpected stimulus. Our architecture takes a sequence of 16 consecutive frames ($\approx$ 0.65s) as input (called *clips* from now on) and predicts the fixation map for the last frame of such clip.
Many of the architectural choices made to design the network come from insights from the dataset analysis presented in Sec.3.2. In particular, we rely on the following results:

- the drivers' FoA exhibits consistent patterns, suggesting that it can be re-

produced by a computational model;

- the drivers' gaze is affected by a strong prior on objects semantics, *e.g.* drivers tend to focus on items lying on the road;
- motion cues, like vehicle speed, are also key factors that influence gaze.

Accordingly, the model output merges three branches with identical architecture, unshared parameters and different input domains: the RGB image, the semantic segmentation and the optical flow field. We call this architecture `multi-branch` model.
Following a bottom-up approach, in Sec. 3.3.2 the building blocks of each branch are motivated and described. Later, in Sec. 3.3.6 it will be shown how the branches merge into the final model.

3.3.2 Single FoA branch

Each branch of the `multi-branch` model is a two-input two-output architecture composed of two intertwined streams. The aim of this peculiar setup is to prevent the network from learning a central bias, that would otherwise stall the learning in early training stages [*]. To this end, one of the streams is given as input (output) a severely cropped portion of the original image (ground truth), ensuring a more uniform distribution of the true gaze, and runs through the `COARSE` module, described below. Similarly, the other stream uses the `COARSE` module to obtain a rough prediction over the full resized image and then refines it through a stack of additional convolutions called `REFINE` model. At test time, only the output of the `REFINE` stream is considered. Both streams rely on the `COARSE` module, the convolutional backbone (with shared weights) which provides the rough estimate of the attentional map corresponding to a given clip. This component is detailed in Fig. 3.9.
The `COARSE` module is based on the C3D architecture [181] that encodes video dynamics by applying a 3D convolutional kernel on the 4D input tensor. As opposed to 2D convolutions that stride along the width and height dimension of the input tensor, a 3D convolution also strides along time. Formally, the j-th feature map in the i-th layer at position (x, y) at time t is computed as:

$$v_{i,j}^{x,y,t} = b_{i,j} + \sum_{m} \sum_{p=0}^{P_{i-1}} \sum_{q=0}^{Q_{i-1}} \sum_{r=0}^{R_{i-1}} w_{i,j,m}^{p,q,r} v_{i-1,m}^{x+p,y+q,t+r} \tag{3.2}$$

[*] For further details the reader can refer to Sec. 3.4.5.5 and Sec. 3.4.5.6

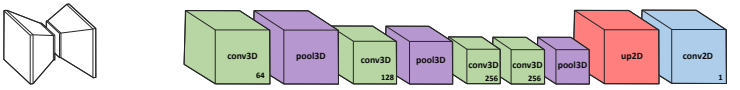


Figure 3.9: The COARSE module is made of an encoder based on C3D network [181] followed by a bilinear upsampling (bringing representations back to the resolution of the input image) and a final 2D convolution. During feature extraction, the temporal axis is lost due to 3D pooling. All convolutional layers are preceded by zero paddings in order keep borders, and all kernels have size 3 along all dimensions. Pooling layers have size and stride of (1, 2, 2, 4) and (2, 2, 2, 1) along temporal and spatial dimensions respectively. All activations are ReLUs.

where m indexes different input feature maps, $w_{i,j,m}^{p,q,r}$ is the value at the position (p, q) at time r of the kernel connected to the m-th feature map, and P_i, Q_i and R_i are the dimensions of the kernel along width, height and temporal axis respectively; $b_{i,j}$ is the bias from layer i to layer j.
From C3D, only the most general-purpose features are retained by removing the last convolutional layer and the fully connected layers which are strongly linked to the original action recognition task. The size of the last pooling layer is also modified in order to cover the remaining temporal dimension entirely. This collapses the tensor from 4D to 3D, making the output independent of time. Eventually, a bilinear upsampling brings the tensor back to the input spatial resolution and a 2D convolution merges all features into one channel. See Fig. 3.9 for additional details on the COARSE module.

3.3.3 Training the two streams together

The architecture of a single FoA branch is depicted in Fig. 3.10. During training, the first stream feeds the COARSE network with random crops, forcing the model to learn the current focus of attention given visual cues rather than prior spatial location. The C3D training process described in [181], employs a 128×128 image resize, and then a 112×112 random crop. However, the small difference in the two resolutions limits the variance of gaze position in ground truth fixation maps and is not sufficient to avoid the attraction towards the center of the image. For this reason, training images are resized to 256×256 before being cropped to 112×112. This crop policy generates samples that cover less than a quarter of the

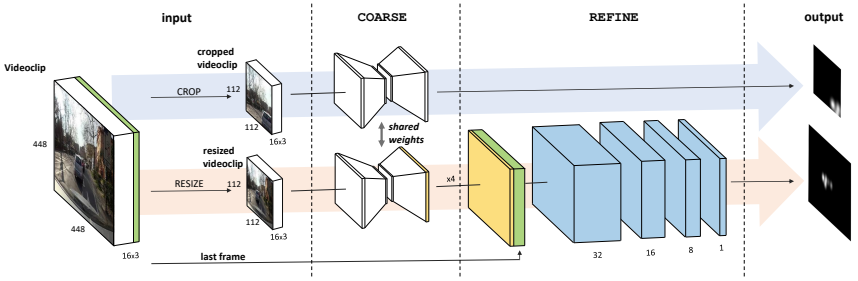


Figure 3.10: A single FoA branch of our prediction architecture. The COARSE module (see Fig. 3.9) is applied to both a cropped and a resized version of the input tensor, which is a videoclip of 16 consecutive frames. The cropped input is used during training to augment the data and the variety of ground truth fixation maps. The prediction of the resized input is stacked with the last frame of the videoclip and fed to a stack of convolutional layers (refinement module) with the aim of refining the prediction. Training is performed end-to-end and weights between COARSE modules are shared. At test time, only the refined predictions are used. Note that the complete model is composed of three of these branches (see Fig. 3.11), each of which predicting visual attention for different inputs (namely image, optical flow and semantic segmentation). All activations in the refinement module are LeakyReLU with $\alpha = 10^{-3}$, except for the last single channel convolution that features ReLUs. Crop and resize streams are highlighted by light blue and orange arrows respectively.

original image thus ensuring a sufficient variety in prediction targets. This comes at the cost of a coarser prediction: as crops get smaller, the ratio of pixels in the ground truth covered by gaze increases, leading the model to learn larger maps. In contrast, the second stream feeds the same COARSE model with the same images, this time *resized* to 112×112 – and not cropped. The coarse prediction obtained from the COARSE model is then concatenated with the final frame of the input clip, *i.e.* the frame corresponding to the final prediction. Eventually, the concatenated tensor goes through the REFINE module to obtain a higher resolution prediction of the FoA.

The overall two-stream training procedure for a single branch is summarized in Algorithm 1.

3.3.4 Training objective

Prediction cost can be minimized in terms of Kullback-Leibler divergence:

$$D_{KL}(Y\|\hat{Y}) = \sum_i Y(i)\ \log\left(\epsilon + \frac{Y(i)}{\epsilon + \hat{Y}(i)}\right) \tag{3.3}$$

where Y is the ground truth distribution, $\hat{Y}$ is the prediction, the summation index i spans across image pixels and ϵ is a small constant that ensures numerical stability[†]. Since each single FoA branch computes an error on both the cropped image stream and the resized image stream, the branch loss can be defined as:

$$\begin{aligned}\mathcal{L}_b(\mathcal{X}_b, \mathcal{Y}) = \sum_m \Big(& D_{KL}(\phi(Y^m)\|\mathcal{C}(\phi(X_b^m))) + \\ & D_{KL}(Y^m\|\mathcal{R}(\mathcal{C}(\psi(X_b^m)), X_b^m))\Big)\end{aligned} \tag{3.4}$$

where $\mathcal{C}$ and $\mathcal{R}$ denote `COARSE` and `REFINE` modules, $(X_b^m, Y^m) \in \mathcal{X}_b \times \mathcal{Y}$ is the m-th training example in the b-th domain (namely RGB, optical flow, semantic segmentation), and ϕ and ψ indicate the crop and the resize functions respectively.

3.3.5 Inference step

While the presence of the $\mathcal{C}(\phi(X_b^m))$ stream is beneficial in training to reduce the spatial bias, at test time only the $\mathcal{R}(\mathcal{C}(\psi(X_b^m)), X_b^m))$ stream producing higher quality prediction is used. The outputs of such stream from each branch b are then summed together, as explained in the following section.

3.3.6 Multi-Branch model

As described at the beginning of this section and depicted in Fig. 3.11, the `multi-branch` model is composed of three identical branches. The architecture of each branch has already been described in Sec. 3.3.2 above. Each branch exploits complementary information from a different domain and contributes to the final prediction accordingly. In detail, the first branch works in the RGB domain and

[†]Please note that D_{KL} inputs are always normalized to be a valid probability distribution despite this may be omitted in notation to improve equations readability.

Algorithm 1 TRAINING. The model is trained in two steps: first each branch is trained separately through iterations detailed in **procedure** SINGLE_BRANCH_TRAINING_ITERATION, then the three branches are fine-tuned altogether as shown by **procedure** MULTI_BRANCH_FINE-TUNING_ITERATION. For clarity, we omit from notation: i) the subscript b denoting the current domain in all X, x and $\hat{y}$ variables in the single branch iteration and ii) the normalization of the sum of the outputs from each branch in line 13.

1: **procedure A:** SINGLE_BRANCH_TRAINING_ITERATION
input: domain data $X = \{x_1, x_2, \ldots, x_{16}\}$, true attentional map y of last frame x_{16} of videoclip X
output: branch loss $\mathcal{L}_b$ computed on input sample (X, y)
2: $X_{\text{res}} \leftarrow \texttt{resize}(X, (112, 112))$
3: $X_{\text{crop}}, y_{\text{crop}} \leftarrow \texttt{get\_crop}((X, y), (112, 112))$
get coarse prediction on uncentered crop
4: $\hat{y}_{\text{crop}} \leftarrow \texttt{COARSE}(X_{\text{crop}})$
get refined prediction over whole image
5: $\hat{y} \leftarrow \texttt{REFINE}(\texttt{stack}(x_{16}, \texttt{upsample}(\texttt{COARSE}(X_{\text{res}}))))$
compute branch loss as in Eq. 3.4
6: $\mathcal{L}_b(X, Y) \leftarrow D_{KL}(y_{\text{crop}} \| \hat{y}_{\text{crop}}) + D_{KL}(y \| \hat{y})$

7: **procedure B:** MULTI_BRANCH_FINE-TUNING_ITERATION
input: data $X = \{x_1, x_2, \ldots, x_{16}\}$ for all domains, true attentional map y of last frame x_{16} of videoclip X
output: overall loss $\mathcal{L}$ computed on input sample (X, y)
8: $X_{\text{res}} \leftarrow \texttt{resize}(X, (112, 112))$
9: $X_{\text{crop}},\ y_{\text{crop}} \leftarrow \texttt{get\_crop}((X,\ y), (112, 112))$
10: **for** branch $b \in \{\text{RGB}, \text{flow}, \text{seg}\}$ **do**
as in line 4 of the above procedure
11: $\hat{y}_{b_{\text{crop}}} \leftarrow \texttt{COARSE}(X_{b_{\text{crop}}})$
as in line 5 of the above procedure
12: $\hat{y}_b \leftarrow \texttt{REFINE}(\texttt{stack}(x_{b_{16}}, \texttt{upsample}(\texttt{COARSE}(X_{b_{\text{res}}}))))$
compute overall loss as in Eq. 3.5
13: $\mathcal{L}(X, Y) \leftarrow D_{KL}(y_{\text{crop}} \| \sum_b \hat{y}_{b_{\text{crop}}}) + D_{KL}(y \| \sum_b \hat{y}_b)$

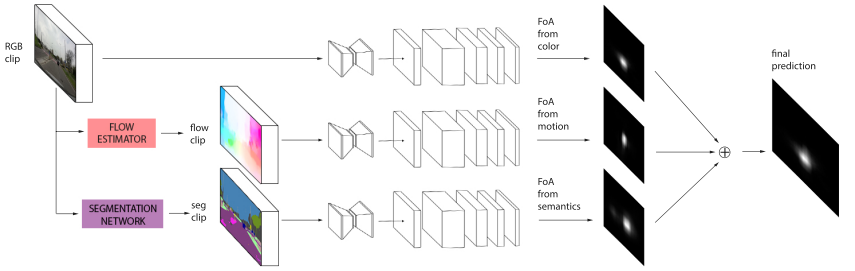


Figure 3.11: The `multi-branch` model is composed of three different branches, each of which has its own set of parameters, and their predictions are summed to obtain the final map. Note that in this figure cropped streams are dropped to ease representation, but are employed during training (as discussed in Sec. 3.3.6 and depicted in Fig. 3.10.

processes raw visual data about the scene X_{RGB}. The second branch focuses on motion through the optical flow representation X_{flow} described in [54]. Eventually, the last branch takes as input semantic segmentation probability maps X_{seg}. For this last branch, the number of input channels depends on the specific algorithm used to extract the results, 19 in our setup (Yu and Koltun [203]). The three independent predicted FoA maps are summed and normalized to result in a probability distribution.
To allow for larger batch size, we choose to bootstrap each branch independently by training it according to Eq. 3.4. Then, the complete `multi-branch` model which merges the three branches is fine-tuned with the following loss:

$$\mathcal{L}(\mathcal{X},\mathcal{Y}) = \sum_m \Bigg(D_{KL}(\phi(Y^m) \| \sum_b \mathcal{C}(\phi(X_b^m))) + D_{KL}(Y^m \| \sum_b \mathcal{R}(\mathcal{C}(\psi(X_b^m)), X_b^m))) \Bigg). \tag{3.5}$$

The algorithm describing the complete inference over the `multi-branch` model in detailed in Alg. 2.

Algorithm 2 INFERENCE. At test time, the data extracted from the resized videoclip is input to the three branches and their output is summed and normalized to obtain the final FoA prediction.

input: data $X = \{x_1, x_2, \ldots, x_{16}\}$ for all domains
output: predicted FoA map $\hat{y}$
1: $X_{\text{res}} \leftarrow \texttt{resize}(X, (112, 112))$
2: **for** branch $b \in \{\text{RGB}, \text{flow}, \text{seg}\}$ **do**
3: $\quad \hat{y}_b \leftarrow \texttt{REFINE}(\texttt{stack}(x_{b_{16}}, \texttt{upsample}(\texttt{COARSE}(X_{b_{\text{res}}}))))$
4: $\hat{y} \leftarrow \sum_b \hat{y}_b / \sum_i \sum_b \hat{y}_b(i)$

3.4 Experiments

In this section we evaluate the performance of the proposed `multi-branch` model. First, we start by comparing our model against some baselines and other methods in literature. Following the guidelines in [24], for the evaluation phase we rely on Pearson's Correlation Coefficient (CC) and Kullback–Leibler Divergence (D_{KL}) measures. Moreover, we evaluate the Information Gain (IG) [94] measure to assess the quality of a predicted map P with respect to a ground truth map Y in presence of a strong bias, as:

$$IG(P, Y, B) = \frac{1}{N} \sum_i Y_i [(\log_2(\epsilon + P_i) - \log_2(\epsilon + B_i)] \tag{3.6}$$

where i is an index spanning all the N pixels in the image, B the bias computed as the average training fixation map and ϵ ensures numerical stability.
Furthermore, we conduct an ablation study to investigate how different branches affect the final prediction and how their mutual influence changes in different scenarios. We then study whether our model captures the attention dynamics observed in Sec. 3.2. Eventually, we assess our model from a human perception perspective.

3.4.1 Implementation details

The three different pathways of the `multi-branch` model (namely FoA from color, from motion and from semantics) have been pre-trained independently using the same cropping policy of Sec. 3.3.6 and minimizing the objective function in Eq. 3.4. Each branch has been respectively fed with:

- 16 frames clips in raw RGB color space;
- 16 frames clips with optical flow maps, encoded as color images through the flow field encoding [54];
- 16 frames clips holding semantic segmentation from [203] encoded as 19 scalar activation maps, one per segmentation class.

During individual branch pre-training clips were randomly mirrored for data augmentation. We employ Adam optimizer with parameters as suggested in the original paper [81], with the exception of the learning rate that we set to 10^{-4}.

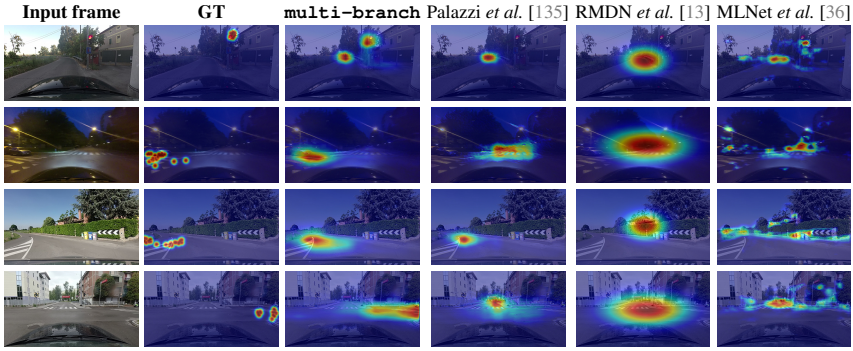


Figure 3.12: Qualitative assessment of the predicted fixation maps. From left to right: input clip, ground truth map, our prediction, prediction of the previous version of the model [135], prediction of RMDN [13] and prediction of MLNet [36].

Eventually, batch size was fixed to 32 and each branch was trained until convergence. The DR(eye)VE dataset is split into train, validation and test set as follows: sequences 1-38 are used for training, sequences 39-74 for testing. The 500 frames in the middle of each training sequence constitute the validation set. Moreover, the complete multi-branch architecture was fine-tuned using the same cropping and data augmentation strategies minimizing cost function in Eq. 3.5. In this phase batch size was set to 4 due to GPU memory constraints and learning rate value was lowered to 10^{-5}. Inference time of each branch of our architecture is $\approx$ 30 milliseconds per videoclip on an NVIDIA Titan X.

3.4.2 Model evaluation

In Tab. 3.4 we report results of our proposal against other state-of-the-art models [196, 171, 36, 135, 13, 197] evaluated both on the complete test set and on *acting* subsequences only. All the competitors, with the exception of [135] are bottom-up approaches and mainly rely on appearance and motion discontinuities. To test the effectiveness of deep architectures for saliency prediction we compare against the Multi-Level Network (MLNet) [36], which scored favourably in the MIT300 saliency benchmark [23], and the Recurrent Mixture Density Network (RMDN) [13], which represents the only deep model addressing video saliency. While MLNet works on images discarding the temporal information, RMDN en-

Table 3.4: Experiments illustrating the superior performance of the `multi-branch` model over several baselines and competitors. We report both the average across the complete test sequences and only the *acting* frames.

	Test sequences			Acting subsequences		
	$CC \uparrow$	$D_{KL} \downarrow$	$IG \uparrow$	$CC \uparrow$	$D_{KL} \downarrow$	$IG \uparrow$
Baseline Gaussian	0.40	2.16	-0.49	0.26	2.41	0.03
Baseline Mean	0.51	1.60	0.00	0.22	2.35	0.00
Mathe *et al.*[171]	0.04	3.30	-2.08	-	-	-
Wang *et al.*[196]	0.04	3.40	-2.21	-	-	-
Wang *et al.*[197]	0.11	3.06	-1.72	-	-	-
MLNet[36]	0.44	2.00	-0.88	0.32	2.35	-0.36
RMDN[13]	0.41	1.77	-0.06	0.31	2.13	0.31
Palazzi *et al.*[135]	0.55	1.48	-0.21	0.37	2.00	0.20
`multi-branch`	**0.56**	**1.40**	**0.04**	**0.41**	**1.80**	**0.51**

codes short sequences in a similar way to our `COARSE` module, and then relies on a LSTM architecture to model long term dependencies and estimates the fixation map in terms of a GMM. To favor the comparison, both models were re-trained on the `DR(eye)VE` dataset.

Results highlight the superiority of our `multi-branch` architecture on all test sequences. The gap in performance with respect to bottom-up unsupervised approaches [196, 197] is higher, and is motivated by the peculiarity of the attention behavior within the driving context, which calls for a task-oriented training procedure. Moreover, MLNet's low performance testifies for the need of accounting for the temporal correlation between consecutive frames that distinguishes the tasks of attention prediction in images and videos. Indeed, RMDN processes video inputs and outperforms MLNet on both D_{KL} and IG metrics, performing comparably on CC. Nonetheless, its performance is still limited: indeed, qualitative results reported in Fig. 3.12 suggest that long term dependencies captured by its recurrent module lead the network towards the regression of the mean, discarding contextual and frame-specific variations that would be preferrable to keep. To support this intuition, we measure the average D_{KL} between RMDN predictions and the mean training fixation map (Baseline Mean), resulting in a value of 0.11. Being lower than the divergence measured with respect to

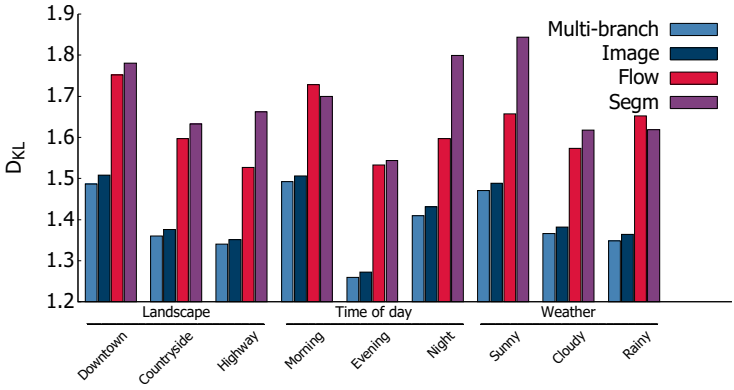


Figure 3.13: D_{KL} of the different branches in several conditions (from left to right: downtown, countryside, highway, morning, evening, night, sunny, cloudy, rainy). Underlining highlights difference of aggregation in terms of landscape, time of day and weather. Please note that lower D_{KL} indicates better predictions.

groundtruth maps, this value highlights the closer correlation to a central baseline rather than to groundtruth. Eventually, we also observe improvements with respect to our previous proposal [135], that relies on a more complex backbone model (also including a deconvolutional module) and processes RGB clips only. The gap in performance resides in the greater awareness of our `multi-branch` architecture of the aspects that characterize the driving task as emerged from the analysis in Sec. 3.2. The positive performances of our model are also confirmed when evaluated on the *acting* partition of the dataset. We recall that *acting* indicates sub-sequences exhibiting a significant task-driven shift of attention from the center of the image (Fig. 3.5). Being able to predict the FoA also on *acting* sub-sequences means that the model captures the strong centered attention bias but is capable of generalizing when required by the context.

This is further shown by the comparison against a centered Gaussian baseline (BG) and against the average of all training set fixation maps (BM). The former baseline has proven effective on many image saliency detection tasks [23] while the latter represents a more task-driven version. The superior performance of the `multi-branch` model w.r.t. baselines highlights that despite the attention is often strongly biased towards the vanishing point of the road, the network is able to deal with sudden task-driven changes in gaze direction.

3.4.3 Model analysis

In this section we investigate the behavior of our proposed model under different landscapes, time of day and weather (Sec. 3.4.3.1); we study the contribution of each branch to the FoA prediction task (Sec. 3.4.3.2); and we compare the learnt attention dynamics against the one observed in the human data (Sec. 3.4.3.3).

3.4.3.1 Dependency on driving environment

The DR(eye)VE data has been recorded under varying landscapes, time of day and weather conditions. We tested our model in all such different driving conditions. As would be expected, Fig. 3.13 shows that the human attention is easier to predict in highways rather than downtown, where the focus can shift towards more distractors. The model seems more reliable in evening scenarios, rather than morning or night, where we observed better lightning conditions and lack of shadows, over-exposure and so on. Lastly, in rainy conditions we notice that human gaze is easier to model, possibly due to the higher level of awareness demanded to the driver and his consequent inability to focus away from vanishing point. To support the latter intuition, we measured the performance of BM baseline (*i.e.* the average training fixation map), grouped for weather condition. As expected, the D_{KL} value in rainy weather (1.53) is significantly lower than the ones for cloudy (1.61) and sunny weather (1.75), highlighting that when rainy the driver is more focused on the road.

3.4.3.2 Ablation study

In order to validate the design of the multi-branch model (see Sec. 3.3.6), here we study the individual contributions of the different branches by disabling one or more of them.
Results in Tab. 3.5 show that the RGB branch plays a major role in FoA prediction. The motion stream is also beneficial and provides a slight improvement, that becomes clearer in the *acting* subsequences. Indeed, optical flow intrinsically captures a variety of peculiar scenarios that are non-trivial to classify when only color information is provided, *e.g.* when the car is still at a traffic light or is turning. The semantic stream, on the other hand, provides very little improvement. In particular, from Tab. 3.5 and by specifically comparing I+F and I+F+S, a slight increase in the IG measure can be appreciated. Nevertheless, such improvement has to be considered negligible when compared to color and motion,

Table 3.5: The ablation study performed on our `multi-branch` model. I, F and S represent image, optical flow and semantic segmentation branches respectively.

	Test sequences			Acting subsequences		
	$CC \uparrow$	$D_{KL} \downarrow$	$IG \uparrow$	$CC \uparrow$	$D_{KL} \downarrow$	$IG \uparrow$
I	0.554	1.415	-0.008	0.403	1.826	0.458
F	0.516	1.616	-0.137	0.368	2.010	0.349
S	0.479	1.699	-0.119	0.344	2.082	0.288
I+F	0.558	1.399	0.033	**0.410**	1.799	0.510
I+S	0.554	1.413	-0.001	0.404	1.823	0.466
F+S	0.528	1.571	-0.055	0.380	1.956	0.427
I+F+S	**0.559**	**1.398**	**0.038**	**0.410**	**1.797**	**0.515**

suggesting that in presence of efficiency concerns or real-time constraints the semantic stream can be discarded with little losses in performance. However, we expect the benefit from this branch to increase as more accurate segmentation models will be released.
In Fig.3.14 we showcase several examples depicting the contribution of each branch of the `multi-branch` model in predicting the visual focus of attention of the driver. As expected, the RGB branch is the one that more heavily influences the overall network output.

3.4.3.3 Do we capture the attention dynamics?

The previous sections validate quantitatively the proposed model. Now, we assess its capability to attend like a human driver by comparing its predictions against the analysis performed in Sec. 3.2.
First, we report the average predicted fixation map in several speed ranges in Fig. 3.15. The conclusions we draw are twofold: i) generally, the model succeeds in modeling the behavior of the driver at different speeds, and ii) as the speed increases fixation maps exhibit lower variance, easing the modeling task, and prediction errors decrease.
We also study how often our model focuses on different semantic categories, in a fashion that recalls the analysis of Sec. 3.2, but employing our predictions rather than ground truth maps as focus of attention. More precisely, we normalize each map so that the maximum value equals 1, and apply the same thresholding

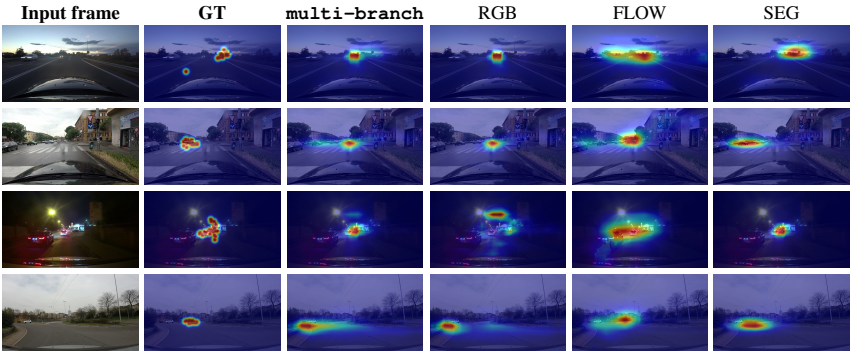
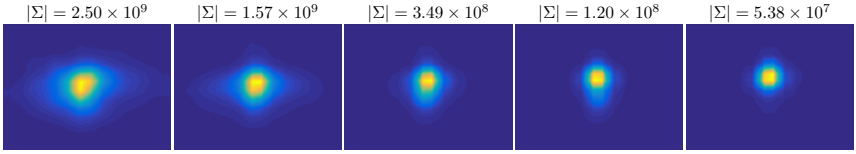


Figure 3.14: Example cases that qualitatively show how each branch contribute to the final prediction. Best viewed on screen.

strategy described in Sec. 3.2. Likewise, for each threshold value a histogram over class labels is built, by accounting all pixels falling within the binary map for all test frames. This results in nine histograms over semantic labels, that we merge together by averaging probabilities belonging to different threshold. Fig. 3.16 shows the comparison. Color bars represent how often the predicted map focuses on a certain category, while gray bars depict ground truth behavior and are obtained by averaging histograms in Fig. 3.8 across different thresholds. Please note that, to highlight differences for low populated categories, values are reported on a logarithmic scale. The plot shows a certain degree of absolute error is present for all categories. However, in a broader sense, our model replicates the relative weight of different semantic classes while driving, as testified by the importance of roads and vehicles, that still dominate, against other categories such as people and cycles that are mostly neglected. This correlation is confirmed by Kendall rank coefficient, which scored 0.51 when computed on the two bar series.

3.4.4 Visual assessment of predicted fixation maps

To further validate the predictions of our model from the human perception perspective, 50 people with at least 3 years of driving experience were asked to par-



(a) $0 \leq$ km/h ≤ 10 (b) $10 \leq$ km/h ≤ 30 (c) $30 \leq$ km/h ≤ 50 (d) $50 \leq$ km/h ≤ 70 (e) $70 \leq$ km/h

Figure 3.15: Model prediction averaged across all test sequences and grouped by driving speed. As the speed increases, the area of the predicted map shrinks, recalling the trend observed in ground truth maps. As in Fig. 3.7, for each map a two dimensional Gaussian is fitted and the determinant of its covariance matrix Σ is reported as a measure of the spread.

ticipate in a visual assessment[‡]. First, a pool of 400 videoclips (40 seconds long) is sampled from the DR(eye)VE dataset. Sampling is weighted such that resulting videoclips are evenly distributed among different scenarios, weathers, drivers and daylight conditions. Also, half of these videoclips contain sub-sequences that were previously annotated as *acting*.

3.4.4.1 Space Variant Imaging System

To approximate as realistically as possible the visual field of attention of the driver, sampled videoclips are pre-processed following the procedure in [195]. As in [195] we leverage the *Space Variant Imaging Toolbox* [139] to implement this phase, setting the parameter that halves the spatial resolution every 2.3° to mirror human vision [195, 97].
The Space Variant Imaging System (SVIS) is a MATLAB toolbox that allows to foveate images in real-time [139], which has been used in a large number of scientific works to approximate human foveal vision since its introduction in 2002. In this frame, the term *foveated imaging* refers to the creation and display of static or video imagery where the resolution varies across the image. In analogy to human foveal vision, the highest resolution region is called the foveation region. In a video, the location of the foveation region can obviously change dynamically. It is also possible to have more than one foveation region in each image.
The foveation process is implemented in the SVIS toolbox as follows: first the the input image is repeatedly low-passed filtered and down-sampled to half of the

[‡]These were students (11 females, 39 males) of age between 21 and 26 ($\mu = 23.4, \sigma = 1.6$) recruited at our University on a voluntary basis through an online form.

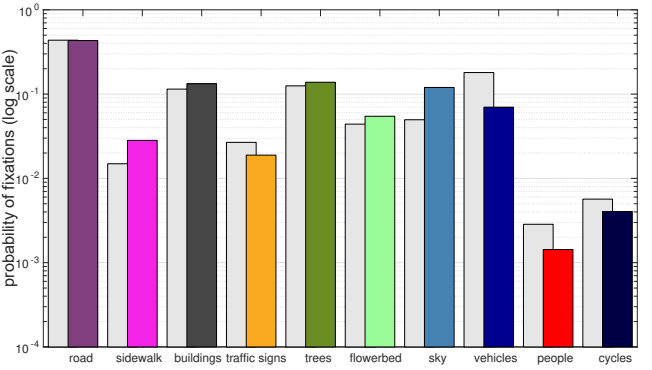


Figure 3.16: Comparison between ground truth (gray bars) and predicted fixation maps (colored bars) when used to mask semantic segmentation of the scene. The probability of fixation (in log-scale) for both ground truth and model prediction is reported for each semantic class. Despite absolute errors exist, the two bar series agree on the relative importance of different categories.

current resolution by a *Foveation Encoder*. In this way a low-pass pyramid of images is obtained. Then a foveation pyramid is created selecting regions from different resolutions proportionally to the distance from the foveation point. Concretely, the foveation region will be at the highest resolution; first ring around the foveation region will be taken from half-resolution image; and so on. Eventually, a *Foveation Decoder* up-sample, interpolate and blend each layer in the foveation pyramid to create the output foveated image.
The software is open-source and publicly available here: `http://svi.cps.utexas.edu/software.shtml`. The interested reader is referred to the SVIS website for further details.
The resulting `DR(eye)VE` videoclips preserve details near to the fixation points in each frame, whereas the rest of the scene gets more and more blurred getting farther from fixations until only low-frequency contextual information survive. Coherently with [195] we refer to this process as *foveation* (in analogy with human foveal vision). Thus, pre-processed videoclips will be called *foveated videoclips* from now on. To appreciate the effect of this step the reader is referred to Fig. 3.17.
Foveated videoclips were created by randomly selecting one of the following three fixation maps as guidance: the ground truth fixation map (G videoclips),

Figure 3.17: The figure depicts a videoclip frame that underwent the foveation process. The attentional map (above) is employed to blur the frame in a way that approximates the foveal vision of the driver [139]. In the foveated frame (below), it can be appreciated how the ratio of high-level information smoothly degrades getting farther from fixation points.

the fixation map predicted by our model (P videoclips) or the average fixation map in the DR(eye)VE training set (C videoclips). The latter central baseline allows to take into account the potential preference for a "stable" attentional map (*i.e.* lack of switching of focus).

3.4.4.2 From fixation maps back to fixations

The SVIS toolbox allows to foveate images starting from a list of (x, y) coordinates which represent the foveation points in the given image. However, we do not have this information as in our work we deal with continuous attentional maps rather than discrete points of fixations. To be able to use the same software API we need to regress from the attentional map (either true or predicted) a list of approximated yet plausible fixation locations. To this aim we simply extract the 25 points with highest value in the attentional map. This is justified by the fact that in the phase of dataset creation the ground truth *fixation map* F_t for a frame at time t is built by accumulating projected gaze points in a temporal sliding win-

dow of $k = 25$ frames, centered in t (see Sec. 3.3). The output of this phase is thus a list of 25 fixation coordinates we can use as input for the SVIS toolbox.

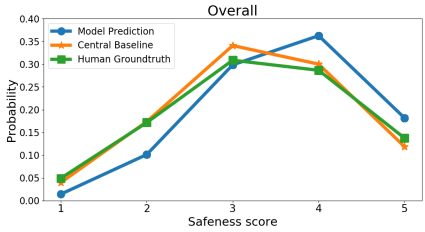
3.4.4.3 Perceived safety assessment

Each participant was asked to watch five randomly sampled foveated videoclips. After each videoclip, he answered the following questions:

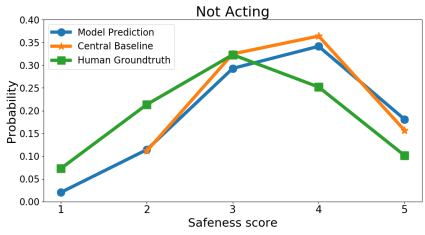
Q1 If you were sitting in the same car of the driver whose attention behavior you just observed, how safe would you feel? (rate from 1 to 5)

Q2 Would you say the observed attention behavior comes from a human driver? (yes/no)

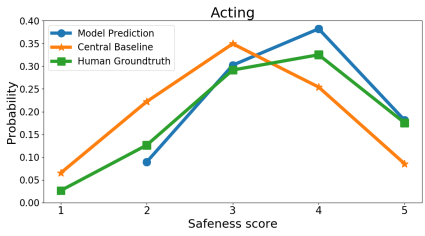
The aim of the question Q1 is to measure the comfort level of the observer during a driving experience when suggested to focus at specific locations in the scene. The underlying assumption is that the observer is more likely to feel safe if he agrees that the suggested focus is lighting up the right portion of the scene, that is what he thinks it is worth looking in the current driving scene. Conversely, if the observer wishes to focus at some specific location but he cannot retrieve details there, he is going to feel uncomfortable. On the contrary, Q2 helps understanding how easy is, in general, to discriminate between the attentive behavior of humans and the one of the proposed model. Each of the 50 participant evaluates five foveated videoclips, for a total of 250 examples.
The answers to Q1 provided by subjects, summarized in Fig. 3.18, indicate that perceived safety for videoclips foveated using the attentional maps predicted by the model is generally higher than for the ones foveated using either human or central baseline maps. Nonetheless the central bias baseline proves to be extremely competitive, in particular in non-acting videoclips in which it scores similarly to the model prediction. It is worth noticing that in this latter case both kind of automatic predictions outperform human ground truth by a significant margin (Fig. 3.18b). Conversely, when we consider only the foveated videoclips containing acting subsequences, the human ground truth is perceived as much safer than the baseline, despite still scores worse than our model prediction (Fig. 3.18c). These results hold despite due to the localization of the fixations the average resolution of the predicted maps is only the 38% of the resolution of ground truth maps (*i.e.* videos foveated using prediction map feature much less information). We did not measure significant difference in perceived safety across the different drivers in the dataset ($\sigma^2 = 0.09$).



(a)



(b)



(c)

Figure 3.18: Distributions of safeness scores for different map sources, namely Model Prediction, Center Baseline and Human Groundtruth. Considering the score distribution over all foveated videoclips (a) the three distributions may look similar, even though the model prediction still scores slightly better. However, when considering only the foveated videos contaning acting subsequences (b) the model prediction significantly outperforms both center baseline and human groundtruth. Conversely, when the videoclips did not contain acting subsequences (*i.e.* the car was mostly going straight) the fixation map from human driver is the one perceived as less safe, while both model prediction and center baseline perform similarly.

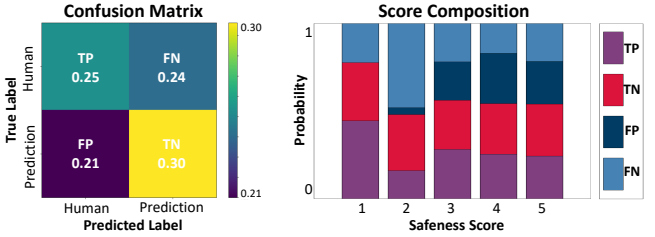


Figure 3.19: (a) The confusion matrix reports the results of participants' guesses on the source of fixation maps. Overall accuracy is about 55% which is fairly close to random chance. (b) The stacked bar graph represents the ratio of TP, TN, FP and FN composing each score. The increasing score of FP – participants falsely thought the attentional map came from a human driver – highlights that participants were tricked into believing that "safer" clips came from humans.

Regarding Q2, the confusion matrix of provided answers is reported in Fig. 3.19 (a). Participants were not particularly good at discriminating between human's gaze and model generated maps, scoring about the 55% of accuracy which is comparable to random guessing; this suggests our model is capable of producing plausible attentional patterns that resemble a proper driving behavior to a human observer. We report in Fig 3.19 (b) the composition of each score from Q1 in terms of answers to Q2. This analysis aims to measure participants' bias towards human driving ability. Indeed, increasing trend of false positives towards higher scores suggests that participants were tricked into believing that "safer" clips came from humans.

3.4.5 Supplementary experimentation

3.4.5.1 Failure cases

In Fig. 3.20 we report several clips in which our architecture fails to capture the groundtruth human attention.

3.4.5.2 Segmentation

In this section we report exemplar cases that particularly benefit from the segmentation branch. In Fig. 3.21 we can appreciate that, among the three branches,

Figure 3.20: Some failure cases of our `multi-branch` architecture.

only the semantic one captures the real gaze, that is focused on traffic lights and street signs.

3.4.5.3 Do subtasks help in FoA prediction?

The driving task is inherently composed of many subtasks, such as turning or merging in traffic, looking for parking and so on. While such fine-grained subtasks are hard to discover (and probably to emerge during learning) due to scarcity, here we show how the proposed model has been able to leverage on more common subtask to get to the final prediction. These subtasks are: turning left/right,

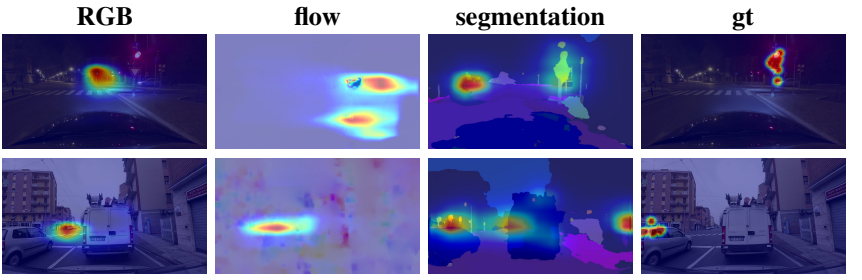


Figure 3.21: Some examples of the beneficial effect of the semantic segmentation branch. In the two cases depicted here, the car is stopped at a crossroad. While the RGB branch remains biased towards the road vanishing point and the optical flow branch focuses on moving objects, the semantic branch tends to highlight traffic lights and signals, coherently with the human behavior.

going straight, being still. We gathered automatic annotation through GPS information released with the dataset. We then train a linear SVM classifier to distinguish the above 4 different actions starting from the activations of the last layer of `multi-path` model, unrolled in a feature vector. The SVM classifier scores a 90% of accuracy on the test set (5000 uniformely sampled videoclips), supporting the fact that network activations are highly discriminative for distinguishing the different driving subtasks. Please refer to Fig. 3.22 for further details.

3.4.5.4 Error analysis for non-planar homographic projection

In Sec. 3.1.3 and Sec. 3.1.4, we take advantage of homography estimation in order to project drivers' fixation points across different views and different frames of the same view. An homography H is a projective transformation from a plane A to another plane B such that the collinearity property is preserved during the mapping. In our use-case, as well as in most real world applications, the homography matrix H is computed through an overdetermined set of image coordinates lying on the same implicit plane, aligning points on the plane in one image with points on the plane in the other image. To this end, we efficiently recover H through least square projection minimization, assuming the input set of points is approximately lying on the true implicit plane.
Once H has been approximated from data, we map an image point $\mathbf{x}$ from the first image to the respective point $H\mathbf{x}$ in the second image. Here, again, the

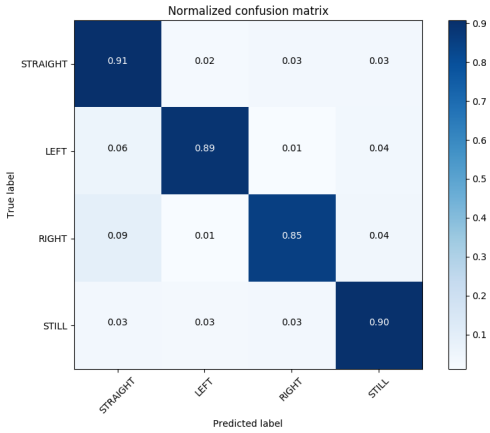
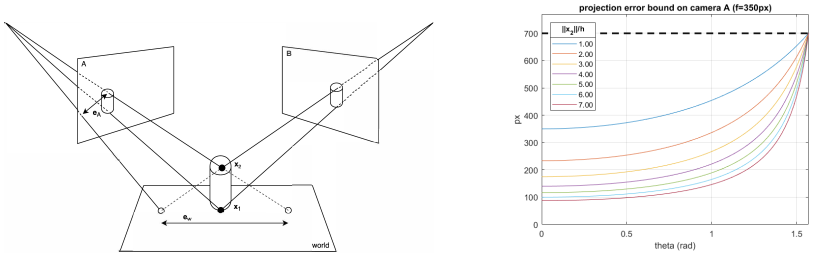


Figure 3.22: Confusion matrix for SVM classifier trained to distinguish driving actions from network activations. The accuracy is generally high, which corroborates the assumption that the model benefits from learning an internal representation of the different driving sub-tasks.

basic assumption is that $\mathbf{x}$ actually lies on the implicit plane. In practice this assumption is widely violated in real world applications, when the process of mapping is automated and the content of the mapping is not known a-priori. In what follows, we derive an error analysis justifying the suitability of homographic projections for our purpose.

The geometry of the problem In Fig.3.23 we show the generic setting of two cameras capturing the same 3D plane. To construct an erroneous case study, we put a cylinder on top of the plane. Points on the implicit 3D world plane can be consistently mapped across views with an homography transformation and retain their original semantic. As an example, the point $\mathbf{x_1}$ is the center of the cylinder base both in world coordinates and across different views. Conversely, the point $\mathbf{x_2}$ on the top of the cylinder cannot be consistently mapped from one view to the other. To see why, suppose we want to map $\mathbf{x_2}$ from view B to view A. Since the homography assumes $\mathbf{x_2}$ to also be on the implicit plane, its inferred 3D position is far from the true top of the cylinder and is depicted with the leftmost empty



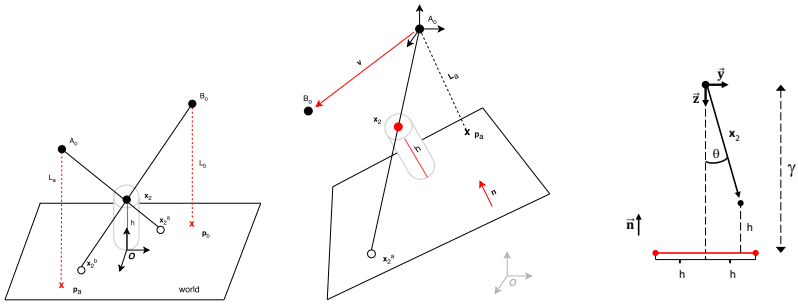
(a) (b)

Figure 3.23: (a) Two image planes capture a 3D scene from different viewpoints and (b) a use case of the bound derived below.

circle in fig:dreyeve.3.23. When this point gets reprojected to view A, its image coordinates are unaligned with the correct position of the cylinder top in that image. We call this offset the *reprojection error* on plane A, or $\mathbf{e_A}$. Analogously, a reprojection error on plane B could be computed with an homographic projection of point $\mathbf{x_2}$ from view A to view B.

The reprojection error is useful to measure the perceptual misalignment of projected points with their intended locations, but due to the (re)projections involved is not an easy tool to work with. Moreover, the very same point can produce different reprojection errors when measured on A and on B. A related error also arising in this setting is the *metric error* $\mathbf{e_W}$, or the displacement in world space of the projected image points at the intersection with the implicit plane. This measure of error is of particular interest because it is view-independent, does not depend on the rotation of the cameras with respect to the plane and is zero if and only if the reprojection error is also zero.

Computing the metric error Since the metric error does not depend on the mutual rotation of the plane with the camera views, we can simplify Fig.3.23 by retaining only the optical centers A and B from all cameras and by setting, without loss of generality, the reference system on the projection of the 3D point on the plane. This second step is useful to factor out the rotation of the world plane, which is unknown in the general setting. The only assumption we make is that the non-planar point $\mathbf{x_2}$ can be seen from both camera views. This simplification is depicted in Fig.3.24(a), where we have also named several important



(a) (b) (c)

Figure 3.24: (a) By aligning the reference system with the plane normal, centered on the projection of the non-planar point onto the plane, the metric error is the magnitude of the difference between the two vectors $\vec{\mathbf{x}}_2^a$ and $\vec{\mathbf{x}}_2^b$. The red lines help to highlight the similarity of inner and outer triangles having $\mathbf{x}_x^a$ as a vertex. (b) The geometry of the problem when the reference system is placed off the plane in an arbitrary position (gray) or, specifically, on one of the camera (black). (c) The simplified setting in which we consider the projection of the metric error $\|\mathbf{e}_w\|$ on the camera plane of A.

quantities such as the distance h of $\mathbf{x}_2$ from the plane.
In Fig. 3.24(a), the metric error can be computed as the magnitude of the difference between the two vectors relating points $\mathbf{x}_2^a$ and $\mathbf{x}_2^b$ to the origin:

$$\mathbf{e}_w = \mathbf{x}_2^a - \mathbf{x}_2^b. \tag{3.7}$$

The aforementioned points are at the intersection of the lines connecting the optical center of the cameras with the 3D point $\mathbf{x}_2$ and the implicit plane. An easy way to get such points is through their magnitude and orientation. As an example, consider the point $\mathbf{x}_2^a$. Starting from $\mathbf{x}_2^a$ the following two similar triangles can be built:

$$\Delta \mathbf{x}_2^a \mathbf{p}_a A \sim \Delta \mathbf{x}_2^a \mathbf{O} \mathbf{x}_2. \tag{3.8}$$

Since they are similar, *i.e.* they share the same shape, we can measure the distance of $\mathbf{x}_2^a$ from the origin. More formally,

$$\frac{L_a}{\|\mathbf{p}_a\| + \|\mathbf{x}_2^a\|} = \frac{h}{\|\mathbf{x}_2^a\|}, \tag{3.9}$$

from which we can recover

$$\|\mathbf{x}_2^a\| = \frac{h\|\mathbf{p}_a\|}{L_a - h}. \tag{3.10}$$

The orientation of the $\mathbf{x}_2^a$ vector can be obtained directly from the orientation of the $\mathbf{p}_a$ vector, which is known and equal to

$$\overrightarrow{\mathbf{x}_2^a} = -\overrightarrow{\mathbf{p}_a} = -\frac{\mathbf{p}_a}{\|\mathbf{p}_a\|}. \tag{3.11}$$

Eventually, with the magnitude and orientation in place, we can locate the vector pointing to $\mathbf{x}_2^a$:

$$\mathbf{x}_2^a = \|\mathbf{x}_2^a\|\overrightarrow{\mathbf{x}_2^a} = -\frac{h}{L_a - h}\mathbf{p}_a. \tag{3.12}$$

Similarly, $\mathbf{x}_2^b$ can also be computed. The metric error can thus be described by the following relation:

$$\mathbf{e}_w = h\left(\frac{\mathbf{p}_b}{L_b - h} - \frac{\mathbf{p}_a}{L_a - h}\right). \tag{3.13}$$

The error $\mathbf{e}_w$ is a vector, but a convenient scalar can be obtained by using the preferred norm.

Computing the error on a camera reference system When the plane inducing the homography remains unknown, the bound and the error estimation from the previous section cannot be directly applied. A more general case is obtained if the reference system is set off the plane, and in particular, on one of the cameras. The new geometry of the problem is shown in Fig. 3.24(b), where the reference system is placed on camera A. In this setting, the metric error is a function of four independent quantities (highlighted in red in the figure): i) the point $\mathbf{x}_2$, ii) the distance of such point from the inducing plane h, iii) the plane normal $\vec{\mathbf{n}}$ and iv) the distance between the cameras $\mathbf{v}$, which is also equal to the position of camera B.
To this end, starting from Eq. (3.13), we are interested in expressing $\mathbf{p}_b$, $\mathbf{p}_a$, L_b and L_a in terms of this new reference system. Since $\mathbf{p}_a$ is the projection of A on the plane it can also be defined as

$$\mathbf{p}_a = A - (A - \mathbf{p}_k)\vec{\mathbf{n}} \otimes \vec{\mathbf{n}} = A + \boldsymbol{\alpha}\vec{\mathbf{n}} \otimes \vec{\mathbf{n}}, \tag{3.14}$$

where $\vec{\mathbf{n}}$ is the plane normal, $\mathbf{p}_k$ is an arbitrary point on the plane that we set to $\mathbf{x}_2 - h \otimes \vec{\mathbf{n}}$, *i.e.* the projection of $\mathbf{x}_2$ on the plane. To ease the readability of the following equations, $\boldsymbol{\alpha} = -(A - \mathbf{x}_2 - h \otimes \vec{\mathbf{n}})$. Now, if $\mathbf{v}$ describes the distance from A to B, we have

$$\begin{aligned} \mathbf{p}_b &= A + \mathbf{v} - (A + \mathbf{v} - \mathbf{p}_k)\vec{\mathbf{n}} \otimes \vec{\mathbf{n}} \\ &= A + \boldsymbol{\alpha}\vec{\mathbf{n}} \otimes \vec{\mathbf{n}} + \mathbf{v} - \mathbf{v}\vec{\mathbf{n}} \otimes \vec{\mathbf{n}}. \end{aligned} \tag{3.15}$$

Through similar reasoning, L_a and L_b are also rewritten as follows:

$$\begin{aligned} L_a &= A - \mathbf{p}_a = -\boldsymbol{\alpha}\vec{\mathbf{n}} \otimes \vec{\mathbf{n}} \\ L_b &= B - \mathbf{p}_b = -\boldsymbol{\alpha}\vec{\mathbf{n}} \otimes \vec{\mathbf{n}} + \mathbf{v}\vec{\mathbf{n}} \otimes \vec{\mathbf{n}}. \end{aligned} \tag{3.16}$$

Eventually, by substituting Eq. (3.14)-(3.16) in Eq. (3.13) and by fixing the origin on the location of camera A so that $A = (0, 0, 0)^T$, we have:

$$\mathbf{e}_w = \frac{h \otimes \mathbf{v}}{(\mathbf{x}_2 - \mathbf{v})\vec{\mathbf{n}}} \left(\vec{\mathbf{n}} \otimes \vec{\mathbf{n}} (1 - \frac{1}{1 - \frac{h}{h - \mathbf{x}_2 \vec{\mathbf{n}}}}) - I \right). \tag{3.17}$$

Notably, the vector $\mathbf{v}$ and the scalar h both appear as multiplicative factors in Eq. (3.17), so that if any of them goes to zero, then the magnitude of the metric error $\mathbf{e}_w$ also goes to zero.
If we assume that $h \neq 0$, we can go one step further and obtain a formulation were $\mathbf{x}_2$ and $\mathbf{v}$ are always divided by h, suggesting that what really matters is not the absolute position of $\mathbf{x}_2$ or camera B with respect to camera A but rather how many times further $\mathbf{x}_2$ and camera B are from A than $\mathbf{x}_2$ from the plane. Such relation is made explicit below:

$$\mathbf{e}_w = h \underbrace{\frac{\|\mathbf{v}\|/h}{(\frac{\|\mathbf{x}_2\|}{h} \otimes \vec{\mathbf{x}}_2 - \frac{\|\mathbf{v}\|}{h} \otimes \vec{\mathbf{v}})\vec{\mathbf{n}}}}_{Q} \otimes \vec{\mathbf{v}} \left(\vec{\mathbf{n}} \otimes \vec{\mathbf{n}} \underbrace{(1 - \frac{1}{1 - \frac{1}{1 - \frac{\|\mathbf{x}_2\|}{h} \otimes \vec{\mathbf{x}}_2 \vec{\mathbf{n}}}})}_{Z} - I \right). \tag{3.18}$$

Working towards a bound. Let $M = \|\mathbf{x}_2\|/\|\mathbf{v}\| |\cos\theta|$, being θ the angle between $\vec{\mathbf{x}}_2$ and $\vec{\mathbf{n}}$, and let β be the angle between $\vec{\mathbf{v}}$ and $\vec{\mathbf{n}}$. Then Q can be rewritten as $Q = (M - \cos\beta)^{-1}$. Note that under the assumption that $M \geq 2$, $Q \leq 1$ always holds. Indeed for $M \geq 2$ to hold, we need to require $\|\mathbf{x}_2\| \geq 2\|\mathbf{v}\|/|\cos\theta|$. Next, consider the scalar Z: it is easy to verify that if $|\|\mathbf{x}_2\|/h \otimes \vec{\mathbf{x}}_2 \vec{\mathbf{n}}| > 1$, then $|Z| \leq 1$. Since both $\vec{\mathbf{x}}_2$ and $\vec{\mathbf{n}}$ are versors, the magnitude of their dot product is at most one. It follows that $|Z| < 1$ if and only if

$\|\mathbf{x}_2\| > h$. Now we are left with a versor $\vec{\mathbf{v}}$ that multiplies the difference of two matrices. If we compute such product we obtain a new vector with magnitude less or equal to one, $\vec{\mathbf{v}}\vec{\mathbf{n}} \otimes \vec{\mathbf{n}}$, and the versor $\vec{\mathbf{v}}$. The difference of such vectors is at most 2. Summing up all the presented considerations, we have that the magnitude of the error is bounded as follows.

Observation 1
If $\|\mathbf{x}_2\| \geq 2\|\mathbf{v}\|/|\cos\theta|$ *and* $\|\mathbf{x}_2\| > h$*, then* $\|\mathbf{e}_w\| \leq 2h$.

We now aim to derive a projection error bound from the above presented metric error bound. In order to do so, we need to introduce the focal length of the camera f. For simplicity, we'll assume that $f = f_x = f_y$. First, we simplify our setting without loosing the upper bound constraint. To do so, we consider the worst case scenario, in which the mutual position of the plane and the camera maximizes the projected error:

- the plane rotation is so that $\vec{\mathbf{n}} /\!/ z$;
- the error segment is just in front of the camera;
- the plane rotation along the z axis is such that the parallel component of the error w.r.t. the x axis is zero (this allows us to express the segment end points with simple coordinates without loosing generality);
- the camera A falls on the middle point of the error segment.

In the simplified scenario depicted in Fig. 3.24(c), the projection of the error is maximized. In this case, the two points we want to project are $\mathbf{p}_1 = [0, -h, \gamma]$ and $\mathbf{p}_2 = [0, h, \gamma]$ (we consider the case in which $||\mathbf{e}_w|| = 2h$, see Observation. 1) where γ is the distance of the camera from the plane. Considering the focal length f of camera A, $\mathbf{p}_1$ and $\mathbf{p}_2$ are projected as follows:

$$K_A\mathbf{p}_1 = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ -h \\ \gamma \end{bmatrix} = \begin{bmatrix} 0 \\ -fh \\ \gamma \end{bmatrix} \rightarrow \begin{bmatrix} 0 \\ -\frac{fh}{\gamma} \end{bmatrix} \tag{3.19}$$

$$K_A\mathbf{p}_2 = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ h \\ \gamma \end{bmatrix} = \begin{bmatrix} 0 \\ fh \\ \gamma \end{bmatrix} \rightarrow \begin{bmatrix} 0 \\ \frac{fh}{\gamma} \end{bmatrix} \tag{3.20}$$

Thus, the magnitude of the projection $\|\mathbf{e}_a\|$ of the metric error $\|\mathbf{e}_w\|$ is bounded by $\frac{2fh}{\gamma}$.
Now, we notice that $\gamma = h + \mathbf{x}_2\vec{\mathbf{n}} = h + \|\mathbf{x}_2\|cos(\theta)$, so

$$\|\mathbf{e}_a\| \leq \frac{2fh}{\gamma} = \frac{2fh}{h + \|\mathbf{x}_2\|cos(\theta)} = \frac{2f}{1 + \frac{\|\mathbf{x}_2\|}{h}cos(\theta)} \tag{3.21}$$

Notably, the right term of the equation is maximized when $\cos(\theta) = 0$ (since when $cos(\theta) < 0$ the point is behind the camera, which is impossible in our setting). Thus, we obtain that $\|\mathbf{e}_a\| \leq 2f$.

Fig. 3.23(b) shows a use case of the bound in Eq. 3.21. It shows values of θ up to $pi/2$, where the presented bound simplifies to $\|\mathbf{e}_a\| \leq 2f$ (dashed black line). In practice, if we require i) $\theta \leq \pi/3$ and ii) that the camera-object distance $\|\mathbf{x}_2\|$ is at least three times the plane-object distance h, and if we let $f = 350$px, then the error is always lower than 200px, which translate to a precision up to 20% of an image at 1080p resolution.

3.4.5.5 The effect of random cropping

In order to advocate for the peculiar training strategy illustrated in Sec. 4.1, involving two streams processing both resized clips and randomly cropped clips, we perform an additional experiment as follows. We first re-train our `multi-branch` architecture following the same procedures explained in Sec. 3.3, except for the cropping of input clips and groundtruth maps, which is always central rather than random. At test time we shift each input clip in the range $[-800, 800]$ pixels (negative and positive shifts indicate left and right translations respectively). After the translation, we apply mirroring to fill borders on the opposite side of the shift direction. We perform the same operation on groundtruth maps and report the mean D_{KL} of the `multi-branch` model when trained with random and central crops, as a function of the translation size, in Fig. 3.25. The figure highlights how random cropping consistently outperforms central cropping. Importantly, the gap in performance increases with amount of shift applied, from a 27.86% relative D_{KL} difference when no translation is performed to 258.13% and 216.17% increases for -800 px and 800 px translations, suggesting the model trained with central crops is not robust to positional shifts.

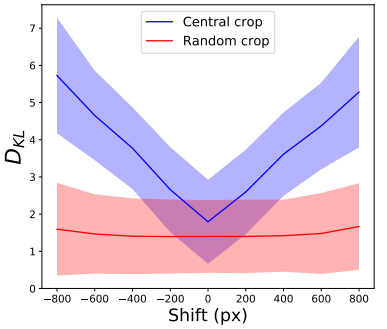


Figure 3.25: Performances of the `multi-branch` model trained with random and central crop policies, in presence of horizontal shifts in input clips. Colored background regions highlight the area within the standard deviation of the measured divergence. Recall that a lower value of D_{KL} means a more accurate prediction.

3.4.5.6 The effect of padded convolutions in learning a central bias

Learning a globally localized solution seems theoretically impossible when using a fully convolutional architecture. Indeed, convolutional kernels are applied uniformly along all spatial dimension of the input feature map. Conversely, a globally localized solution requires knowing where kernels are applied during convolutions. We argue that a convolutional element can know its absolute position if there are latent statistics contained in the activations of the previous layer. In what follows, we show how the common habit of padding feature maps before feeding them to convolutional layers, in order to maintain borders, is an underestimated source of spatially localized statistics. Indeed, padded values are always constant, and unrelated to the input feature map. Thus, a convolutional operation, depending on its receptive field, can localize itself in the feature map by looking for statistics biased by the padding values. To validate this claim, we design a toy experiment in which a fully convolutional neural network is tasked to regress a white central square on a black background, when provided with a uniform or a noisy input map (in both cases, the target is independent from the input). We position the square (bias element) at the center of the output as it is the furthest position from borders, *i.e.* where the bias originates. We perform

the experiment with several networks featuring the same number of parameters yet different receptive fields[§]. Moreover, to advocate for the random cropping strategy employed in the training phase of our network (recall that it was introduced to prevent a saliency-branch to regress a central bias), we repeat each experiment employing such strategy during training. All models were trained to minimize the mean squared error between target maps and predicted maps, by means of Adam optimizer[¶]. The outcome of such experiments, in terms of regressed prediction maps and loss function value at convergence, are reported in Fig. 3.26 and Fig. 3.27. As shown by the top-most region of both figures, despite the uncorrelation between input and target all models can learn a central biased map. Moreover, the receptive field plays a crucial role, as it controls the amount of pixels able to "localize themselves" within the predicted map. As the receptive field of the network increases, the responsive area shrinks to the groundtruth area, and loss value lowers reaching zero. For an intuition of why this is the case, we refer the reader to Fig. 3.28. Conversely, as clearly emerges from the bottom-most region of both figures, random cropping prevents the model to regress a biased map, regardless the receptive field of the network. The reason underlying this phenomenon is that padding is applied after the crop, so its relative position with respect to the target depends on the crop location, which is random.

[§] a simple way to decrease the receptive field without changing the number of parameters is to replace two convolutional layers featuring C output channels into a single one featuring $2C$ output channels.

[¶] the code to reproduce this experiment is publicly released at https://github.com/DavideA/can_i_learn_central_bias/.

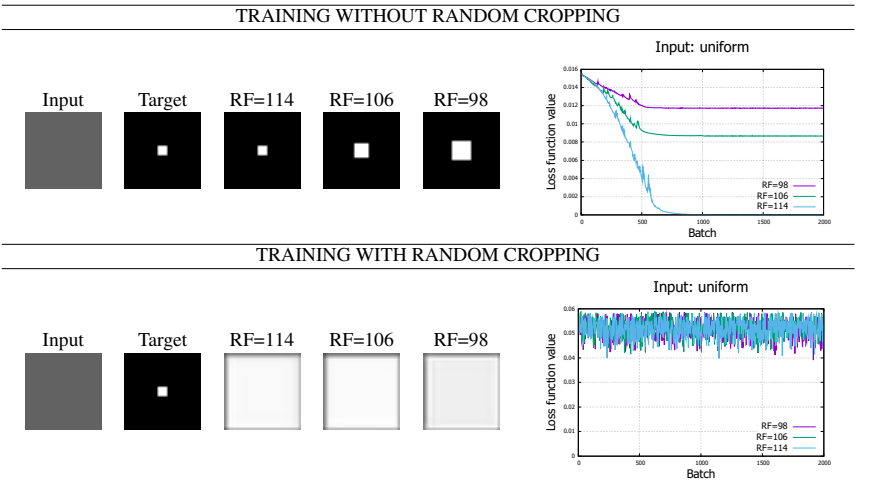


Figure 3.26: To show the beneficial effect of random cropping in preventing a convolutional network to learn a biased map, we train several models to regress a fixed map from uniform input images. We argue that, in presence of padded convolutions and big receptive fields, the relative location of the groundtruth map with respect to image borders is fixed, and proper kernels can be learned to localize the output map. We report output solutions and loss functions for different receptive fields. As the receptive field grows, the portion of the image accessing borders grows and the solution improves (reaching lower loss values). Please note that in this setting the input image is 128x128, while the output map is 32x32 and the central bias is 4x4. Therefore, the minimum receptive field required to solve the task is $2 * (32/2 - 4/2) * (128/32) = 112$. Conversely, when trained by randomly cropping images and groundtruth maps, the reliable statistic (in this case, the relative location of the map with respect to image borders) ceases to exist, making the training process hopeless.

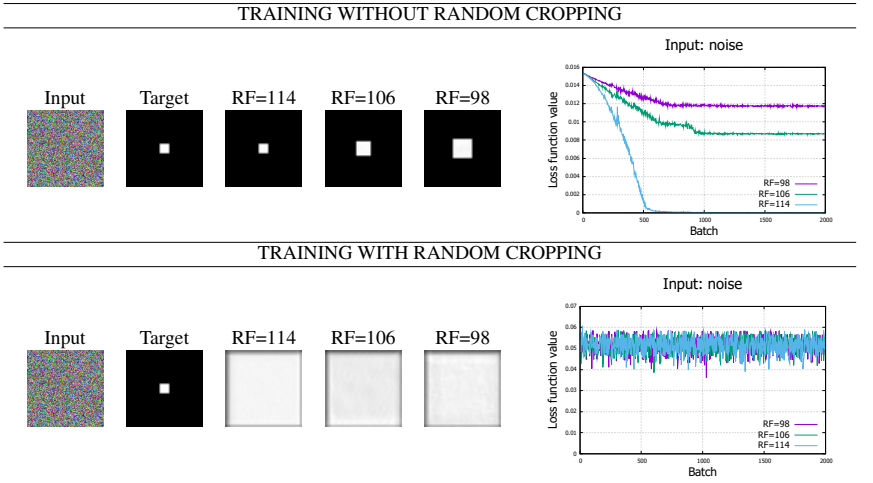


Figure 3.27: This figure illustrates the same content as Fig. 3.26, but reports output solutions and loss functions obtained from noisy inputs. See Fig. 3.26 for further details.

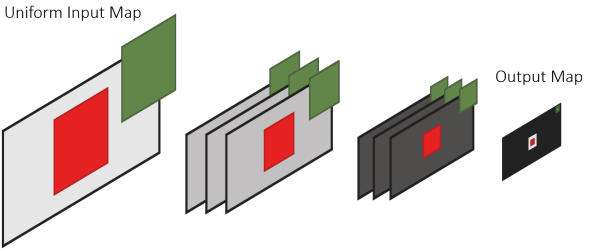


Figure 3.28: Importance of the receptive field in the task of regressing a central bias exploiting padding: the receptive field of the red pixel in the output map has no access to padding statistics, so it will be activated. On the contrary, the green pixel's receptive field exceeds image borders: in this case, padding is a reliable anchor to break the uniformity of feature maps.

Chapter 4

Latent Space Autoregression for Novelty Detection

In this chapter, we study novelty detection, which is defined as the identification of samples which exhibit significantly different traits with respect to an underlying model of regularity, built from a collection of normal samples. The awareness of an autonomous system to recognize unknown events enables applications in several domains, ranging from video surveillance [25, 59], to defect detection [91] to medical imaging [161]. Moreover, the surprise inducted by unseen events is emerging as a crucial aspect in reinforcement learning settings, as an enabling factor in curiosity-driven exploration [138].

However, in this setting, the definition and labeling of novel examples are not possible. Accordingly, the literature agrees on approximating the ideal shape of the boundary separating normal and novel samples by modeling the intrinsic characteristics of the former. Therefore, prior works tackle such problem by following principles derived from the unsupervised learning paradigm [34, 159, 59, 105, 112]. Due to the lack of a supervision signal, the process of feature extraction and the rule for their normality assessment can only be guided by a proxy objective, assuming the latter will define an appropriate boundary for the application at hand.

According to cognitive psychology [10], novelty can be expressed either in terms of capabilities to *remember* an event or as a degree of *surprisal* [183] aroused by its observation. The latter is mathematically modeled in terms of low probability

to occur under an expected model, or by lowering a variational free energy [71]. In this framework, prior models take advantage of either parametric [213] or non-parametric [63] density estimators. Differently, remembering an event implies the adoption of a memory represented either by a dictionary of normal prototypes - as in sparse coding approaches [34] - or by a low dimensional representation of the input space, as in the self-organizing maps [86] or, more recently, in deep autoencoders. Thus, in novelty detection, the remembering capability for a given sample is evaluated either by measuring reconstruction errors [59, 105] or by performing discriminative in-distribution tests [159].

Our proposal contributes to the field by merging remembering and surprisal aspects into a unique framework: we design a generative unsupervised model (i.e., an autoencoder, represented in Fig. 4.1) that exploits end-to-end training in order to maximize remembering effectiveness for normal samples whilst minimizing the surprisal of their latent representation. This latter point is enabled by the maximization of the likelihood of latent representations through an autoregressive density estimator, which is performed in conjunction with the reconstruction error minimization. We show that, by optimizing both terms jointly, the model implicitly seeks for minimum entropy representations maintaining its remembering/reconstructive power. While entropy minimization approaches have been adopted in deep neural compression [8], to our knowledge this is the first proposal tailored for novelty detection. In memory terms, our procedure resembles the concept of prototyping the normality using as few templates as possible. Moreover, evaluating the output of the estimator enables the assessment of the surprisal aroused by a given sample.

4.1 Proposed model

Maximizing the probability of latent representations is analogous to lowering the surprisal of the model for a normal configuration, defined as the negative log-density of a latent variable instance [183]. Conversely, remembering capabilities can be evaluated by the reconstruction accuracy of a given sample under its latent representation.

We model the aforementioned aspects in a latent variable model setting, where the density function of training samples $p(\mathbf{x})$ is modeled through an auxiliary random variable $\mathbf{z}$, describing the set of causal factors underlying all observations. By

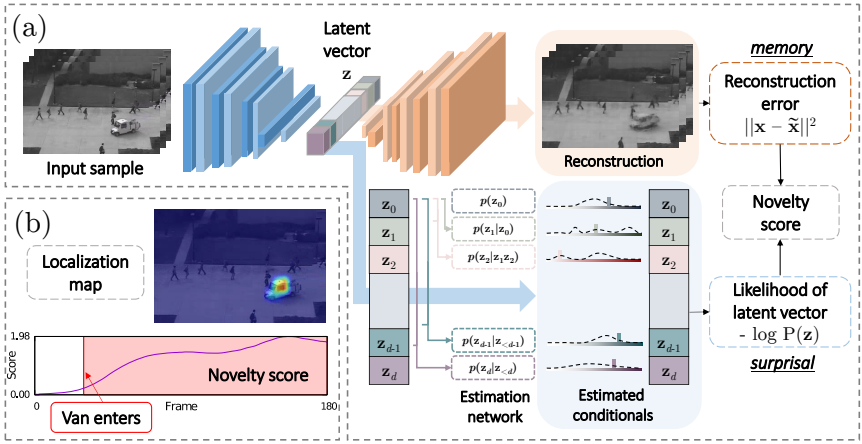


Figure 4.1: The proposed novelty detection framework. The overall architecture, depicted in (a), consists of a deep autoencoder and an autoregressive estimation network operating on its latent space. The joint minimization of their respective objective leads to a measure of novelty - (b) - obtained by assessing the remembrance of the model when looking to a new sample, combined with its surprise aroused by causal factors.

factorizing

$$p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z}, \tag{4.1}$$

where $p(\mathbf{x}|\mathbf{z})$ is the conditional likelihood of the observation given a latent representation $\mathbf{z}$ with prior distribution $p(\mathbf{z})$, we can explicit both the memory and surprisal contribution to novelty. We approximate the marginalization by means of an inference model responsible for the identification of latent space vector for which the contribution of $p(\mathbf{x}|\mathbf{z})$ is maximal. Formally, we employ a deep autoencoder, in which the reconstruction error plays the role of the negative logarithm of $p(\mathbf{x}|\mathbf{z})$, under the hypothesis that $p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|\tilde{\mathbf{x}}, I)$ where $\tilde{\mathbf{x}}$ denotes the output reconstruction. Additionally, surprisal is injected in the process by equipping the autoencoder with an auxiliary deep parametric estimator learning the prior distribution $p(\mathbf{z})$ of latent vectors, and training it by means of Maximum Likelihood Estimation (MLE). Our architecture is therefore composed of three building blocks (Fig. 4.1): an encoder $f(\mathbf{x}; \theta_f)$, a decoder $g(\mathbf{z}; \theta_g)$ and a probabilistic

model $h(\mathbf{z};\theta_h)$:

$$f(\mathbf{x};\theta_f) : \mathbb{R}^m \rightarrow \mathbb{R}^d, \qquad g(\mathbf{z};\theta_g) : \mathbb{R}^d \rightarrow \mathbb{R}^m, \\ h(\mathbf{z};\theta_h) : \mathbb{R}^d \rightarrow [0,1]. \tag{4.2}$$

The encoder processes input $\mathbf{x}$ and maps it into a compressed representation $\mathbf{z} = f(\mathbf{x};\theta_f)$, whereas the decoder provides a reconstructed version of the input $\tilde{\mathbf{x}} = g(\mathbf{z};\theta_g)$. The probabilistic model $h(\mathbf{z};\theta_h)$ estimates the density in $\mathbf{z}$ via an autoregressive process, allowing to avoid the adoption of a specific family of distributions (i.e., Gaussian), potentially unrewarding for the task at hand. On this latter point, please refer to Sec. 4.2.5.2 for a comparison w.r.t. variational autoencoders [83].
With such modules, at test time, we can assess the two sources of novelty: elements whose observation is poorly explained by the causal factors inducted by normal samples (i.e., high reconstruction error); elements exhibiting good reconstructions whilst showing surprising underlying representations under the learned prior.

4.1.1 Autoregressive density estimation.

Autoregressive models provide a general formulation for tasks involving sequential predictions, in which each output depends on previous observations [111, 133]. We adopt such a technique to factorize a joint distribution, thus avoiding to define its landscape a priori [96, 186]. Formally, $p(\mathbf{z})$ is factorized as

$$p(\mathbf{z}) = \prod_{i=1}^{d} p(z_i|\mathbf{z}_{<i}), \tag{4.3}$$

so that estimating $p(\mathbf{z})$ reduces to the estimation of each single Conditional Probability Density (CPD) expressed as $p(z_i|\mathbf{z}_{<i})$, where the symbol $<$ implies an order over random variables. We refer the reader to Fig. 4.2 for a graphical representation of the factorization in Eq. 4.3.
Some prior models obey handcrafted orderings [190, 189], whereas others rely on order agnostic training [187, 53]. Nevertheless, it is still not clear how to estimate the proper order for a given set of variables. In our model, this issue is directly tackled by the optimization. Indeed, since we perform autoregression on learned latent representations, the MLE objective encourages the autoencoder to impose over them a pre-defined causal structure. Empirical evidence of this phenomenon is given in Sec. 4.2.5.3.

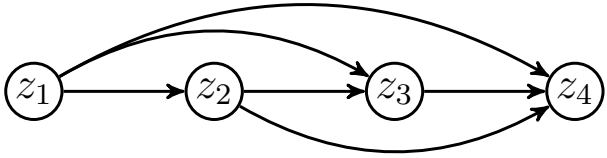


Figure 4.2: An example of autoregression over the joint probability distribution of four variables, represented as a graphical model. According to this model, estimating the joint probability of z_1, z_2, z_3, z_4 reduces to the estimation of the conditionals $p(z_4|z_3z_2z_1)$, $p(z_3|z_2z_1)$, $p(z_2|z_1)$ and the marginal $p(z_1)$.

From a technical perspective, the estimator $h(\mathbf{z};\theta_h)$ outputs parameters for d distributions $p(z_i|\mathbf{z}_{<i})$. In our implementation, each CPD is modeled as a multinomial over B=100 quantization bins. To ensure a conditional estimate of each underlying density, we design proper layers guaranteeing that the CPD of each symbol z_i is computed from inputs $\{z_1, \ldots, z_{i-1}\}$ only.

4.1.2 Objective and connection with differential entropy.

The three components f, g and h are jointly trained to minimize $\mathcal{L} \equiv \mathcal{L}(\theta_f, \theta_g, \theta_h)$ as follows:

$$\begin{aligned}\mathcal{L} &= \mathcal{L}_{\text{REC}}(\theta_f, \theta_g) + \lambda \mathcal{L}_{\text{LLK}}(\theta_f, \theta_h) \\ &= \mathbb{E}_{\mathbf{x}}\Big[\underbrace{||\mathbf{x} - \tilde{\mathbf{x}}||^2}_{\text{reconstruction term}} - \lambda \underbrace{\log(h(\mathbf{z};\theta_h))}_{\text{log-likelihood term}} \Big],\end{aligned} \tag{4.4}$$

where λ is a hyper-parameter controlling the weight of the $\mathcal{L}_{\text{LLK}}$ term.
Importantly, as mentioned before, we model each CPD through a multinomial. To this aim, we firstly need that the encoder acts as a bounded function. To achieve such desideratum, we simply employ a sigmoidal activation, ensuring that latent representations $\mathbf{z} = f(\mathbf{x};\theta_f)$ reside in $[0,1]^d$. Therefore, for each z_j with $j = 1, 2, \ldots, d$, we perform a linear quantization of the space $[0,1]$ in B bins (where B is a hyperparameter). This latter step provides for z_j a B-dimensional categorical distribution $\phi(z_j)$, highlighting the correct bin to which z_j belongs. For each CPD, such distribution will serve as ground truth for the estimator $h(\mathbf{z};\theta_h)$, the latter coherently predicting d distributions $p(z_j|\mathbf{z}_{<j})$ across

the B bins, employing a softmax activation. This way, as shown in Eq. 4.5, the $\mathcal{L}_{\text{LLK}}$ loss turns out to be a valid likelihood term, defined as the cross-entropy loss between each one of the estimated CPD and their categorical counterparts:

$$\mathcal{L}_{\text{LLK}}(\theta_f, \theta_h) = \mathbb{E}_{\mathbf{x}\sim P}\Big[-\sum_{j=1}^{d}\sum_{k=1}^{B} \phi(z_j)_k \log(p(z_j|\mathbf{z}_{<j})_k)\Big]. \tag{4.5}$$

It is worth noting that multinomials are just one of the plausible models for the CPDs. Indeed, if we replace them with Gaussians, the overall framework would leave standing. However, as we observed in different trials, this choice does not yield considerable improvements but rather numerical instabilities, as described in prior works [190].

We now delve deeper into the objective and its implications from an information-theory perspective. It is worth noting that it is possible to express the log-likelihood term as

$$\begin{aligned}\mathcal{L}_{\text{LLK}}(\theta_f, \theta_h) =& \mathbb{E}_{\mathbf{z}\sim p^*(\mathbf{z};\theta_f)}\big[-\log h(\mathbf{z};\theta_h)\big] \\ =& \mathbb{E}_{\mathbf{z}\sim p^*(\mathbf{z};\theta_f)}\big[-\log h(\mathbf{z};\theta_h) + \log p^*(\mathbf{z};\theta_f) - \log p^*(\mathbf{z};\theta_f)\big] \\ =& D_{\text{KL}}(p^*(\mathbf{z};\theta_f) \parallel h(\mathbf{z};\theta_h)) + \mathbb{H}[p^*(\mathbf{z};\theta_f)],\end{aligned} \tag{4.6}$$

where $p^*(\mathbf{z};\theta_f)$ denotes the true distribution of the codes produced by the encoder, and is therefore parametrized by θ_f. This reformulation of the MLE objective yields meaningful insights about the entities involved in the optimization. On the one hand, the Kullback-Leibler divergence ensures that the information gap between our parametric model h and the true distribution p^* is small. On the other hand, this framework leads to the minimization of the differential entropy of the distribution underlying the codes produced by the encoder f. Such constraint constitutes a crucial point when learning normality. Intuitively, if we think about the encoder as a source emitting symbols (namely, the latent representations), its desired behavior, when modeling normal aspects in the data, should converge to a 'boring' process characterized by an intrinsic low entropy, since surprising and novel events are unlikely to arise during the training phase. Accordingly, among all the possible settings of the hidden representations, the objective begs the encoder to exhibit a low differential entropy, leading to the extraction of features that are easily predictable, therefore common and recurrent within the training set. This kind of features is indeed the most useful to distinguish novel samples

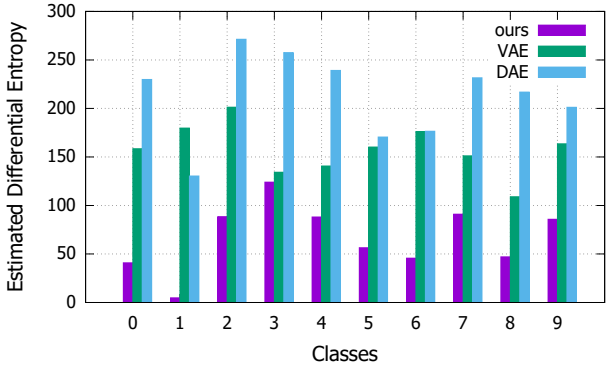


Figure 4.3: Estimated differential entropies delivered on each MNIST class in the presence of different regularization strategies: our, divergence w.r.t a Gaussian prior (VAE) and input perturbation (DAE). For each class, the estimate is computed on the training samples' hidden representations, whose distribution are fit utilizing a Gaussian KDE in a 3D-space. All models being equal, ours exhibits lower entropies on all classes.

from the normal ones, making our proposal a suitable regularizer in the anomaly detection setting.
We report empirical evidence of the decreasing differential entropy in Fig. 4.3, that compares the behavior of the same model under different regularization strategies.

4.1.3 Architectural Components

4.1.3.1 Autoencoder blocks.

Encoder and decoder are respectively composed by downsampling and upsampling residual blocks depicted in Fig. 4.4. The encoder ends with fully connected (FC) layers. When dealing with video inputs, we employ *causal* 3D convolutions [7] within the encoder (i.e., only accessing information from previous time-steps). Moreover, at the end of the encoder, we employ a temporally-shared full connection (TFC, namely a linear projection sharing parameters across the time axis on the input feature maps) resulting in a temporal series of feature vectors. This way, the encoding procedure does not shuffle information across time-steps, ensuring temporal ordering.

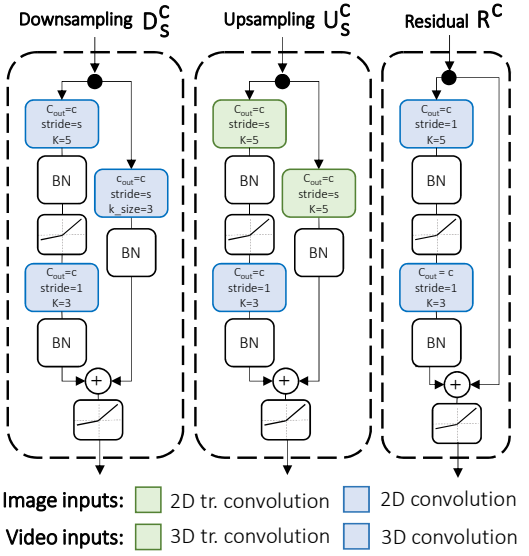


Figure 4.4: Building blocks employed in the autoencoder's architecture.

4.1.3.2 Autoregressive layers.

To guarantee the autoregressive nature of each output CPD, we need to ensure proper connectivity patterns in each layer of the estimator h. Moreover, since latent representations exhibit different shapes depending on the input nature (image or video), we propose two different solutions.

When dealing with images, the encoder provides feature vectors with dimensionality d. The autoregressive estimator is composed by stacking multiple Masked Fully Connections (MFC, Fig. 4.5). Formally, it computes output feature map $\mathbf{o} \in \mathbb{R}^{d \times co}$ (where co is the number of output channels) given the input $\mathbf{h} \in \mathbb{R}^{d \times ci}$ (assuming $ci = 1$ at the input layer). The connection between the input

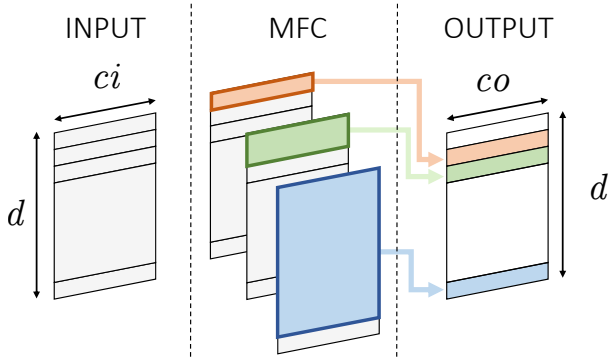


Figure 4.5: Proposed autoregressive layer for images, namely the Masked Fully Connection (Eq. 4.7) We represent type A structure. Different colors represent different parametrizations.

element $\mathbf{h}_i^k$ in position i, channel k and the output element $\mathbf{o}_j^l$ is parametrized by

$$\begin{cases} w_{i,j}^{k,l} & \text{if } i < j \\ \begin{cases} w_{i,j}^{k,l} & \text{if type = B} \\ 0 & \text{if type = A} \end{cases} & \text{if } i = j \\ 0 & \text{if } i > j. \end{cases} \tag{4.7}$$

Type A forces a strict dependence on previous elements (and is employed only as the first estimator layer), whereas type B masks only succeeding elements. Assuming each CPD modeled as a multinomial, the output of the last autoregressive layer (in $\mathbb{R}^{d \times B}$) provides probability estimates for the B bins that compose the space quantization.
On the other hand, the compressed representation of video clips has dimensionality $t \times d$, being t the number of temporal time-steps and d the length of the code. Accordingly, the estimation network is designed to capture two-dimensional patterns within observed elements of the code. However, naively plugging 2D convolutional layers would assume translation invariance on both axes of the input map, whereas, due to the way the compressed representation is built, this assumption is only correct along the temporal axis. To cope with this, we apply d different convolutional kernels along the code axis, allowing the observation of the whole feature vector in the previous time-step as well as a portion of the

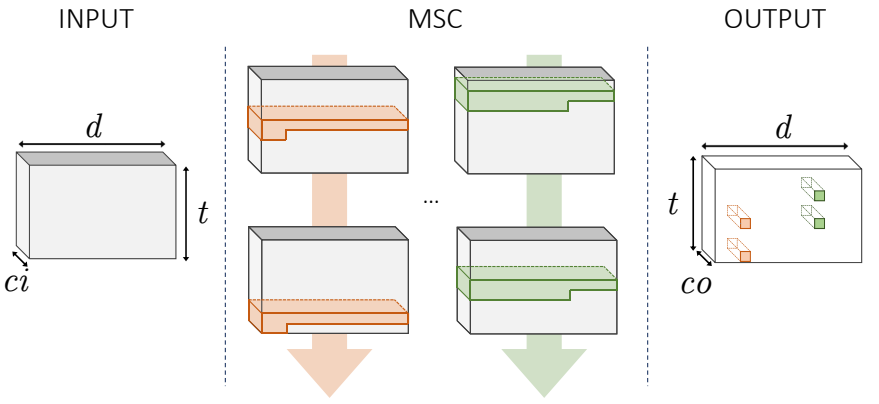


Figure 4.6: Proposed autoregressive layer fr videos, namely the Masked Stacked Convolution (b, Eq. 4.8). We represent type A structure. Different kernel colors represent different parametrizations.

current one. Every convolution is free to stride along the time axis and captures temporal patterns. In such operation, named Masked Stacked Convolution (MSC, Fig. 4.6), the i-th convolution is equipped with a kernel $\mathbf{w}^{(i)} \in \mathbb{R}^{3\times d}$ kernel, that gets multiplied by the binary mask $\mathbf{M}^{(i)}$, defined as

$$m_{j,k}^{(i)} \in \mathbf{M}^{(i)} = \begin{cases} 1 & \text{if } j = 0 \\ 1 & \text{if } j = 1 \text{ and } k < i \text{ and type=A} \\ 1 & \text{if } j = 1 \text{ and } k \leq i \text{ and type=B} \\ 0 & \text{otherwise,} \end{cases} \tag{4.8}$$

where j indexes the temporal axis and k the code axis.
Every single convolution yields a column vector, as a result of its stride along time. The set of column vectors resulting from the application of the d convolutions to the input tensor $\mathbf{h} \in \mathbb{R}^{t\times d\times ci}$ are horizontally stacked to build the output tensor $\mathbf{o} \in \mathbb{R}^{t\times d\times co}$, as follows:

$$\mathbf{o} = \Big|\Big|_{i=1}^{d} [(\mathbf{M}^{(i)} \odot \mathbf{w}^{(i)}) * \mathbf{h}], \tag{4.9}$$

where $||$ represents the horizontal concatenation operation.

4.2 Experiments

	MNIST	CIFAR-10	UCSD Ped2	ShanghaiTech	DR(eye)VE
Input Shape	1,28,28	3,32,32	1,8,32,32*	3,16,256,512	1,16,160,256
Encoder Network	$\mathrm{D}^{32}_{2,2}$ $\mathrm{D}^{64}_{2,2}$ FC^{64} FC^{64}	$\mathrm{2D\ Conv}^{32}_{3\mathrm{X}3}$ R^{32} $\mathrm{D}^{64}_{2,2}$ $\mathrm{D}^{128}_{2,2}$ $\mathrm{D}^{256}_{2,2}$ FC^{256} FC^{64}	$\mathrm{D}^{8}_{1,2,2}$ $\mathrm{D}^{12}_{2,1,1}$ $\mathrm{D}^{18}_{1,2,2}$ $\mathrm{D}^{27}_{2,1,1}$ $\mathrm{D}^{40}_{1,2,2}$ TFC^{64}	$\mathrm{D}^{8}_{1,2,2}$ $\mathrm{D}^{16}_{1,2,2}$ $\mathrm{D}^{32}_{2,2,2}$ $\mathrm{D}^{64}_{1,2,2}$ $\mathrm{D}^{64}_{2,2,2}$ TFC^{512} TFC^{64}	$\mathrm{D}^{8}_{1,2,2}$ $\mathrm{D}^{16}_{1,2,2}$ $\mathrm{D}^{32}_{2,2,2}$ $\mathrm{D}^{64}_{1,2,2}$ $\mathrm{D}^{64}_{2,2,2}$ TFC^{512} TFC^{64}
Decoder Network	FC^{64} FC^{64} $\mathrm{U}^{32}_{2,2}$ $\mathrm{U}^{16}_{2,2}$ $\mathrm{2D\ Conv}^{1}_{1\mathrm{x}1}$	FC^{256} FC^{256} $\mathrm{U}^{128}_{2,2}$ $\mathrm{U}^{64}_{2,2}$ $\mathrm{U}^{32}_{2,2}$ R^{32} $\mathrm{2D\ Conv}^{3}_{1\mathrm{x}1}$	TFC^{64} $\mathrm{U}^{40}_{1,2,2}$ $\mathrm{U}^{27}_{1,2,2}$ $\mathrm{U}^{18}_{2,1,1}$ $\mathrm{U}^{12}_{1,2,2}$ $\mathrm{U}^{8}_{2,1,1}$ $\mathrm{3D\ Conv}^{1}_{1\mathrm{x}1}$	TFC^{64} TFC^{512} $\mathrm{U}^{64}_{2,2,2}$ $\mathrm{U}^{32}_{1,2,2}$ $\mathrm{U}^{16}_{2,2,2}$ $\mathrm{U}^{8}_{1,2,2}$ $\mathrm{U}^{8}_{1,2,2}$ $\mathrm{3D\ Conv}^{3}_{1\mathrm{x}1}$	TFC^{64} TFC^{512} $\mathrm{U}^{64}_{2,2,2}$ $\mathrm{U}^{32}_{1,2,2}$ $\mathrm{U}^{16}_{2,2,2}$ $\mathrm{U}^{8}_{1,2,2}$ $\mathrm{U}^{8}_{1,2,2}$ $\mathrm{3D\ Conv}^{1}_{1\mathrm{x}1}$
Estimator Network	MFC^{32} MFC^{32} MFC^{32} MFC^{32} MFC^{100}	MFC^{32} MFC^{32} MFC^{32} MFC^{32} MFC^{100}	MSC^{4} MSC^{4} MSC^{4} MSC^{4} MSC^{100}	MSC^{4} MSC^{4} MSC^{100}	MSC^{4} MSC^{4} MSC^{4} MSC^{4} MSC^{100}
Mini Batch	256	256	2760	8	16
Learning Rate	10^{-4}	10^{-3}	10^{-3}	10^{-3}	10^{-3}
λ	1	0.1	0.1	1	1

*Patches extracted from input clips having shape 1,16,256,384.

Table 4.1: Architectural and optimization hyperparameters of each setting. We denote with D^{C}_{S} (downsampling), U^{C}_{S} (upsampling) and R^{C} (residual) the parametrizations for the employed building blocks (see Fig. 4.4). On the one hand, C is the number of output channels, whereas S is the stride of the first convolution in the block. Additionally, FC^{C} and TFC^{C} denote dense layers and temporally-shared full connections respectively (in this case, C is the number of output features). Finally, we refer to MFC^{C} and MSC^{C} for the proposed autoregressive layers, illustrated in Fig. 4.5 and 4.6. For a comprehensive description of each type of layer, please refer to Sec. 4.1.3.

We test our solution in three different settings: images, videos, and cognitive data. Code to reproduce results in this section is released at https://github.com/aimagelab/novelty-detection. In all experiments the novelty assessment on the i-th example is carried out by summing the reconstruction term (REC_i) and the log-likelihood term (LLK_i) in Eq. 4.4 in a single novelty score NS_i:

$$NS_i = norm_S(REC_i) + norm_S(LLK_i). \tag{4.10}$$

Individual scores are normalized using a reference set of examples S (different for every experiment),

$$norm_S(L_i) = \frac{L_i - \max_{j \in S} L_j}{\max_{j \in S} L_j - \min_{j \in S} L_j}. \tag{4.11}$$

Architectures and hyperparameters employed for each experiment are reported in Tab. 4.1, in terms of the type of blocks, autoregressive layers, mini-batch size, learning rate and weight of the log-likelihood objective. All intermediate layers are Leaky ReLU activated. The objective function is optimized using Adam [81]. All hyperparameters are tuned on a held-out validation set, by minimizing the raw objective (Eq. 4.4 with $\lambda = 1$).

4.2.1 One-class novelty detection on images

To assess the model's performances in one class settings, we train it on each class of either MNIST or CIFAR-10 separately. In the test phase, we present the corresponding test set, which is composed of 10000 examples of all classes, and expect our model to assign a lower novelty score to images sharing the label with training samples. We use standard train/test splits, and isolate 10% of training samples for validation purposes, and employ it as the normalization set (S in Eq. 4.10) for the computation of the novelty score.
As for the baselines, we consider the following:

- standard methods such as OC-SVM [162] and Kernel Density Estimator (KDE), employed out of features extracted by PCA-whitening;
- a denoising autoencoder (DAE) sharing the same architecture as our proposal, but defective of the density estimation module. The reconstruction error is employed as a measure of normality vs. novelty;
- a variational autoencoder (VAE) [83], also sharing the same architecture as our model, in which the Evidence Lower Bound (ELBO) is employed as the score;

- Pix-CNN [189], modeling the density by applying autoregression directly in the image space;
- the GAN-based approach illustrated in [161].

MNIST							
	OC SVM	KDE	DAE	VAE	Pix CNN	GAN	ours
0	0.988	0.885	0.991	0.998	0.531	0.926	0.993
1	0.999	0.996	0.999	0.999	0.995	0.995	0.999
2	0.902	0.710	0.891	0.962	0.476	0.805	0.959
3	0.950	0.693	0.935	0.947	0.517	0.818	0.966
4	0.955	0.844	0.921	0.965	0.739	0.823	0.956
5	0.968	0.776	0.937	0.963	0.542	0.803	0.964
6	0.978	0.861	0.981	0.995	0.592	0.890	0.994
7	0.965	0.884	0.964	0.974	0.789	0.898	0.980
8	0.853	0.669	0.841	0.905	0.340	0.817	0.953
9	0.955	0.825	0.960	0.978	0.662	0.887	0.981
avg	0.951	0.814	0.942	0.969	0.618	0.866	**0.975**
CIFAR10							
	OC SVM	KDE	DAE	VAE	Pix CNN	GAN	ours
0	0.630	0.658	0.718	0.688	0.788	0.708	0.735
1	0.440	0.520	0.401	0.403	0.428	0.458	0.580
2	0.649	0.657	0.685	0.679	0.617	0.664	0.690
3	0.487	0.497	0.556	0.528	0.574	0.510	0.542
4	0.735	0.727	0.740	0.748	0.511	0.722	0.761
5	0.500	0.496	0.547	0.519	0.571	0.505	0.546
6	0.725	0.758	0.642	0.695	0.422	0.707	0.751
7	0.533	0.564	0.497	0.500	0.454	0.471	0.535
8	0.649	0.680	0.724	0.700	0.715	0.713	0.717
9	0.508	0.540	0.389	0.398	0.426	0.458	0.548
avg	0.586	0.610	0.590	0.586	0.551	0.592	**0.641**

Table 4.2: AUROC results for novelty detection on MNIST and CIFAR10. Each row represents a different class on which baselines and our model are trained.

We report the comparison in Tab. 4.2 in which performances are measured by the Area Under Receiver Operating Characteristic (AUROC), which is the standard metric for the task. As the table shows, our proposal outperforms all baselines in both settings.

Considering MNIST, most methods perform favorably. Notably, Pix-CNN fails in modeling distributions for all digits but one, possibly due to the complexity of modeling densities directly on pixel space and following a fixed autoregression order. Such poor test performances are registered despite good quality samples that we observed during training: indeed, the weak correlation between sample quality and test log-likelihood of the model has been motivated in [178]. Surprisingly, OC-SVM outperforms most deep learning based models in this setting.

On the contrary, CIFAR10 represents a much more significant challenge, as testified by the low performances of most models, possibly due to the poor image resolution and visual clutter between classes. Specifically, we observe that our proposal is the only model outperforming a simple KDE baseline; however, this finding should be put into perspective by considering the nature of non-parametric estimators. Indeed, non-parametric models are allowed to access the whole training set for the evaluation of each sample. Consequently, despite they benefit large sample sets in terms of density modeling, they lead into an unfeasible inference as the dataset grows in size.

The possible reasons behind the difference in performance w.r.t. DAE are twofold. Firstly, DAE can recognize novel samples solely based on the reconstruction error, hence relying on its memorization capabilities, whereas our proposal also considers the likelihood of their representations under the learned prior, thus exploiting surprisal as well. Secondly, by minimizing the differential entropy of the latent distribution, our proposal increases the discriminative capability of the reconstruction. Intuitively, this last statement can be motivated observing that novelty samples are forced to reside in a high probability region of the latent space, the latter bounded to solely capture unsurprising factors of variation arising from the training set. On the other hand, the gap w.r.t. VAE suggests that, for the task at hand, a more flexible autoregressive prior should be preferred over the isotropic multivariate Gaussian. On this last point, VAE seeks representations whose average surprisal converges to a fixed and expected value (i.e., the differential entropy of its prior), whereas our solution minimizes such quantity within its MLE objective. This flexibility allows modulating the richness of the latent representation vs. the reconstructing capability of the model. On the contrary, in VAEs, the fixed prior acts as a blind regularizer, potentially leading to

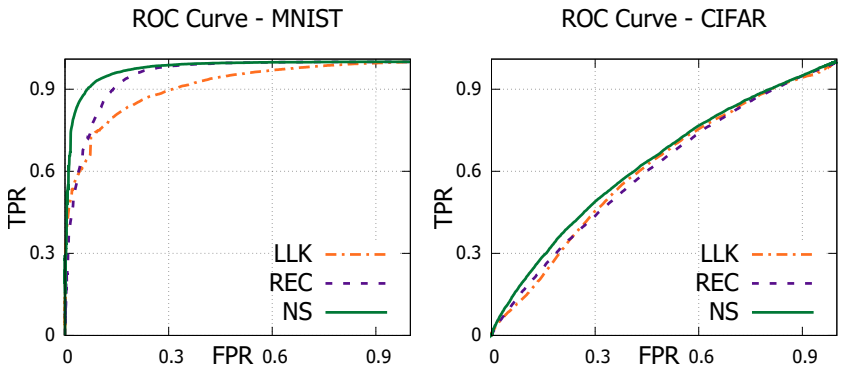


Figure 4.7: ROC curves delivered by different scoring strategies on MNIST and CIFAR-10 test sets. Each curve is an interpolation over the ten classes.

over-smooth representations; this aspect is also appreciable when sampling from the model, as will be shown in Sec 4.2.5.2. Fig. 4.7 reports an ablation study questioning the loss functions aggregation presented in Eq. 4.10. The figure illustrates ROC curves under three different novelty scores: i) the log-likelihood term, ii) the reconstruction term, and iii) the proposed scheme that accounts for both. As highlighted in the picture, accounting for both memorization and surprisal aspects is advantageous in each dataset. Additional evidence of this behavior will be illustrated in Sec. 4.2.5.1.
A qualitative illustration of the behavior of the trained one-class models, in terms of images yielding low and high novelty scores, is reported in Fig. 4.8 and 4.9 for MNIST and CIFAR-10 respectively.

4.2.2 Video anomaly detection

In video surveillance contexts, novelty is often considered in terms of abnormal human behavior. Thus, we evaluate our proposal against state-of-the-art anomaly detection models. For this purpose, we considered two standard benchmarks in literature, namely UCSD Ped2 [26] and ShanghaiTech [112]. Despite the differences in the number of videos and their resolution, they both contain anomalies that typically arise in surveillance scenarios (e.g., vehicles in pedestrian walkways, pick-pocketing, brawling). For UCSD Ped, we preprocessed input clips of 16 frames to extract smaller patches (details about this preprocessing are re-

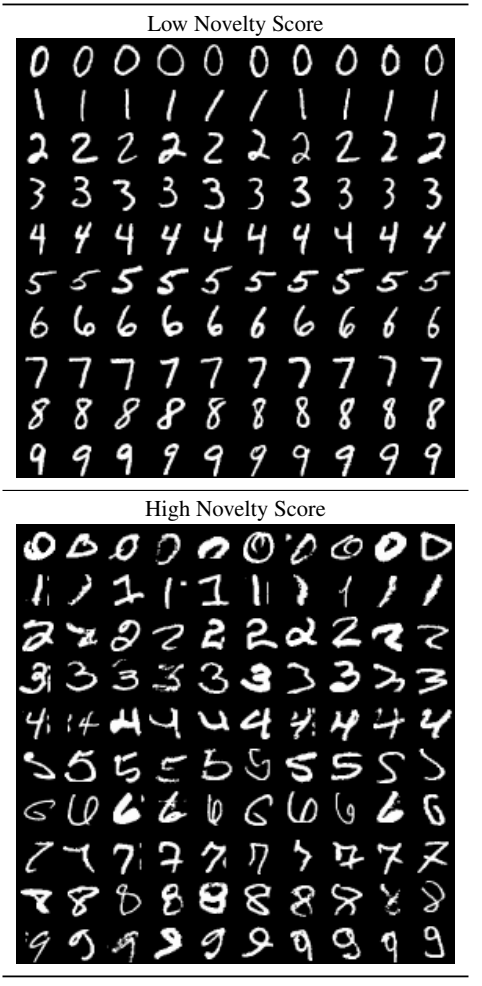


Figure 4.8: For each class in the MNIST test set, we report images assigned to a low novelty score (i.e. more normal, top image) and a high novelty score (i.e. more novel, top image) by our model. Images considered "normal" clearly showcase a higher level of regularity.

Figure 4.9: The same illustration as Fig. 4.8, but created from the CIFAR-10 test set. As for the MNIST dataset, images considered "normal" are more regular in terms of object size (airplanes, boats), context (deer) and orientation (horses).

ported in Tab. 4.1) and perturbed such inputs with random Gaussian noise with $\sigma = 0.025$. We compute the novelty score of each input clip as the mean novelty

	UCSD Ped2	ShanghaiTech
MPPCA [79]	0.693	-
MPPC+SFA [116]	0.613	-
MDT [116]	0.829	-
ConvAE [59]	0.850	0.609
ConvLSTM-AE [113]	0.881	-
Unmasking [70]	0.822	-
Hinami *et al.* [63]	0.922	-
TSC [112]	0.910	0.679
Stacked RNN [112]	0.922	0.680
FFP [105]	0.935	-
FFP+MC [105]	**0.954**	**0.728**
Ours	**0.954**	**0.725**

Table 4.3: AUROC performances of our model w.r.t. state-of-the-art competitors.

score among all patches. Concerning ShanghaiTech, we removed the dependency on the scenario by estimating the foreground for each frame of a clip with a standard MOG-based approach and removing the background. We fed the model with 16-frames clips, but ground-truth anomalies are labeled at frame level. In order to recover the novelty score of each frame, we compute the mean score of all clips in which it appears. We then merge the two terms of the loss function following the same strategy illustrated in Eq. 4.10, computing however normalization coefficients in a per-sequence basis, following the standard approach in the anomaly detection literature. The scores for each sequence are then concatenated to compute the overall AUROC of the model. Additionally, we envision localization strategies for both datasets. To this aim, for UCSD, we denote a patch exhibiting the highest novelty score in a frame as anomalous. Differently, in ShanghaiTech, we adopt a sliding-window approach [206]: as expected, when occluding the source of the anomaly with a rectangular patch, the novelty score drops significantly.

Tab. 4.3 reports results in comparison with prior works, whilst Fig. 4.10 illustrates qualitative assessments regarding the novelty score and localization capabilities. Despite a more general formulation, our proposal scores on-par with the current state-of-the-art solutions specifically designed for video applications and taking advantage of optical flow estimation and motion constraints. Indeed, in the ab-

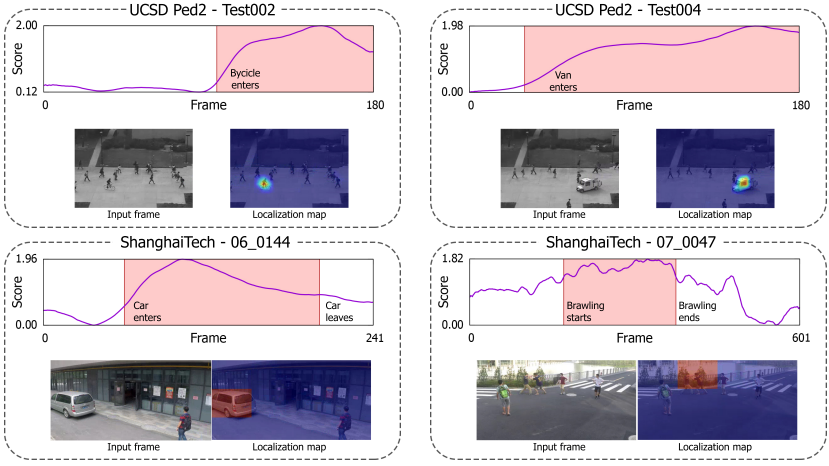


Figure 4.10: Novelty scores and localizations maps for samples drawn from UCSD Ped2 and ShanghaiTech. For each example, we report the trend of the assessed score, highlighting with a different color the time range in which an anomalous subject comes into the scene.

sence of such hypotheses (FFP entry in Tab. 4.3), our method outperforms future frame prediction on UCSD Ped2.

4.2.3 Model Analysis

4.2.3.1 CIFAR-10 with semantic features

We investigate the behavior of our model in the presence of different assumptions regarding the expected nature of novel samples. We expect that, as the correctness of such assumptions increases, novelty detection performances will scale accordingly. Such a trait is particularly desirable for applications in which prior beliefs about novel examples can be envisioned. To this end, we leverage the CIFAR-10 benchmark described in Sec. 4.2.1 and change the type of information provided as input. Specifically, instead of raw images, we feed our model with semantic representations extracted by ResNet-50 [60], either pre-trained on Imagenet (i.e., assume semantic novelty) or CIFAR-10 itself (i.e., assume data-specific novelty). The two models achieved respectively 79.26 and 95.4 top-1

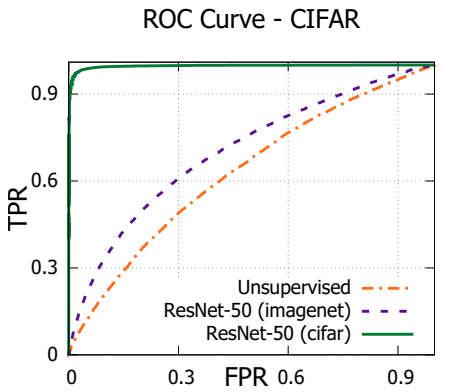


Figure 4.11: CIFAR-10 ROC curves with semantic input vectors. Each curve is an interpolation among the ten classes.

classification accuracies on the respective test sets. Even though this procedure is to be considered unfair in novelty detection, it serves as a sanity check delivering the upper-bound performances our model can achieve when applied to even better features. To deal with dense inputs, we employ a fully connected autoencoder and MFC layers within the estimation network.

Fig. 4.11 illustrates the resulting ROC curves, where semantic descriptors improve AUROC w.r.t. raw image inputs (entry "Unsupervised"). Such results suggest that our model profitably takes advantage of the separation between normal and abnormal input representations and scales accordingly, even up to optimal performances for the task under consideration. Nevertheless, it is interesting to note how different degrees of supervision deliver significantly different performances. As expected, dataset-specific supervision increases the AUROC from 0.64 up to 0.99 (a perfect score). Surprisingly, semantic feature vectors trained on Imagenet (which contains all CIFAR classes) provide a much lower boost, yielding an AUROC of 0.72. Such result suggests that, even in the rare cases where the semantic of novelty can be known in advance, its contribution has a limited impact in modeling the normality, mostly because novelty can depend on other cues (e.g., low-level statistics).

CIFAR-10		UCSD Ped2	
$\text{LSTM}_{[100]}$	0.623	$\text{LSTM}_{[100]}$	0.849
$\text{LSTM}_{[32,32,32,32,100]}$	0.622	$\text{LSTM}_{[4,4,4,4,100]}$	0.845
$\text{MFC}_{[100]}$	0.625	$\text{MSC}_{[100]}$	0.849
$\text{MFC}_{[32,32,32,32,100]}$	**0.641**	$\text{MSC}_{[4,4,4,4,100]}$	**0.954**

Table 4.4: Comparison of different architectures for the autoregressive density estimation in feature space. We indicate with $\text{LSTM}_{[F_1,F_2,...,F_N]}$ - same goes for MFC and MSC - the output shape for each of the N layers composing the estimator. Results are reported in terms of test AUROC.

4.2.3.2 Autoregression via recurrent layers

To measure the contribution of the proposed MFC and MSC layers described in Sec. 4.1, we test on CIFAR-10 and UCSD Ped2, alternative solutions for the autoregressive density estimator. Specifically, we investigate recurrent networks, as they represent the most natural alternative featuring autoregressive properties. We benchmark the proposed building blocks against an estimator composed of LSTM layers, which is designed to sequentially observe latent symbols $\mathbf{z}_{<i}$ and output the CPD of z_i as the hidden state of the last layer. We test MFC, MSC and LSTM in single-layer and multi-layer settings, and report all outcomes in Tab. 4.4.

It emerges that, even though our solutions perform similarly to the recurrent baseline when employed in a shallow setting, they significantly take advantage of their depth when stacked in consecutive layers. MFC and MSC, indeed, employ disentangled parametrizations for each output CPD. This property is equivalent to the adoption of a specialized estimator network for each z_i, thus increasing the proficiency in modeling the density of its designated CPD. On the contrary, LSTM networks embed all the history (i.e., the observed symbols) in their memory cells, but manipulate each input of the sequence through the same weight matrices. In such a regime, the recurrent module needs to learn parameters shared among symbols, losing specialization and eroding its modeling capabilities.

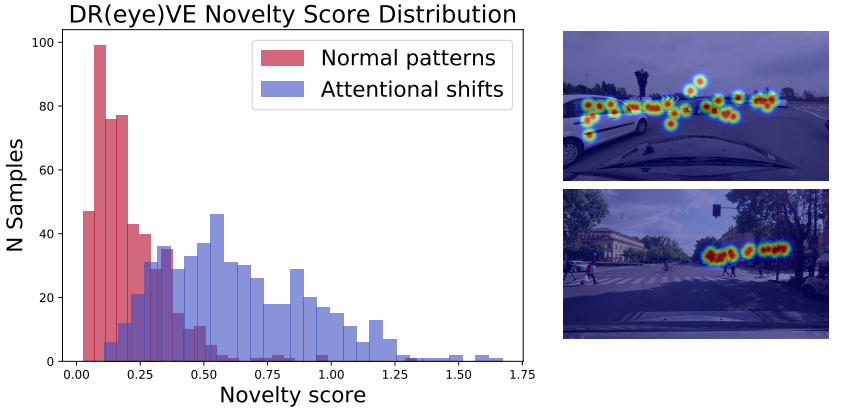


Figure 4.12: Left, the distribution of novelty scores assigned to normal patterns against attentional shifts labeled within the DR(eye)VE dataset. Right, DR(eye)VE clips yielding the highest novelty score (i.e., clips in which the attentional pattern shifts from the expected behavior). Interestingly, they depict some peculiar situations such as waiting for the traffic light or approaching a roundabout.

4.2.4 Novelty in cognitive temporal processes

As a potential application of our proposal, we investigate its capability in modeling human attentional behavior. To this end, we employ the DR(eye)VE dataset, introduced in Sec. 3.1 for the prediction of focus of attention in driving contexts. It features 74 driving videos where frame-wise fixation maps are provided, highlighting the region of the scene attended by the driver. In order to capture the dynamics of attentional patterns, we purposely discard the visual content of the scene and optimize our model on clips of fixation maps, randomly extracted from the training set. After training, we rely on the novelty score of each clip as a proxy for the uncommonness of an attentional pattern. Moreover, since the dataset features annotations of peculiar and unfrequent patterns (such as distractions, recording errors), we can measure the correlation of the captured novelty w.r.t. those. In terms of AUROC, our model scores 0.926, highlighting that novelty can arise from unexpected behaviors of the driver, such as distractions or other shifts in attention. Fig. 4.12 reports the different distribution of novelty scores for ordinary and peculiar events.

	LLK	REC	NS
MNIST	0.926	0.949	**0.975**
CIFAR-10	0.627	0.603	**0.641**
UCSD Ped2	0.933	0.909	**0.954**
ShanghaiTech	0.695	**0.726**	0.725
DR(eye)VE	0.917	0.863	**0.926**

Table 4.5: For each setting, AUROC performances under three different novelty scores: i) the log-likelihood term (LLK), ii) the reconstruction term (REC), and iii) the proposed scheme accounting for both (NS).

4.2.5 Additional experiments and analysis

4.2.5.1 Contribution to the novelty score

In this section, we stress how significant is the presence of both terms for obtaining a highly discriminative novelty score (NS, Eq. 4.10): namely the reconstruction error (REC), modelinge the memory capabilities, and the log-likelihood term (LLK), capturing the surprisal inducted from latent representations. Aiming to reinforce this latter point, just briefly illustrated in Fig. 4.7 so far, we report in Tab. 4.5 performances - expressed in AUROC - delivered by different scoring strategies on each experiment. Except for ShanghaiTech, we systematically observe a reward when accounting for both aspects. Furthermore, for MNIST and CIFAR-10, we find particularly interesting the gap in performance arising from our reconstruction error w.r.t. the one arising from the denoising autoencoder (DAE) variants (0.942 and 0.590 for the two datasets respectively, as reported in Tab. 4.2). In this respect, we gather new evidence supporting that surprisal minimization acts as a novelty-oriented regularizer for the overall architecture, as it improves the discriminative capability of the reconstruction (as already conjectured in Sec. 4.2.1).

4.2.5.2 On the relations to Variational Autoencoders

Our framework yields some similarities with Variational Autoencoders (VAEs) [83]. Indeed, they both approximate the integral of Eq. 4.1 through the minimization of the reconstruction error under a regularization constraint involving a prior distribution on latent vectors. However, it is worth noting several fundamental dis-

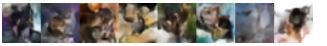
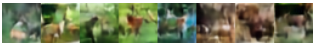
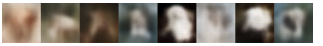
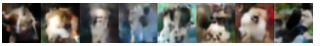
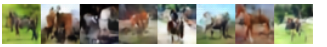
FID	VAE Samples	Our Samples	FID
149.72			72.96
172.02			72.53
181.56			76.27
188.37			67.33
202.06			68.33
207.47			73.92
186.48			62.26
220.79			64.38
164.36			52.53
204.84			67.17

Figure 4.13: For all CIFAR-10 classes (organized in different rows), we report images sampled from VAEs (left) and the proposed autoencoders with autoregressive priors. As can be seen, our samples visually exhibit fine-grained details and sharpness, differently from the heavily blurred ones coming from VAEs. Finally, the over-regularization arising from VAE is confirmed when looking at FID scores (at the extremes of the figure, the lower, the better).

tinctions. Firstly, our model does not provide an explicit strategy to sample from the posterior distribution, thus resulting in a deterministic mapping from the input to the hidden representation. Secondly, while VAE specifies an explicit and adamant form for modeling the prior $p(\mathbf{z})$, in our formulation its landscape is free from any assumption and directly learnable as a result of the estimator's autoregressive nature. On this point, our proposal leads to two beneficial aspects. First, as the VAE forces the codes' distribution to match the prior, their differential entropy converges to be the same as the prior. This behavior results in approximately stationary entropies across different settings (appreciable in Fig. 4.3, where we discuss the intuition behind the entropy minimization within a novelty detection task). Secondly, the employment of a too simplistic prior may lead to over-regularized representations, whereas our proposal is less prone to such risk. Empirical evidence of such behavior can also be appreciated in Fig. 4.13, where

we draw new samples from VAE and our model, both of which has been trained on CIFAR-10. All settings being equal, our hallucinations are visually much more realistic than the ones coming from VAEs, the latter leading to over-smooth shapes and lacking any details, as further confirmed by the substantial differences in Fréchet Inception Distance (FID) scores [62].

The perspective representing our model as a VAE with a more sophisticated prior might suggest it should benefit from the employment of even more advanced design choices, inspired by recent progress in the density estimation community (e.g., about the approximate posterior or the conditional likelihood). Such a consideration arises from a very reasonable assumption: the better the density model, the higher its novelty detection performances. However, in our experience, this matter seems more intricate than it looks, as it also emerges from recent works [127, 32]. In Tab. 4.6 top, we experiment with several solutions for improving the VAE baseline on one-class novelty detection. Specifically, we adopt (i) Inverse Autoregressive Flow (IAF) [82] as a powerful approximate posterior, (ii) an Autoregressive Conditional Likelihood (ACL) and (iii) a more flexible prior, as the one induced by a mixture of posteriors (Vamp Prior) [179]. Notably, despite being more suitable likelihood models, none of those clearly outperforms the standard VAE when applied to novelty detection. ACL, similarly to Pix-CNN (Tab. 4.2), yields instead a catastrophic behavior for anomaly detection, whereas improving the prior seems the most encouraging direction.

Nevertheless, we believe the influence of approximate posteriors and conditional likelihoods within novelty detection is yet to be understood. As a first step, we explore two variants of our architecture (Tab. 4.6 bottom), featuring either i) a Gaussian Approximate Posterior (GAP) or ii) a Mean-Field Gaussian (MFG) as conditional likelihood (in lieu of the Mean Squared Error).

4.2.5.3 On the causal structure of representations

We now investigate the capability of our encoder to produce representations that respect the autoregressive causal structure imposed by the LLK loss (mentioned in Sec. 4.1). To this aim, we extract representations out of the ten models trained on MNIST digits and fit their distribution using a structured density estimator. Specifically, we employ Bayesian Networks (BNs) with different autoregressive structures. In this respect, each BN is modeled with Linear Gaussians [87], s.t.

	MNIST	CIFAR10
VAE	**0.969**	0.586
VAE + IAF	0.955	0.585
VAE + ACL	0.580	0.530
VAE + Vamp Prior	0.968	**0.600**
ours	**0.975**	0.641
ours + GAP	0.959	0.630
ours + MFG	0.934	**0.651**

Table 4.6: Average AUROC results on MNIST and CIFAR10 for different VAE alternatives.

each CPD $p(z_i|Pa(z_i))$ with $i = 1, 2, \ldots, d$ is given by:

$$p(z_i|Pa(z_i)) = \mathcal{N}(z_i \mid w_0^{(i)} + \sum_{z_j \in Pa(z_i)} w_j^{(i)} z_j, \sigma_i^2), \tag{4.12}$$

where each $w_j^{(i)}$, σ_i^2 are learnable parameters. We indicate with $Pa(z_i)$ the parent variables of z_i in the BN. The previous equation holds for all nodes, except for the root one, which is modeled through a Gaussian distribution. Concerning the BN structure, we test:

- Autoregressive order: the BN structure follows the autoregressive order imposed during training, namely $Pa(z_i) = \{z_j \mid j = 1, 2, \ldots, i-1\}$
- Random order: the BN structure follows a random autoregressive order.
- Inverse order: the BN structure follows an autoregressive order which is the inverse with respect to the one imposed during training, namely $Pa(z_i) = \{z_j \mid j = i+1, i+2, \ldots, d\}$

It is worth noting that, as the three structures exhibit the same number of edges and independent parameters, the difference in their fitting capabilities is only due to the causal order imposed over variables.
Fig. 4.14 reports the sample training log-likelihood of all BN models. Remarkably, the autoregressive order delivers a better fit, supporting the capability of the encoder network to extract features with learned autoregressive properties.

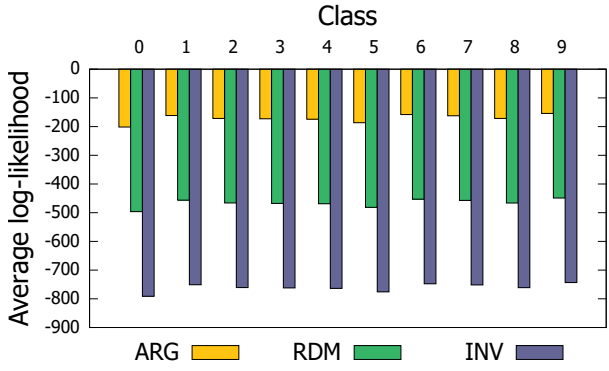


Figure 4.14: Sample training log-likelihood of a Bayesian Network modeling the distribution of latent codes produced by the encoder of our model trained on MNIST digits. When the BN structure resembles the autoregressive order imposed during training, a much higher likelihood is achieved. This behavior is consistent in all classes and supports the capability of the encoder to produce codes that respect a pre-imposed autoregressive structure.

Moreover, to show that this result is not due to overfitting or other lurking behaviors, we report in Tab. 4.7 log-likelihoods for training, validation and test set.

4.2.5.4 On the entropy minimization

To provide an additional grasp about the role of the representation's entropy minimization, we focus on a single MNIST digit (class 7) and report in Fig. 4.15 some randomly sampled reconstructions from the training set. Such reconstructions are learned under three different regularization regimes, represented by different weights on the log-likelihood objective (λ, Eq. 4.4). As shown in Fig. 4.15, higher degrees of regularization (i.e., stricter constraints on entropy) deliver near mode-collapsed reconstructions, losing sharp variations in favor of capturing fewer prototypes for the input distribution.

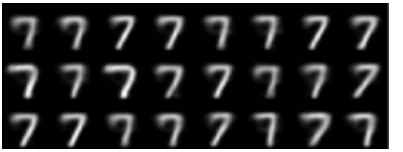
Loss weight	Reconstructions
$\lambda = 0.01$	
$\lambda = 1$	
$\lambda = 100$	

Figure 4.15: MNIST reconstructions delivered by different values of λ, the latter controlling the impact of the differential entropy minimization.

		Classes									
		0	1	2	3	4	5	6	7	8	9
ARG	Train	-201.60	-161.60	-171.43	-172.73	-174.17	-186.48	-158.22	-162.37	-171.65	-154.11
	Val	-200.96	-160.38	-170.10	-172.29	-173.85	-185.25	-157.22	-162.20	-171.42	-154.02
	Test	-200.89	-159.73	-169.64	-170.75	-172.40	-184.27	-157.74	-161.65	-170.10	-152.70
RDM	Train	-496.33	-456.34	-466.16	-467.47	-468.90	-481.21	-452.95	-457.10	-466.39	-448.84
	Val	-495.69	-455.11	-464.83	-467.02	-468.58	-479.98	-451.95	-456.93	-466.15	-448.75
	Test	-495.62	-454.47	-464.37	-465.48	-467.13	-479.00	-452.48	-456.38	-464.83	-447.43
INV	Train	-791.06	-751.07	-760.89	-762.20	-763.63	-775.94	-747.68	-751.83	-761.12	-743.57
	Val	-790.42	-749.84	-759.56	-761.75	-763.31	-774.71	-746.68	-751.66	-760.88	-743.48
	Test	-790.35	-749.20	-759.11	-760.22	-761.86	-773.73	-747.21	-751.12	-759.56	-742.16

Table 4.7: Sample log-likelihood obtained by different BN structures when fitting MNIST representations. Each BN is trained on latent codes computed from the training set of a single class, following either the autoregression order (ARG), a random order (RDM) or the order inverse to autoregression (INV). We report the log-likelihood also on the validation and test set. For train-val-test split, see Sec 4.2.1. Only "normal" test samples are used in this evaluation.

4.2.5.5 On the complexity of autoregressive layers

In this section, we briefly discuss the complexity of Masked Fully Connected (MFC) and Masked Stacked Convolution (MSC) layers (Fig. 4.5 and 4.6)*: adhering to the notation introduced in Sec. 4.1, MFC exhibits $\frac{d^2+d}{2} \cdot ci \cdot co + d \cdot co$ trainable parameters and a computational complexity $\mathcal{O}(d^2 \cdot ci \cdot co)$. MSC, instead, features $\frac{3d^2+d}{2} ci \cdot co + d \cdot c_o$ free parameters and a time complexity $\mathcal{O}(d^2 \cdot ci \cdot co \cdot t)$.

4.2.5.6 Additional figures

We show in Fig. 4.16 other qualitative evidence of the behavior of our model in video anomaly detection settings, namely UCSD Ped2 and ShanghaiTech.

*We refer to the type 'B' of both layers, since it is an upper bound to the type 'A'

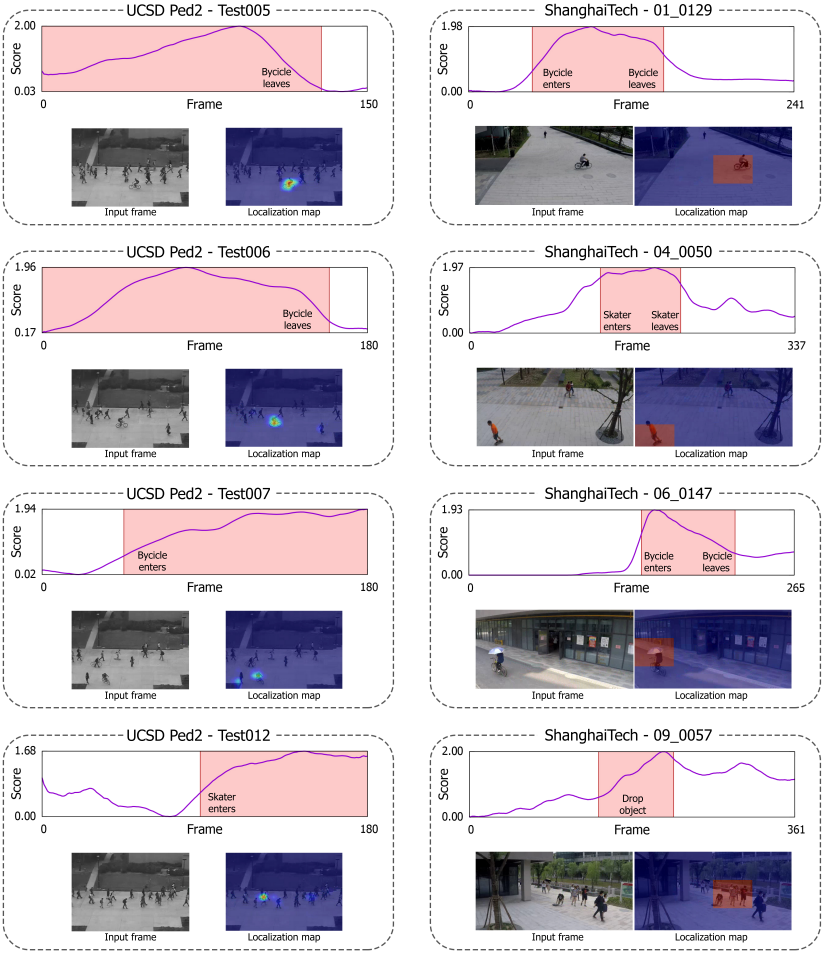


Figure 4.16: Novelty scores and localizations maps for several test clips from UCSD Ped2 (left) and ShanghaiTech (right).

Chapter 5

Conditional Channel Gated Networks for Task-Aware Continual Learning

Davide Abati[1*], Jakub Tomczak[2], Tijmen Blankevoort[2], Simone Calderara[1], Rita Cucchiara[1], Babak Ehteshami Bejnordi[2]
[1]University of Modena and Reggio Emilia
[2]Qualcomm AI Research, Qualcomm Technologies Netherlands B.V.[†]

Current machine learning and deep learning models are typically trained offline, by sampling examples independently from the distribution they are expected to deal with at test time. However, when trained online in real-world settings, models may encounter multiple different tasks as a sequential stream of activities, without having any knowledge about their relationship or duration in time. Such challenges typically arise in robotics [4], reinforcement learning [152], computer vision systems [134] and many more (cf. Chapter 4 in [29]). In such scenarios, deep learning models suffer from *catastrophic forgetting* [122, 45], meaning they discard previously acquired knowledge to fit the current observations. The reason behind this behavior is that, while learning the current task, the model overwrites the parameters that were critical for previous tasks.

[*]Work done while first author was intern at Qualcomm AI Research.
[†]Qualcomm AI Research is an initiative of Qualcomm Technologies Inc.

Continual learning research (also called *lifelong* or *incremental* learning) tackles the above mentioned issues [29]. The typical setting considered in the literature is that of a model learning disjoint classification problems one-by-one. Depending on the application requirements, the task for which the current input should be analyzed may or may not be known. The majority of the methods in the literature assume that the label of the task is provided during inference. Such a continual learning setting is generally referred to as task-incremental. In many real-world applications, such as classification and anomaly detection systems, a model can seamlessly instantiate a new task whenever novel classes emerge from the training stream. However, once deployed in the wild, it has to process inputs without knowing in which training task similar observations were encountered. Such a setting, in which task labels are available only during training, is known as class-incremental [188]. Existing methods employ different strategies to mitigate catastrophic forgetting, such as memory buffers [147, 108], knowledge distillation [101], synaptic consolidation [85] and parameters masking [118, 165]. However, recent evidence has shown that existing solutions fail, even for simple datasets, whenever task labels are not available at test time [188]. In this chapter, we introduce a solution based on conditional-computing to tackle both task-incremental and class-incremental learning problems. Specifically, our framework relies on separate task-specific classification heads (*multi-head* architecture), and it employs channel-gating [30, 14] in every layer of the (shared) feature extractor.

To this aim, we introduce task-dedicated gating modules that dynamically select which filters to apply conditioned on the input feature map. Along with a sparsity objective encouraging the use of fewer units, this strategy enables per-sample model selection and can be easily queried for information about which weights are essential for the current task. Those weights are frozen when learning new tasks, but gating modules can dynamically select to either use or discard them. Contrarily, units that are never used by previous tasks are reinitialized and made available for acquiring novel concepts. This procedure prevents any forgetting of past tasks and allows considerable computational savings in the forward propagation.

Moreover, we obviate the need for a task label during inference by introducing a task classifier selecting which classification head should be queried for the class prediction. We train the task classifier alongside the classification heads under the same incremental learning constraints. To mitigate forgetting on the task classification side, we rely on example replay from either episodic or generative memories. In both cases, we show the benefits of performing rehearsal at a task-

level, as opposed to previous replay methods that operate at a class-level [147, 27]. To the best of our knowledge, this is the first work that carries out supervised task prediction in a class-incremental learning setting.
We perform extensive experiments on four datasets of increasing difficulty, both in the presence and absence of a task oracle at test time. Our results show that, whenever task labels are available, our model effectively prevents the forgetting problem, and performs similarly to or better than state-of-the-art solutions. In the task agnostic setting, we consistently outperform competing methods.

5.1 Model

5.1.1 Problem setting and objective

We are given a parametric model, i.e., a neural network, called a *backbone* or *learner* network, which is exposed to a sequence of N tasks to be learned, $\mathcal{T} = \{T_1, \dots, T_N\}$. Each task T_i takes the form of a classification problem, $T_i = \{\mathbf{x}_j, y_j\}_{j=1}^{n_i}$, where $\mathbf{x}_j \in \mathbb{R}^m$ and $y_j \in \{1, \dots, C_i\}$.
In a task-incremental setting, the following objective has to be optimized:

$$\max_{\theta} \quad \mathbb{E}_{\mathbf{t}\sim\mathcal{T}} \left[\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim T_{\mathbf{t}}} \left[\log p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{t}) \right] \right], \tag{5.1}$$

where θ identifies the parametrization of the learner network, and $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{t}$ are random variables associated with the observation, the label and the task of each example, respectively. Such a maximization problem is subject to the continual learning constraints: as the model observes tasks sequentially, the outer expectation in Eq. 5.1 is troublesome to compute or approximate. Notably, this setting requires the assumption that the identity of the task each example belongs to is known at both training and test stages. Such information can be exploited in practice to isolate relevant output units of the classifier, preventing the competition between classes belonging to different tasks through the same softmax layer (*multi-head*).
Class-incremental settings require solving the following optimization problem:

$$\max_{\theta} \quad \mathbb{E}_{\mathbf{t}\sim\mathcal{T}} \left[\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim T_{\mathbf{t}}} \left[\log p_\theta(\mathbf{y}|\mathbf{x}) \right] \right]. \tag{5.2}$$

Here, the absence of task conditioning prevents any form of task-aware reasoning in the model. This setting requires to merge the output units into a single classifier (*single-head*) in which classes from different tasks compete with each other, often

resulting in more severe forgetting [188]. Although the model could learn based on task information, this information is not available during inference.
To deal with observations from unknown tasks, while retaining advantages of multi-head settings, we propose to jointly optimize for class as well as task prediction, as follows:

$$\begin{aligned}\max_{\theta} \quad & \mathbb{E}_{\mathbf{t}\sim\mathcal{T}}\left[\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim T_{\mathbf{t}}}\left[\log p_\theta(\mathbf{y},\mathbf{t}|\mathbf{x})\right]\right] = \\ & \mathbb{E}_{\mathbf{t}\sim\mathcal{T}}\left[\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim T_{\mathbf{t}}}\left[\log p_\theta(\mathbf{y}|\mathbf{x},\mathbf{t}) + \log p_\theta(\mathbf{t}|\mathbf{x})\right]\right]. \end{aligned} \tag{5.3}$$

Eq. 5.3 describes a twofold objective. On the one hand, the term $\log p(\mathbf{y}|\mathbf{x},\mathbf{t})$ is responsible for the *class classification given the task*, and resembles the multi-head objective in Eq. 5.1. On the other hand, the term $\log p(\mathbf{t}|\mathbf{x})$ aims at *predicting the task* from the observation. This prediction relies on a task classifier, which is trained incrementally in a single-head fashion. Notably, the proposed objective in Eq. 5.3 shifts the single-head complexities from a class prediction to a task prediction level, with the following benefits:

- given the task label, there is no drop in class prediction accuracy;
- classes from different tasks never compete with each other, neither during training nor during test;
- the challenging single-head prediction step is shifted from class to task level; as tasks and classes form a two-level hierarchy, the prediction of the former is arguably easier (as it acts at a coarser semantic level).

5.1.2 Multi-head learning of class labels

In this section, we introduce the conditional computation model we used in our work. Fig. 5.1 illustrates the gating mechanism used in our framework. We limit the discussion of the gating mechanism to the case of convolutional layers, as it also applies to other parametrized mappings such as fully connected layers or residual blocks. Consider $\mathbf{h}^l \in \mathbb{R}^{c^l_{in},h,w}$ and $\mathbf{h}^{l+1} \in \mathbb{R}^{c^l_{out},h',w'}$ to be the input and output feature maps of the l-th convolutional layer respectively. Instead of $\mathbf{h}^{l+1}$, we will forward to the following layer a sparse feature map $\hat{\mathbf{h}}^{l+1}$, obtained by pruning uninformative channels. During the training of task t, the decision regarding which channels have to be activated is delegated to a gating module G^l_t, that is conditioned on the input feature map $\mathbf{h}^l$:

$$\hat{\mathbf{h}}^{l+1} = G^l_t(\mathbf{h}^l) \odot \mathbf{h}^{l+1}, \tag{5.4}$$

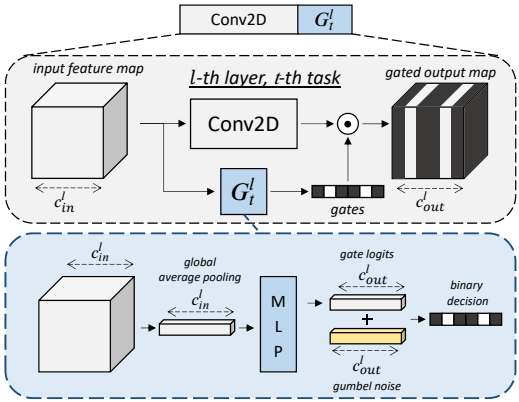


Figure 5.1: The proposed gating scheme for a convolution layer. Depending on the input feature map, the gating module G_t^l decides which kernels should be used.

where $G_t^l(\mathbf{h}^l) = [g_1^l, \ldots, g_{c_{out}^l}^l]$, $g_i^l \in \{0, 1\}$, and $\odot$ refers to channel-wise multiplication. To be compliant with the incremental setting, we instantiate a new gating module each time the model observes examples from a new task. However, each module is designed as a light-weight network with negligible computation costs and number of parameters. Specifically, each gating module comprises a Multi-Layer Perceptron (MLP) with a single hidden layer featuring 16 units, followed by a batch normalization layer [68] and a ReLU activation. A final linear map provides log-probabilities for each output channel of the convolution. Back-propagating gradients through the gates is challenging, as non-differentiable thresholds are employed to take binary on/off decisions. Therefore, we rely on the Gumbel-Softmax sampling [74, 115], and get a biased estimate of the gradient utilizing the straight-through estimator [15]. Specifically, we employ the hard threshold in the forward pass (zero-centered) and the sigmoid function in the backward pass (with temperature $\tau = 2/3$).
Moreover, we penalize the number of active convolutional kernels with the sparsity objective:

$$\mathcal{L}_{sparse} = \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim T_\mathbf{t}} \left[\frac{\lambda_s}{L} \sum_{l=1}^{L} \frac{\|G_t^l(\mathbf{h}^l)\|_1}{c_{out}^l} \right], \tag{5.5}$$

where L is the total number of gated layers, and λ_s is a coefficient controlling the

level of sparsity. The sparsity objective instructs each gating module to select a minimal set of kernels, allowing us to conserve filters for the optimization of future tasks. Moreover, it allows us to effectively adapt the capacity of the allocated network depending on the difficulty of the task and the observation at hand. Such a data-driven model selection contrasts with other continual learning strategies that employ fixed ratios for model growing [157] or weight pruning [118].
At the end of the optimization for task t, we compute a relevance score r_k^l for each unit in the l-th layer by estimating the firing probability of their gates on a validation set $T_{\mathbf{t}}^{val}$:

$$r_k^{l,t} = \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim T_{\mathbf{t}}^{val}}[p(\mathbb{I}[g_k^l = 1])], \tag{5.6}$$

where $\mathbb{I}[\cdot]$ is an indicator function, and $p(\cdot)$ denotes a probability distribution. By thresholding such scores, we obtain two sets of kernels. On the one hand, we *freeze relevant kernels* for the task t, so that they will be available but not updatable during future tasks. On the other hand, we *re-initialize non-relevant kernels*, and leave them learnable by subsequent tasks. In all our experiments, we use a threshold equal to 0, which prevents any forgetting at the expense of a reduced model capacity left for future tasks.
Note that within this framework, it is trivial to monitor the number of learnable units left in each layer. As such, if the capacity of the backbone model saturates, we can quickly grow the network to digest new tasks. However, because the gating modules of new tasks can dynamically choose to use previously learned filters (if relevant for their input), learning of new tasks generally requires less learnable units. In practice, we never experienced the saturation of the backbone model for learning new tasks. Apart from that, because of our conditional channel-gated network design, increasing the model capacity for future tasks will have minimal effects on the computation cost at inference. We will provide an analysis of the computation cost of our network in Sec. 5.2.5.

5.1.3 Single-head learning of task labels

The gating scheme presented in Sec. 5.1.2 allows the immediate identification of important kernels for each past task. However, it cannot be applied in the task-agnostic setting as is, since it requires the knowledge about which gating module G_x^l has to be applied for layer l, where $x \in \{1, \dots, t\}$ represents the unknown task. Our solution is to employ all gating modules $[G_1^l, \dots, G_t^l]$, and to propagate all gated layer outputs $[\hat{\mathbf{h}}_1^{l+1}, \dots, \hat{\mathbf{h}}_t^{l+1}]$ forward. In turn, the following

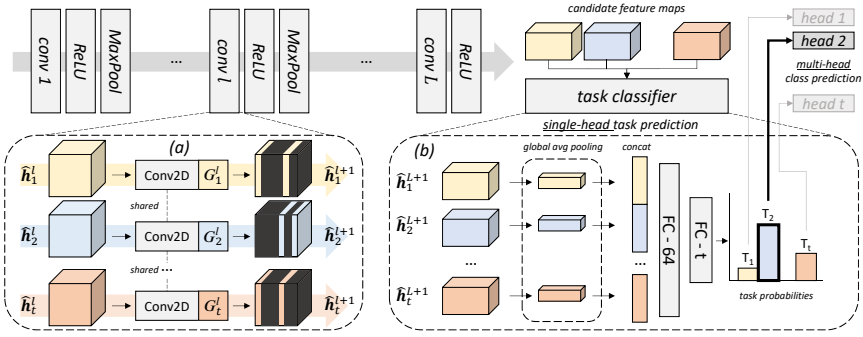


Figure 5.2: Illustration of the task prediction mechanism for a generic backbone architecture. First (block 'a'), the l-th convolutional layer is fed with multiple gated feature maps, each of which is relevant for a specific task. Every feature map is then convolved with kernels selected by the corresponding gating module G_x^l, and forwarded to the next module. At the end of the network the task classifier (block 'b') takes as input candidate feature maps and decides which task to solve.

layer $l+1$ receives the list of gated outputs from layer l, applies its gating modules $[G_1^{l+1}, \ldots, G_t^{l+1}]$ and yields the list of outputs $[\hat{\mathbf{h}}_1^{l+2}, \ldots, \hat{\mathbf{h}}_t^{l+2}]$. This mechanism generates parallel streams of computation in the network, sharing the same layers but selecting different sets of units to activate for each of them (Fig. 5.2). Despite the fact that the number of parallel streams grows with the number of tasks, we found our solution to be computationally cheaper than the backbone network (see Sec. 5.2.5). This is because of the gating modules which select a limited number of convolutional filters in each stream.
After the last convolutional layer, indexed by L, we are given a list of t candidate feature maps $[\hat{\mathbf{h}}_1^{L+1}, \ldots, \hat{\mathbf{h}}_t^{L+1}]$ and as many classification heads. The task classifier is fed with a concatenation of all feature maps:

$$h = \bigoplus_{i=1}^{t} [\mu(\hat{\mathbf{h}}_i^{L+1})], \tag{5.7}$$

where μ denotes the global average pooling operator over the spatial dimensions and $\bigoplus$ describes the concatenation along the feature axis. The architecture of the task classifier is based on a shallow MLP with one hidden layer featuring

64 ReLU units, followed by a softmax layer predicting the task label. We use the standard cross-entropy objective to train the task classifier. Optimization is carried out jointly with the learning of class labels at task t. Thus, the network not only learns features to discriminate the classes inside task t, but also to allow easier discrimination of input data from task t against all prior tasks.
The single-head task classifier is exposed to catastrophic forgetting. Recent papers have shown that replay-based strategies represent the most effective continual learning strategy in single-head settings [188]. Therefore, we choose to ameliorate the problem by rehearsal. In particular, we consider the following approaches.

Episodic memory. A small subset of examples from prior tasks is used to rehearse the task classifier. During the training of task t, the buffer holds C random examples from past tasks $1, \ldots, t-1$ (where C denotes a fixed capacity). Examples from the buffer and the current batch (from task t) are re-sampled so that the distribution of task labels in the rehearsal batch is uniform. At the end of task t, the data in the buffer is subsampled so that each past task holds $m = C/t$ examples. Finally, m random examples from task t are selected for storage.

Generative memory. A generative model is employed for sampling fake data from prior tasks. Specifically, we utilize Wasserstein GANs with Gradient Penalty (WGAN-GP [58]). To overcome forgetting in the sampling procedure, we use multiple generators, each of which is devoted to modeling the distribution of examples of a specific task.

In both cases, replay is only employed for rehearsing the task classifier and not the classification heads. To summarize, the complete objective of our model includes: the cross-entropy at a class level ($p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{t})$ in Eq. 5.3), the cross-entropy at a task level ($p_\theta(\mathbf{t}|\mathbf{x})$ in Eq. 5.3) and the sparsity term ($\mathcal{L}_{sparse}$ in Eq. 5.5).

5.2 Experimental results

5.2.1 Datasets, architectures and implementation details

We experiment with the following datasets:

- Split MNIST: the MNIST handwritten classification benchmark [98] is

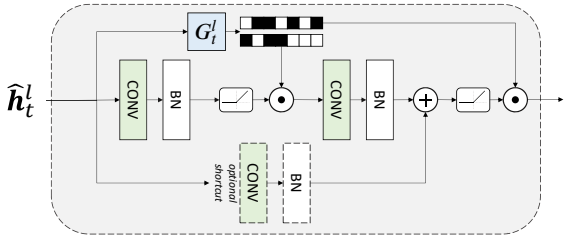


Figure 5.3: The gating scheme applied to the ResNet-18 backbone architecture. It provides two sets of gating vectors: the former is applied to the output of the first convolution, the latter after the residual connection.

split into 5 subsets of consecutive classes. This results into 5 binary classification tasks that are observed sequentially.

- Split SVHN: the same protocol applied as in Split MNIST, but employing the SVHN dataset [128].
- Split CIFAR-10: the same protocol applied as in Split MNIST, but employing the CIFAR-10 dataset [89].
- Imagenet-50 [134]: a subset of the iILSVRC-2012 dataset [40] containing 50 randomly sampled classes and 1300 images per category, split into 5 consecutive 10-way classification problems. Images are resized to a resolution of 32x32 pixels.

As for the backbone models, for the MNIST and SVHN benchmarks, we employ a three-layer CNN with 100 filters per layer and ReLU activations (*SimpleCNN* in what follows). All convolutions except for the last one are followed by a 2x2 max-pooling layer. Gating is applied after the pooling layer. A final global average pooling followed by a linear classifier yields class predictions. For the CIFAR-10 and Imagenet-50 benchmarks we employed a ResNet-18 [60] model as backbone. The gated version of a ResNet basic block is represented in Fig. 5.3. As illustrated, two independent sets of gates are applied after the first convolution and after the residual connection, respectively. All models were trained with SGD with momentum until convergence. After each task, model selection is performed for all models by monitoring the corresponding objective on a held-out set of examples from the current task (i.e., we don't rely on examples of past tasks for validation purposes). We apply the sparsity objective introduced in Sec. 5.1.2

		Split MNIST	Split SVHN
optim	batch size	256	256
	learning rate	0.01	0.01
	momentum	0.9	0.9
	lr decay	-	$[400, 600]$
	weight decay	$5e-4$	$5e-4$
	epochs per task	400	800
	grad. clip	1	1
our	λ_s	0.5	0.5
	$\mathcal{L}_{sparse}$ patience	20	20
		Split CIFAR-10	Imagenet-50
optim	batch size	64	64
	learning rate	0.1	0.1
	momentum	0.9	0.9
	lr decay	$[100, 150]$	$[100, 150]$
	weight decay	$5e-4$	$5e-4$
	epochs per task	200	200
	grad. clip	1	1
our	λ_s	1	1
	$\mathcal{L}_{sparse}$ patience	10	0

Table 5.1: Hyperparameters table for continual learning experiments.

only after a predetermined number of epochs, to provide the model the possibility to learn meaningful kernels before starting pruning the uninformative ones. During training, gradient clipping was utilized, ensuring the gradient magnitude to be lower than a predetermined threshold. Moreover, we employed a scheduler dividing the learning rate by a factor of 10 at certain epochs. Such details can be found, for each dataset, in Tab. 5.1, where we highlighted two sets of hyperparameters:

- *optim*: general optimization choices that were kept fixed both for our model and competing methods, in order to ensure fairness.
- *our*: hyperparameters that only concern our model, such as the weight of

the sparsity loss and the number of epochs after which sparsity was introduced (patience).

5.2.2 Task-incremental setting

In the task-incremental setting, an oracle can be queried for task labels during test time. Therefore, we don't rely on the task classifier, exploiting ground-truth task labels to select which gating modules and classification head should be active. This section validates the suitability of the proposed data-dependent gating scheme for continual learning. We compare our model against several competing methods:

- *Joint*: the backbone model trained jointly on all tasks while having access to the entire dataset. We considered its performance as the upper bound.
- *Ewc-On* [163]: the online version of Elastic Weight Consolidation, relying on the latest MAP estimate of the parameters and a running sum of Fisher matrices.
- *LwF* [101]: an approach in which the task loss is regularized by a distillation objective, employing the initial state of the model on the current task as a teacher.
- *HAT* [165]: a mask-based model conditioning the active units in the network on the task label. Despite being the most similar approach to our method, it can only be applied in task-incremental settings.

Tab. 5.2 reports the comparison between methods, in terms of accuracy on all tasks after the whole training procedure. Despite performing very similarily for MNIST, the gap in the consolidation capability of different models emerges as the dataset grows more and more challenging. It is worth mentioning several recurring patterns. First, LwF struggles when the number of tasks grows larger than two. Although its distillation objective is an excellent regularizer against forgetting, it does not allow enough flexibility to the model to acquire new knowledge. Consequently, its accuracy on the most recent task gradually decreases during sequential learning, whereas the performance on the first task is kept very high. Moreover, results highlight the suitability of gating-based schemes (HAT and ours) with respect to other consolidation strategies such as EWC Online. Whereas the former ones prevent any update of relevant parameters, the latter approach only penalizes updating them, eventually incurring a significant degree

	Split MNIST					
	T_1	T_2	T_3	T_4	T_5	avg
Joint (UB)	0.999	0.999	0.999	1.000	0.995	0.999
EWC-On	0.971	0.994	0.934	0.982	0.932	0.963
LwF	0.998	0.979	0.997	**0.999**	0.985	0.992
HAT	0.999	**0.996**	0.999	0.998	0.990	**0.997**
ours	**1.00**	0.994	**1.00**	**0.999**	**0.993**	**0.997**
	Split SVHN					
	T_1	T_2	T_3	T_4	T_5	avg
Joint (UB)	0.983	0.972	0.982	0.983	0.941	0.972
EWC-On	0.906	0.966	0.967	0.965	0.889	0.938
LwF	0.974	0.928	0.863	0.832	0.513	0.822
HAT	0.971	0.967	0.970	0.976	0.924	0.962
ours	**0.978**	**0.972**	**0.983**	**0.988**	**0.946**	**0.974**
	Split CIFAR-10					
	T_1	T_2	T_3	T_4	T_5	avg
Joint (UB)	0.996	0.964	0.979	0.995	0.983	0.983
EWC-On	0.758	0.804	0.803	0.952	0.960	0.855
LwF	0.948	0.873	0.671	0.505	0.514	0.702
HAT	0.988	0.911	**0.953**	**0.985**	0.977	0.963
ours	**0.994**	**0.917**	0.950	0.983	**0.978**	**0.964**

Table 5.2: Task-incremental continual learning results. For each method, we report the final accuracy on all task after incremental training.

of forgetting. Finally, the table shows that our model either performs on-par or outperforms HAT on all datasets, suggesting the beneficial effect of our data-dependent gating scheme and sparsity objective.

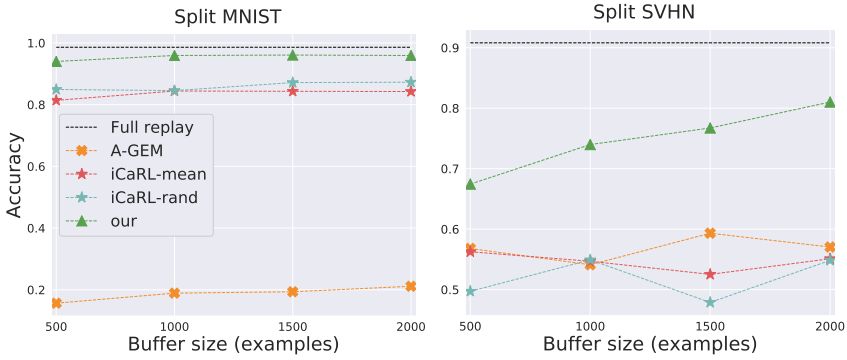


Figure 5.4: Final mean accuracy on all tasks when an episodic memory is employed, as a function of the buffer capacity.

5.2.3 Class-incremental with episodic memory

Next, we move to a class-incremental setting in which no awareness of task labels is available at test time. The unavailability of a task oracle at test time significantly increases the difficulty of the continual learning problem. In this section, we set up an experiment for which the storage of a limited amount of examples (buffer) is allowed. We compare against:

- Full replay: upper bound performance given by replay to the network of an unlimited number of examples.
- iCaRL [147] an approach based on a nearest-neighbor classifier exploiting examples in the buffer. We report the performances both with the original buffer-filling strategy (*iCaRL-mean*) and with the randomized algorithm used for our model (*iCaRL-rand*);
- A-GEM [27]: a buffer-based method correcting parameter updates on the current task so that they don't contradict the gradient computed on the stored examples.

Results are summarized in Fig. 5.4, illustrating the final average accuracy on all tasks at different buffer sizes for the class-incremental Split-MNIST and Split-SVHN benchmarks. The figure highlights several findings. Surprisingly, A-GEM yields a very low performance on MNIST, while providing higher results on SVHN. Further examination on the former dataset revealed that it consistently reaches competitive accuracy on the most recent task, while mostly forgetting the

prior ones. The performance of iCaRL, on the other hand, does not seem to be significantly affected by changing its buffer filling strategy. Moreover, its accuracy seems not to scale with the number of stored examples. In contrast to these methods, our model primarily utilizes the few stored examples for the rehearsal of coarse-grained task prediction, while retaining the accuracy of fine-grained class prediction. As shown in Fig. 5.4, our approach consistently outperforms competing approaches in the class-incremental setting with episodic memory.

5.2.4 Class-incremental with generative memory

Next, we experiment with a class-incremental setting in which no examples are allowed to be stored whatsoever. A popular strategy in this framework is to employ generative models to approximate the distribution of prior tasks and rehearse the backbone network by sampling fake observations from them. Among these, DGM [134] is the state-of-the-art approach, which proposes a class-conditional GAN architecture paired with a hard attention mechanism similar to the one of HAT [165]. Fake examples from the GAN generator are replayed to the discriminator, which includes an auxiliary classifier providing a class prediction. As for our model, as mentioned in Sec. 5.1.3, we rely on multiple task-specific generators.
Tab. 5.4 compares the results of DGM and our model for the class-incremental setting with generative memory. Once again, our method of exploiting rehearsal for only the task classifier proves beneficial. DGM performs particularly well on Split MNIST, where hallucinated examples are almost indistinguishable from real examples. On the contrary, results suggest that class-conditional rehearsal becomes potentially unrewarding as the complexity of the modeled distribution increases, and the visual quality of generated samples degrades.
We report the specification of the generator and discriminator architectures in Tab. 5.3. For every dataset, we trained the WGANs for 2×10^5 total iterations, each of which was composed by 5 and 1 discriminator and generator updates respectively. As for the optimization, we rely on Adam [81] with a learning rate of 10^{-4}, fixing $\beta_1 = 0.5$ and $\beta_2 = 0.9$. The batch size was set to 64. The weight for gradient penalty [58] was set to 10. Inputs were normalized before being fed to the discriminator. Specifically, for MNIST we normalize each image into the range $[0, 1]$, whilst for other datasets we map inputs into the range $[-1, 1]$.

On mixing real and fake images for rehearsal. The common practice when adopting generative replay for continual learning is to exploit a generative model

	Generator	Discriminator
Split MNIST	Linear(128,4096) ReLU Reshape(256,4,4) TrConv2D(256,128,ks=(5,5)) ReLU TrConv2D(128, 64, ks=(5,5)) ReLU TrConv2D(64, 1, ks=(8,8), s=(2,2)) Sigmoid	Conv2D(1,64,ks=(5,5),s=(2, 2)) ReLU Conv2D(64,128,ks=(5,5),s=(2, 2)) ReLU Conv2D(64,128,ks=(5,5),s=(2,2)) ReLU Flatten Linear(4096,1)
Split SVHN	Linear(128,8192) BatchNorm1d ReLU Reshape(512,4,4) TrConv2D(512,256,ks=(2,2)) BatchNorm2d ReLU TrConv2D(256, 128, ks=(2,2)) BatchNorm2d ReLU TrConv2D(128, 3, ks=(2,2), s=(2,2)) TanH	Conv2D(3,128,ks=(3,3),s=(2,2)) LeakyReLU(ns=0.01) Conv2D(128,256,ks=(3,3),s=(2,2)) LeakyReLU(ns=0.01) Conv2D(256,512,ks=(3,3),s=(2,2)) LeakyReLU(ns=0.01) Flatten Linear(8192,1)
Split CIFAR-10	Linear(128,8192) BatchNorm1d ReLU Reshape(512,4,4) TrConv2D(512,256,ks=(2,2)) BatchNorm2d ReLU TrConv2D(256, 128, ks=(2,2)) BatchNorm2d ReLU TrConv2D(128, 3, ks=(2,2), s=(2,2)) TanH	Conv2D(3,128,ks=(3,3),s=(2,2)) LeakyReLU(ns=0.01) Conv2D(128,256,ks=(3,3),s=(2,2)) LeakyReLU(ns=0.01) Conv2D(256,512,ks=(3,3),s=(2,2)) LeakyReLU(ns=0.01) Flatten Linear(8192,1)
Imagenet-50	Linear(128,8192) BatchNorm1d ReLU Reshape(512,4,4) TrConv2D(512,256,ks=(2,2)) BatchNorm2d ReLU TrConv2D(256, 128, ks=(2,2)) BatchNorm2d ReLU TrConv2D(128, 3, ks=(2,2), s=(2,2)) TanH	Conv2D(3,128,ks=(3,3),s=(2,2)) LeakyReLU(ns=0.01) Conv2D(128,256,ks=(3,3),s=(2,2)) LeakyReLU(ns=0.01) Conv2D(256,512,ks=(3,3),s=(2,2)) LeakyReLU(ns=0.01) Flatten Linear(8192,1)

Table 5.3: Architecture of the WGAN employed for the generative experiment. In the table, *ks* indicates kernel sizes, *s* identifies strides, and ns refers to the negative slope of Leaky ReLU activations.

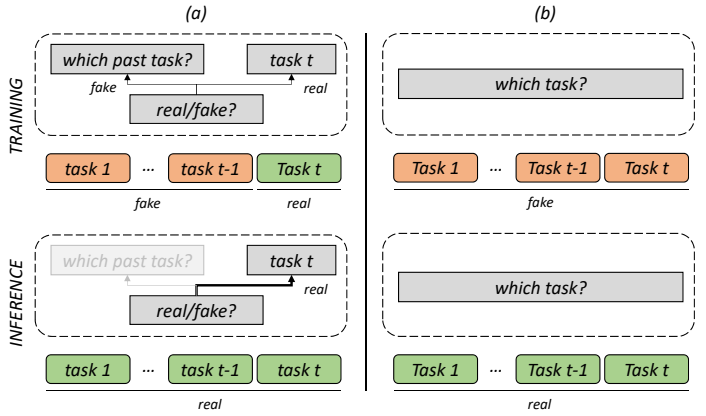


Figure 5.5: Illustration of (a) the degenerate behavior of the task classifier when rehearsed with a mix of real and generated examples and (b) the proposed solution.

to synthesize examples for prior tasks $\{1, \ldots, t-1\}$, while utilizing real examples as representative of the current task t. In early experiments we followed this exact approach, but it led to sub-optimal results. Indeed, the task classifier consistently reached good discrimination capabilities during training, yielding very poor performances at test time. After an in-depth analysis, we conjectured that the task classifier, while being trained on a mixture of real and fake examples, fell into the following very poor classification logic (Fig. 5.5). It first discriminated between the nature of the image (real/fake), learning to map real examples to task t. Only for inputs deemed as fake, a further categorization into tasks $\{1, \ldots, t-1\}$ was carried out. Such a behavior, perfectly legit during training, led to terrible test performances. Indeed, during test only real examples are presented to the network, causing the task classifier to consistently label them as coming from task t.

To overcome such an issue, we remove mixing of real and fake examples during rehearsal, by presenting to the task classifier fake examples also for the task t. In the incremental learning paradigm, this only requires to shift the training of the WGAN generators from the end of a given task to its beginning.

	MNIST	SVHN	CIFAR-10	Imagenet-50
DGMw [134]	0.9646	0.7438	0.5621	0.1782
DGMa [134]	**0.9792**	0.6689	0.5175	0.1516
ours	0.9727	**0.8341**	**0.7006**	**0.3524**

Table 5.4: Class-incremental continual learning results, when replayed examples are provided by a generative model.

5.2.5 Model analysis

In this section, we report further analysis of our model, concerning memory consumption, cost of inference, and investigation of gating patterns.

5.2.5.1 Episodic vs. generative memory

To understand which rehearsal strategy has to be preferred when dealing with class-incremental learning problems, we raise the following question: What is more beneficial between a limited amount of real examples and a (potentially) unlimited amount of generated examples? To shed light on this matter, we report our models' performances on Split SVHN and Split CIFAR-10 as a function of memory budget. Specifically, we compute the memory consumption of episodic memories as the cumulative size of the stored examples. As for generative memories, we consider the number of bytes needed to store their parameters (in single-precision floating-point format), discarding the corresponding discriminators as well as inner activations generated in the sampling process. Fig. 5.6 presents the result of the analysis. As can be seen, the variant of our model relying on memory buffers consistently outperforms its counterpart relying on generative modeling. In the case of CIFAR-10, the generative replay yields an accuracy comparable with an episodic memory of $\approx$ 1.5 MBs, which is more than 20 times smaller than its generators. The gap between the two strategies shrinks on SVHN, due to the simpler image content resulting in better samples from the generators. Finally, our method, when based on memory buffers, outperforms the DGMw model [134] on Split-SVHN, albeit requiring 3.6 times less memory consumption.

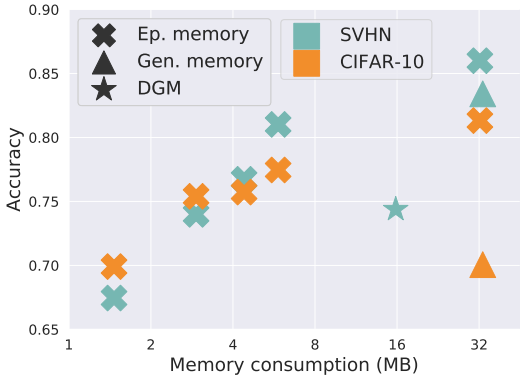


Figure 5.6: Accuracy as a function of replay memory budget.

5.2.5.2 Gate analysis.

We provide a qualitative analysis of the activation of gates across different tasks in Fig. 5.7. Specifically, we use the validation sets of Split MNIST and Imagenet-50 to compute the probability of each gate to be triggered by images from different tasks*. The analysis of the figure suggests two pieces of evidence: First, as more tasks are observed, previously learned features are re-used. This pattern shows that the model does not fall into degenerate solutions, e.g., by completely isolating tasks into different sub-networks. On the contrary, our model profitably exploits pieces of knowledge acquired from previous tasks for the optimization of the future ones. Moreover, a significant number of gates never fire, suggesting that a considerable portion of the backbone capacity is available for learning even more tasks. Additionally, we showcase how images from different tasks activating the same filters show some resemblance in low-level or semantic features (see the caption for details).

5.2.5.3 On the cost of inference.

We next measure the inference cost of our model as the number of tasks increases. Tab. 5.5 reports the average number of multiply-add operations (MAC count) of our model on the test set of Split MNIST and Split CIFAR-10 after learning each

*we report such probabilities for specific layers: layer 1 for Split MNIST (Simple CNN), block 5 for Imagenet-50 (ResNet-18).

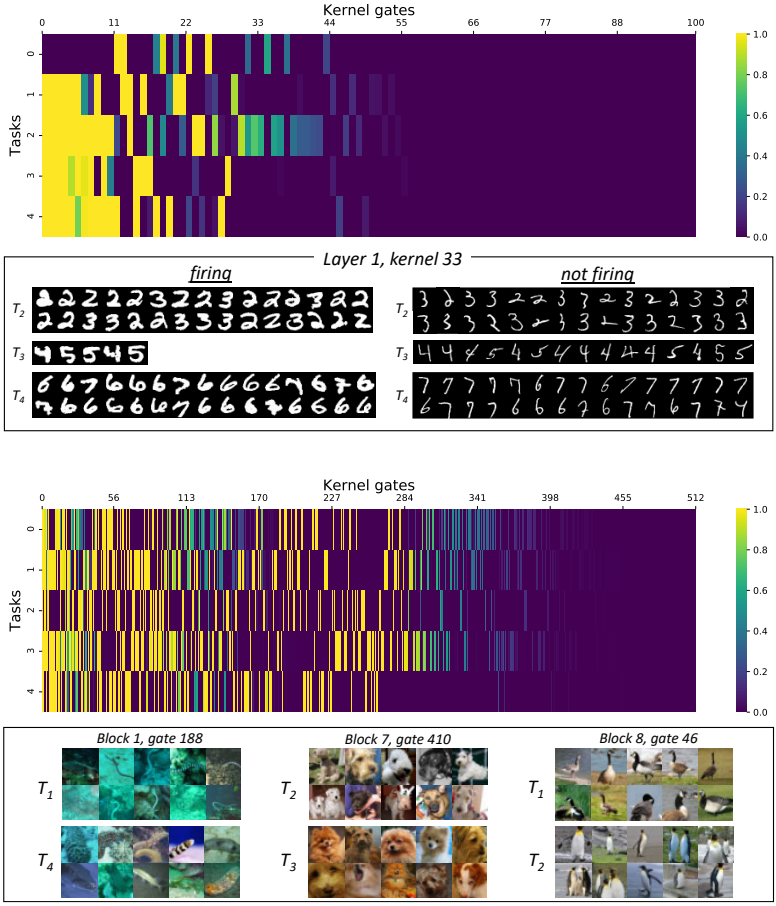


Figure 5.7: Illustration of the gate execution patterns for continually trained models on MNIST (top) and Imagenet-50 (bottom) datasets. The histograms in the top of each subplot show the firing probability of gates in the 1st layer and the 5th residual block respectively. For better illustration, gates are sorted by overall execution rate over all tasks. The bottom subfigures show images from different tasks either triggering or not triggering a specific gate on Split MNIST as well as how - on Imagenet-50 - correlated classes from different tasks fire the same gates (e.g., fishes, different breeds of dogs, birds).

	Split MNIST *(Simple CNN)*		Split CIFAR-10 *(ResNet-18)*	
	TI	CI	TI	CI
Up to T_1	0.064	0.064	2.650	2.650
Up to T_2	0.101	0.209	4.628	9.199
Up to T_3	0.137	0.428	5.028	15.024
Up to T_4	0.136	0.559	5.181	20.680
Up to T_5	0.142	0.725	5.005	24.927
backbone	0.926	0.926	479.920	479.920

Table 5.5: Average MAC counts ($\times 10^6$) of inference in Split MNIST and Split CIFAR-10. We compute MACs on the test sets, at different stages of the optimization (up to T_t), both in task-incremental (TI) and class-incremental (CI) setups.

task, along with the cost of the backbone network. In the task-incremental setting, our model obtains a meaningful saving in the number of operations, thanks to the gating modules selecting only a small subset of filters to apply. In contrast, forward propagation in a class-incremental setting requires as many computational streams as the number of tasks observed so far. However, each of them is extremely cheap as few convolutional units are active. As presented in the table, also in the class-incremental setting, the number of operations never exceeds the cost of forward propagation in the backbone model. The reduction in inference cost is particularly significant for Split CIFAR-10, which is based on a ResNet-18 backbone.

5.2.5.4 Comparison w.r.t. conditional generators

To validate the beneficial effect of the employment of generated examples for the rehearsal of task prediction only, we compare our model based on generative memory (Sec. 5.2.4) against a further baseline. To this end, we still train a WGAN-GP for each task, but instead of training *unconditional* models we train *class-conditional* ones, following the AC-GAN framework [132]. After training N conditional generators, we train the backbone model by generating labeled examples in an i.i.d fashion. We refer to this baseline as C-Gen, and report the final results in Tab. 5.6. The results presented for Split SVHN and Split CIFAR-10, illustrate that generative rehearsal at a task level, instead of at a class level, is be-

neficial in both datasets. We believe our method behaves better for two reasons. First, our model never updates classification heads guided by a loss function computed on generated examples (i.e., potentially poor in visual quality). Therefore, when the task label gets predicted correctly, the classification accuracy is comparable to the one achieved in a task-incremental setup. Moreover, given equivalent generator capacities, conditional generative modeling may be more complex than unconditional modeling, potentially resulting in higher degradation of generated examples.

	class conditioning	rehearsal level	SVHN	CIFAR-10
C-Gen	✓	class	0.7847	0.6384
ours	✗	task	**0.8341**	**0.7006**

Table 5.6: Performance of our model based on generative memory against a baseline comprising a class-conditional generator for each task (C-Gen).

Chapter 6

Conclusions

This thesis illustrated the main research activities I carried out as a Ph.D. student, resulting in contributions in several areas. The overall activity has followed three branches, namely attention prediction in automotive settings, novelty detection and continual learning. This final chapter summarizes the main contributions for each of them, as well as future directions.

Driver attention prediction. Our work started within the field of computer vision for automotive. Specifically, we illustrated a study of human attention dynamics underpinning the driving experience, involving the data collection, its analysis and a computational model for predicting human attention while driving. As a first contribution of this research, we collected and released `DR(eye)VE`, a dataset consisting of driver-centric and car-centric clips, labeled with driver's fixation points on the outer scene. Such dataset is the first one involving large-scale data acquisition in real-world driving settings that is publicly released. An analysis of the collected data highlighted i) the existence of common gaze patterns across drivers and different scenarios and ii) a consistent correlation between factors such as speed or scene semantic, and changes in the driver's focus of attention.
Such evidence guided the development of a computational model replicating the driver's focus of attention from raw video sequences: specifically, it consists of a multi-branch deep neural network, whose components take as input raw color information, motion vectors and semantic maps. Experiments with the proposed architecture and related training strategies yielded state-of-the-art results against prior attention prediction methods. Moreover, we illustrated that our model qual-

itatively replicates some peculiar attentive patterns carried out by humans, and validated its behavior within a user study assessing its perceived safety. Our proposal is the first deep network capable of predicting human attention in real-world driving sequences. Moreover, as it only requires car-centric videos as input, it can seamlessly be integrated with already existing assisted-driving technologies.
Regarding open research questions, the study is not conclusive about the role of the segmentation branch within the full model architecture. Such a branch was introduced after a detailed analysis of the DR(eye)VE dataset highlighted a strong correlation between some semantic classes (e.g. street and cars) and the driver's eye fixations. However, the ablation study didn't show a significant boost in performance when the segmentation branch is employed, with respect to a baseline composed of the RGB and optical flow branch only. This outcome may be due to the fact that the network we employed to compute segmentation maps (i.e. Dilation-10 [203]) was pre-trained on the Cityscapes dataset [35]) and showed poor generalization to the DR(eye)VE dataset. Indeed, especially in very challenging sequences (e.g. rainy, night), we observed poor semantic predictions. Since our work, the research community yielded better and better solutions for semantic segmentation, as well as new large scale datasets of urban scenes (e.g. Mapillary [129]). It is worth questioning whether the outcomes of the ablation study would change by updating the semantic segmentation network and its pre-training.
Moreover, the work could sound even more complete by devising a propotypical Advanced Driver-Assistance System integrating the presented model for attention prediction. As mentioned within the thesis, such a prototype could involve an head pose of an eye tracking system monitoring the driver inside the cabin, and infer where he/her is actually looking in the outside scene. The prediction of the multi-branch network, providing a sort of 'expectation' describing what most drivers would pay attention to in that same situation, could be integrated in (at least) two ways:

- acting as a prior for where the driver is looking, e.g. by refining the predicted fixation point of the driver;

- detecting situations in which the actual gaze of the driver frequently differs from the expected, potentially indicating drowsiness or distraction.

Novelty detection. We then moved our focus towards the the problem of novelty (anomaly) detection, a field of primary importance for various applications,

including fault prediction, defect detection and video surveillance. Novelty detection is typically formalized as a binary classification problem for which examples from the positive class are available at training time. As such, abnormalities are unknown during learning, and their nature cannot be anticipated or conjectured.

Our research led to the introduction of a novel comprehensive learning framework: From a technical perspective, we proposed the utilization of a deep generative autoencoder, paired with an additional autoregressive density estimator learning the distribution of latent vectors by maximum likelihood principles. We showed that the introduction of such an auxiliary module operating in latent space leads to the minimization of the encoder's differential entropy, acting as a regularized for the whole training process. Moreover, we provided intuition on why the minimization of the differential entropy is particularly suitable for novelty detection.

Our model has proved to be capable of learning, in video surveillance contexts, what characterizes stationary and normal situations and, therefore, of recognizing unseen and suspicious events. Experimental results show state-of-the-art performances both in one-class and video anomaly detection settings, fostering the flexibility of our framework for different tasks, even in absence of any data-related assumption.

A first future direction for this research concerns the explainability of the model's novelty score. Indeed, nor in its architecture, neither in its objective, our model involves a component allowing easy interpretation of its behavior. Interpretability is a desirable trait for deep learning models in general, and it is even more important when applied to public monitoring applications (such as video surveillance). However, we tried different strategies for explainability after the model has been trained (e.g. gradient based techniques) but none of them turned out to be suitable. Moreover, the localization capabilities presented in the thesis are devised a posteriori and not really part of the model architecture or problem formulation. It would be interesting, in future works, to directly tackle the interpretability of the network's prediction within the model specification itself.

A further limitation of this research is a direct consequence of the novelty detection formulation, that requires only normal examples to be part of the training set. This assumption is not very realistic in practice, where the data distribution used for training may present a significant mode representing the normality and capture a long tail of unlikely events. It is our belief that our framework would stand even in such training regimes, but our research so far lacks a proper evaluation in such challenging settings.

Continual learning. The last topic met in this thesis is continual learning in deep neural models. As part of a collaboration with Qualcomm AI Research, we presented a novel framework based on conditional computation to tackle catastrophic forgetting in convolutional neural networks. Despite being motivated by the possibility of extending the abovementioned novelty detection model to sequential learning settings, we tackled, as a first step, the forgetting problems in image classification. More precisely, we focused on learning settings in which a single learner encounters a sequence of disjoint classification tasks.
Our framework can be employed both in the presence and in the absence of task labels during test time. In the former case, previously learned knowledge is protected by the employment of task-specific light-weight gating modules, that pair each layer in the learner network. In the latter and more realistic case, a task classifier is trained to take the place of a task oracle. Through extensive experiments, we validated the performance of our model against existing solutions both in presence and absence of task oracles, and demonstrated its suitability in four continual learning benchmarks.
A potential limitation of our model lies in the need of knowledge about task boundaries during training. Indeed, the precise moment when a task ends and a new one begins is needed to instantiate new gating models for each convolution. Moreover, an output unit has to be added to the task classifier. Although such assumption is standard in literature, and fairly reasonable when dealing with classification problems (a memory of 'seen classes' suffice for the identification of novel tasks during training), it may however fall for unsupervised learning problems or whenever the distribution shifts gradually and not instantly.
Moreover, as mentioned within the thesis, we started this line of research to provide anomaly detection models with online learning capabilities. However, our proposal has only been studied in classification settings (in order to provide a fair comparison with respect to prior models). As a future direction, it would be interesting to explore its potential in different contexts, starting from anomaly detection problems.

Publications and achivements

The research and results presented in this thesis resulted in publications in international conferences and journals. Specifically, the work on driver attention prediction has been published on the IEEE Transactions on Pattern Analysis and Machine Intelligence. Since its release the `DR(eye)VE` dataset has reached more than $1,000$ downloads, and it is currently being used by several international

automotive companies. The work on anomaly detection has been presented at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR19). Finally, the work on continual learning will be presented at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR20).
The following appendix will illustrate additional research activities that I carried out, as well as an exhaustive list of publications comprehensive of works falling outside the main lines of this thesis. Among them, the conference papers "Wearable vision for retrieving architectural details in augmented tourist experiences" and "Learning to map vehicles into bird's eye view" has been assigned the best paper award and the best paper honorable mention respectively.

Appendix

This appendix briefly illustrates two additional research projects I have been involved in within my Ph.D. studies. The topics that will be covered are loosely related to the ones discussed so far. However, due to the relevance of the problems tackled, the encouraging outcomes and the related publications, I believe it is important to mention them.

A.1 Optical Flow Estimation for Automotive Applications

Within the automotive context, optical flow estimation represents one of the most active research fields but, despite the efforts, it is still an open problem.

This is due to several factors. First, the car ego-motion heavily affects the optical flow field. Indeed, urban scenes are mainly composed of static, still objects, whose moving patterns within the image plane are strongly correlated to the camera movement. However, several surrounding objects can move independently (such as cars and pedestrians), and their movements are perhaps even more important. For an optical flow model to be reliable, both types of motion need to be correctly estimated. Furthermore, machine learning models typically rely on a groundtruth signal during their optimization, and precise pixel-level optical flow ground-truth maps are extremely hard to collect. Alternatively, computer graphics approaches [50] or videogames [150] are employed to acquire larger synthetic datasets, at the expense of a significant domain shift due to a depleted photorealism. Eventually, computer vision models for automotive applications are constrained by real-time requirements, both in autonomous and assisted driving contexts.

The following section will illustrate our solution to the the above-mentioned issues. Our first contribution is to bootstrap the optical flow estimation by a projective geometric transformation between consecutive image frames, as we argue that the global ego-motion of a car can be coarsely approximated by the motion field implied by such transformation. This choice allows us to delegate the estimation of most flow vectors to an extremely shallow network, and then to refine its prediction accounting for depth structure and moving objects. Moreover, we

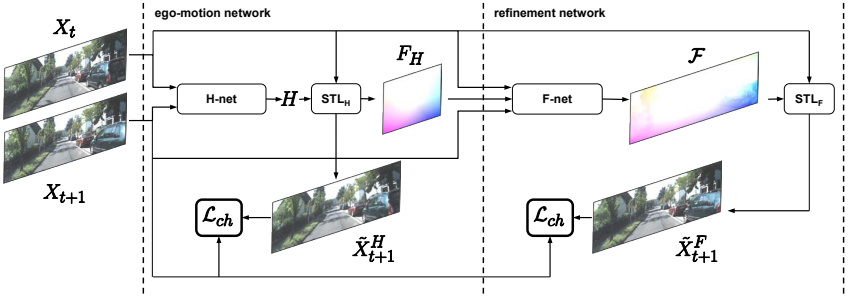


Figure A.1: Architecture of the proposed model. Frames X_t and X_{t+1} are fed into a first ego-motion network that estimates a projective transformation. STL_H then warps X_t accordingly, resulting in the estimated frame $\tilde{X}^H_{t+1}$. The bootstrapped flow is then fed to a second network along with input frames, and its refinement $\mathcal{F}$ is employed to obtain a second reconstruction, $\tilde{X}^F_{t+1}$. Both reconstructions are guided by a Charbonnier penalty $\mathcal{L}_{ch}$ to approximate X_{t+1}. During inference, the output of F-net, $\mathcal{F}$, is the predicted optical flow.

only rely on self-supervision to train our model. Indeed, building on the Spatial Transformer Layer [72] (STL), we embrace an optimization framework in which, instead of directly minimizing flow vector errors, we warp frames according to the estimate flow field and reconstruction failures.

A.1.1 The model

The estimation of a dense optical flow between two adjacent frames X_t, X_{t+1} requires to output two translation components u and v for each pixel, resulting in a transformation with $2 \times h \times w$ parameters (where h, w represent the height and width of the two frames respectively). On the contrary, we advocate for the existence of dominant motion patterns in the automotive scenario, which are due to the motion of the car itself. By approximating such global patterns with a single projective transformation, it is possible to drop the parameters characterizing the transformation down to eight. Our architecture estimates the motion field in two sequential steps (see Fig. A.1. First, a shallow network (H-net) outputs the parameters of a projective transformation embedding a motion field approximating the one implied by camera motion (that we refer to as homographic flow). Then,

a second deeper network (F-net) is responsible for its refinement, and for the recovery of depth and local fields belonging to moving objects.

Before introducing these two components, we briefly illustrate how a STL can be employed to train an optical flow estimator by self-supervision. Given a geometric transformation T_ϕ estimated from two frames X_t, X_{t+1} by a generic parametric function (e.g. a neural network), $T_\phi(X_t)$ can be computed by means of a STL [72]. Being fully differentiable, this layer also allows to propagate the gradient of a given reconstruction loss to the transformation estimator. In our setting, T_ϕ embeds optical flow vectors while $\tilde{X}_{t+1} = T_\phi(X_t)$ results in the estimate of frame X_{t+1} under the transformation T_ϕ. During training, our model is tasked to optimize a Charbonnier reconstruction penalty, expressed as

$$\mathcal{L}_{ch} = \sqrt{(\tilde{X}_{t+1} - X_{t+1})^2 + \epsilon}, \tag{A.1}$$

where ϵ is a small regularization constant (fixed to 0.1 in our experiments). The Charbonnier penalty is a differentiable version of the l_1 norm and guides the model towards better reconstructions, i.e. better optical flow estimates, without the need of ground-truth optical flow fields.

Ego-motion network. The H-net architecture is small sized, featuring 6 convolutional layers and 3 fully connected layers resulting in 600k parameters overall. It is fed with two subsequent frames X_t and X_{t+1} stacked on the channel dimension and outputs the 8 parameters of the transformation. Such projection is fed to the spatial transformer layer STL_H, warping Xt into the reconstruction $\tilde{X}^H_{t+1}$. Notably, the flow field implied by the projective transformation that can be recovered from STL_H. More precisely, given a uniform sampling grid G holding column-wise pixel homogeneous coordinates, STL_H applies the input transformation to obtain the warped grid $\tilde{G} = H \cdot G$. Since $\tilde{G}$ holds the warped coordinates of each pixel, the flow field can seamlessly be recovered as $F_H = \tilde{G} - G$.

Refinement network. The flow field produced by H-net cannot cope with details and moving objects. Therefore we refine it employing a second, deeper network (F-net) inspired by the encoder-decoder architecture in [43]. The encoder consists of five layer blocks, each of which is composed by three 3×3 convolutional layers with leaky ReLU activations. All convolutions in a block share unary strides except for the last one, which has stride 2. The decoder mirrors

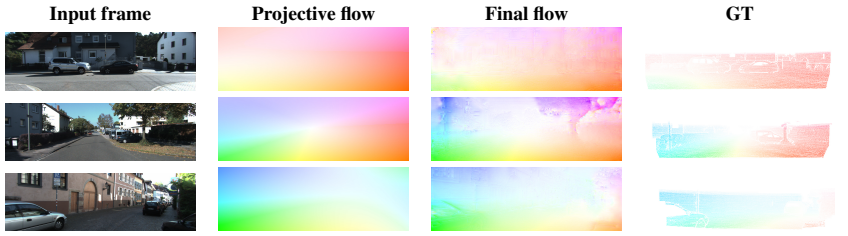


Figure A.2: Some examples of estimated optical flow fields on the Kitti 2012 dataset.

the encoder structure with transposed-convolution blocks, and a top 2-channel convolution layer produces the final estimate in the range $[-1, 1]$ using a TanH non-linearity. F-net is fed with a channel-wise concatenation of X_t, X_{t+1} and F_H, and outputs a dense flow map $\mathcal{F} \in \mathbb{R}^{2\times w\times h}$. Hence, the spatial transformer layer STL_F warps X_t into $\tilde{X}^F_{t+1}$ and the reconstruction error w.r.t. the frame X_{t+1} is minimized by means of the Charbonnier loss (Eq. (A.1)). This guides F-net towards focusing on moving objects and fine details neglected by the projective bootstrap.

A.1.2 Experimental results

Datasets and training We employ large-scale automotive datasets such as Kitti raw [52] and `DR(eye)VE` [5], which was introduced in Sec. 3.1.
To train our network, we utilize frame pairs sampled from the Kitti raw dataset; when not specifically mentioned otherwise, no fine-tuning has been performed on the individual datasets used for testing. We train the network using the Adam optimizer [81] with default parameters except for the learning rate (10^{-4} in our setting) and β_1 that was set to 0.5. Mini-batch size was set to 16, and training was stopped after 250 epochs, each of which was composed of 1000 mini-batches. The loss function of Eq. A.1 is employed during the training minimizing the reconstruction errors of both the H-net and the F-net.

Evaluation: Kitti Flow 2012. We first test our proposal on the labeled examples of the Kitti benchmark, namely Kitti Flow. Fig. A.2 illustrates several successful network predictions. Following standard optical flow evaluation protocols, to evaluate the performance of our method we adopt the following metrics:

Method	Training		Testing (noc)		Testing (all)		Time (s)
	Acc@5	APE	Acc@5	APE	Acc@5	APE	
FlowNetS [43]	-	-	0.759	5.0	0.673	9.1	0.08
DeepFlow [199]	-	-	0.953	1.5	0.851	5.8	17.0
EpicFlow [149]	-	-	0.946	1.5	0.871	3.8	15.0
SpyNet [145]	0.851	4.0	0.831	4.7	0.739	10.0	0.16
SpyNet (ft) [145]	-	-	0.916	2.0	0.842	4.1	0.16
FlowNet 2.0 [67]	-	4.1	-	-	-	-	0.12
FlowNet 2.0 (ft)[67]	-	1.3	-	1.8	-	-	0.12
Long *et al.* [107]	0.716	4.7	-	-	-	-	486
Yu *et al.* [204]	-	4.3	0.779	4.6	0.681	11.3	0.03
Liu *et al.* (ft)[106]	-	-	-	-	-	9.5	-
SIFT+F-net	0.725	5.9	-	-	-	-	0.28
Ours	0.866	3.1	0.842	3.6	0.759	7.8	0.05

Table A.1: Performance comparison on the Kitti Flow 2012 dataset. Note that not all the methods report results on the public leaderboards (testing set). The table is divided in three sections: hand-crafted and supervised methods, unsupervised/self-supervised, our proposal. *(ft)* indicates the model fine-tuned on Kitti Flow.

Accuracy@5 (i.e. ratio of motion vectors with end point error lower than 5 pixels) and APE (i.e. average point error of all motion vectors).

We evaluate our proposal against three recent self-supervised deep-learning based approaches [107, 204, 106]. We also report the results of recent methods tackling optical flow estimation using both hand-crafted and learned features [43, 199, 149]. Finally, we evaluate the performance of a baseline where the homography matrix between the two frames is computed using traditional computer vision techniques (SIFT+RANSAC) instead of using the H-net (SIFT+F-net).

Tab. A.1 reports the results of this evaluation. In particular, our proposal performs favorably against recent self-supervised approaches. It shows competitive performance and whilst not reaching the average point error results of DeepFlow [199] or EpicFlow [149], it is three orders of magnitude faster, requiring 0.05 sec/pair compared to the 17 sec/pair and 15 sec/pair of DeepFlow and EpicFlow respectively. Furthermore, we compare our results to more recent

Method	PSNR				SSIM
	Night	Rain	Day	Avg	Avg
DeepFlow [199]	23.99	18.45	19.02	20.49	0.878
FlowNet 2.0 [67]	22.65	19.44	20.02	20.70	0.859
EpicFlow [149]	24.58	18.67	19.44	20.90	0.875
Ours	34.82	27.91	29.33	30.69	0.913

Table A.2: Performance comparison on the DR(eye)VE dataset (higher is better). The evaluation is performed on *Downtown* sequences of the dataset.

deep-learning based methods who can achieve comparable running-times to ours, namely Flownet 2.0 [67] and SpyNet [145]. Tab. A.1 shows that, in absence of fine-tuning, our method outperforms both approaches. On the other hand, both Flownet 2.0 and SpyNet considerably improve their performance when fine-tuning on Kitti training set.
Concerning the baseline performance (SIFT+F-net), it can achieve acceptable results, but performs worse than bootstrapping with the transformation learned by the H-net. Moreover, our full network only requires a single forward propagation, eliminating the need for extraction and matching of SIFT keypoints: this results in our approach being more than 5 time faster than the baseline.

Evaluation: `DR(eye)VE`. As the `DR(eye)VE`dataset lacks ground truth flow information, we evaluate the performance of our method by assessing the quality of the warping of between successive frames: intuitively, the better the flow estimate, the lower the measured error in the reconstruction $\tilde{X}^F_{t+1}$. In more detail, given a frame X_t and the optical flow transformation T_ϕ, we use the Spatial Transformer Layer to get the reconstructed estimate $\tilde{X}^F_{t+1}$ and measure its peak signal-to-noise ratio (PSNR, according to the protocol in [106]). Due to PSNR sensibility to small changes resulting, for example, from small flow intensity errors shifting objects by a few pixels but keeping the overall scene intact, we also evaluate the quality of $\tilde{X}^F_{t+1}$ using the Structural Similarity Index Measure (SSIM), which not only considers individual pixel intensities but also accounts for the structural similarity of small patches. Tab. A.2 reports the results of such evaluation where no fine-tuning has been performed. To analyze the impact of different environmental conditions on optical flow estimation, Tab. A.2

also reports results divided by sequence type. The slightly higher PSNR across all methods during the Night sequences is due to the lower intensity of the images, effectively resulting in a lower error. Beside that, the experiment shows that environmental conditions do not have a significant impact on optical flow estimation. SSIM, while confirming the same overall trend, shows a smaller gap between methods. As mentioned earlier, this is due to SSIM being a more stable metric where the coherence of pixel neighborhoods are also taken into account. Please note that using the spatial transformer layer as warper ensures that the reconstruction pipeline is the same for every method and differences in PSNR are only due to the flow.

A.2 Convolutional Cluster Pooling: a hierarchical approach to graph classification

Convolutional Neural Networks (CNNs) constitute the standard technology in different machine learning applications, such as speech recognition [64], image classification [90], and video analysis [176]. In these domains, data can be described as a signal defined on a regular grid. One of the key aspects of CNNs is that such a regular structure enables the exploitation of local and stationary properties of data. Moreover, the convolution operator, being equivariant to translations, allows filters with a limited support on the input grid, leading to a significantly smaller number of parameters with respect to Fully Connected Networks. However, we are surrounded by structured data, whose anatomy typically present an irregular and non-euclidean nature: document relational databases, 3D skeletal data, social networks data and chemical compounds are only a few examples. In all these domains, the relationships among entities transcend grid-like connectivities, and graphs emerge as powerful tools, due to their capability of modeling complex topological structures. Therefore, many efforts have recently been made [22, 84, 38] in an attempt to generalise CNNs for graph-structured data. In this work we focus on signal classification in homogeneus graphs. In such context, each example obeys a single $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ weighted graph, which reflects the structure of the given problem. However, samples differ by the value (i.e. a feature vector) of each vertex in the graph. Similarily to [66], we refer to each sample as a realisation of a signal on $\mathcal{G}$. The aim is to learn a function which maps each sample into the label space. By doing so, similarly to what CNNs do for images, at each step we shall exploit information coming from the neighbouring nodes. Our architectures are composed by a novel building block,

that we name Convolutional Cluster Pooling (CCP) operator. Such a layer, which is the main subject of this study, firstly performs a clustering operation on the input graph, resulting in a coarser output graph, whose affinity matrix reflects relationships among clusters learned at training time. By doing so, a good basis for building local receptive fields is achieved. Secondly, according to the neighbourhood provided by the clustering step, the layer selects for each cluster a fixed number of candidate nodes for the aggregation phase, and sorts them depending on a centrality-based rank within the cluster. In this respect, it is worth noting that weight sharing across the graph's neighbourhoods can be successfully exploited. The following section will illustrate the CCP layer in more detail.

A.2.1 The model

A graph $\mathcal{G}$ can be defined as an ordered pair $(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of $\mathcal{N}$ nodes and $\mathcal{E} \in \mathcal{V} \times \mathcal{V}$ a set of edges. In this work we are interested in classifying signals defined on an undirected and weighted graph, in which $\mathcal{E}$ can be described by a real symmetric matrix $\mathcal{A} \in \mathbb{R}^{\mathcal{N} \times \mathcal{N}}$ which, for each couple of vertices $\mathcal{V}_i$ and $\mathcal{V}_j \in \mathcal{V}$, provides the strength (weight) of their connections. More generally, we refer to $\mathcal{A}$ as an affinity matrix, in which each entry $\mathcal{A}_{i,j}$ gives an affinity score between $\mathcal{V}_i$ and $\mathcal{V}_j$. In addition to the affinity matrix, which describes the topology of the graph and the relationships between nodes, it is common practice to define a signal $\mathcal{F} : \mathcal{V} \rightarrow \mathbb{R}^{d_{IN}}$ on the vertex set, which associates a d_{IN} dimensional feature vector to each node of the graph.

The purpose of our proposal is to exploit the clustering mechanism in order to define a convolutional-like operator, able to ensure equivariance to translation and weight sharing in graph contexts as standard convolutions do. At a high level, our CCP operator can be considered as a layer which, at step m, takes in input an affinity matrix $\mathcal{A}^{\mathcal{K}_m}$ and a multi-dimensional $\mathcal{F}^{(m)} \in \mathbb{R}^{|\mathcal{K}_m| \times d_{IN}}$ signal defined on the vertex set. The output is composed by a new reduced affinity matrix $\mathcal{A}^{\mathcal{K}_{m+1}}$ (reflecting the results of the cluster step) and a pooled signal $\mathcal{F}^{(m+1)} \in \mathbb{R}^{|\mathcal{K}_{m+1}| \times d_{OUT}}$ (reflecting the results of the filter step where d_{OUT} is the dimension of the newly computed features). All architectures used in our experiments are composed by stacking CCP layers, which combine the pooling and filtering stage and, at the same time, increase the number of feature maps, as suggested in [22].

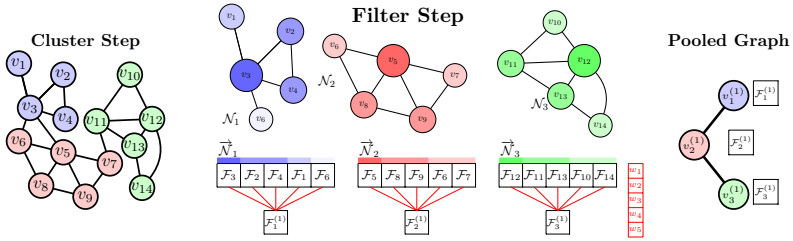


Figure A.3: An illustration of the proposed CCP layer. Left, the cluster step outputs node's membership distribution among a pre-defined number of clusters. Centre, the filter step: a) selects, for each cluster, candidate nodes whose feature vectors will be aggregated; b) arranges such candidates according to a with-in cluster centrality score, building the support for the next step; c) aggregates feature vectors by means of a standard 1-d convolution, with stride equal to the kernel width (in this case, $L = 5$). Right, the result of CPP layer consists in a coarsened graph coupled with its filtered pooled signal. Best viewed in color.

Cluster Step. First of all, our model performs a soft-clustering step on the input graph (Fig. A.3, left). To the purpose we define the learnable matrix $U^{(m+1)} \in \mathbb{R}^{|\mathcal{K}_m| \times |\mathcal{K}_{m+1}|}$ and its row-wise softmax activation $K^{(m+1)}$, defining the mapping of nodes at layer m to the clusters in the downsampled graph at layer $m+1$. The downsampled affinity matrix $\mathcal{A}^{\mathcal{K}}_{m+1}$ describing the soft-partitioned graph induced by $K^{(m+1)}$ is computed by means of a quadratic form as

$$\mathcal{A}^{\mathcal{K}}_{m+1} = K^{(m+1)T}(\mathcal{A}^{\mathcal{K}}_{m} - I \odot \mathcal{A}^{\mathcal{K}}_{m})K^{(m+1)}, \tag{A.2}$$

where I identifies the identity matrix. Eventually, we add a normalisation operation based on the degree matrix D [84] in order to prevent numerical instabilities:

$$\overline{\mathcal{A}}^{\mathcal{K}_{m+1}} = D^{-\frac{1}{2}}\mathcal{A}^{\mathcal{K}_{m+1}}D^{-\frac{1}{2}}. \tag{A.3}$$

Neighbourhood selection. For each cluster $\mathcal{K}^{(m+1)}_k$, we select as candidate set $\mathcal{N}^{(m+1)}_k$ for the filtering stage the set containing the most L representative nodes (where L is an hyperparameter) as:

$$\mathcal{N}^{(m+1)}_k = \underset{\mathcal{V}' \subset \mathcal{V}^{(m)}, |\mathcal{V}'|=L}{\operatorname{argmax}} \sum_{v \in \mathcal{V}'} Rank\,(v \to \mathcal{K}^{(m+1)}_k), \tag{A.4}$$

where the rank of a vertex $\mathcal{V}_i^{(m)}$ for a particular cluster $\mathcal{K}_k^{(m+1)}$ is given by its centrality in that cluster:

$$Rank\left(\mathcal{V}_i^{(m)} \rightarrow \mathcal{K}_k^{(m+1)}\right) = (1 + K_{i,k}^{(m+1)}) \sum_{\substack{j=1 \\ j \neq i}}^{|\mathcal{K}_m|} \mathcal{A}_{i,j}^{\mathcal{K}_m} K_{j,k}^{(m+1)} \tag{A.5}$$

More intuitively, we consider a node more central if it has a high membership value for the cluster under consideration and, at the same time, a large part of its direct neighbours nodes share the same cluster in the input graph (Fig. A.3, centre top).
Further, for each cluster, we compute its features as a linear combination over the feature vectors of its inner nodes. In doing so, we want to exploit the weight sharing property across all neighbours. To this end, we create a coherent support across clusters, in terms of their inner topological structure. In this respect, our proposal is to sort candidates by their centrality within the neighbourhood and, afterwards, apply the same kernel to all clusters. Therefore, an ordered set $\overrightarrow{\mathcal{N}}_k^{(m+1)}$ is recovered by sorting the candidates set $\mathcal{N}_k^{(m+1)}$ according to the $Rank$ function. By doing so, the l-th weight of the kernel is always multiplied by the feature vector being owned by the l-th node of the neighbour in terms of centrality.

Neighbourhood Aggregation. The problem we face when sorting nodes by cluster centrality and then applying the same kernel to all neighbours, is that, by doing so we do not take into account the irregularity of the neighbour's shapes. As a matter of fact, the risk of this solution consists in the equal treatment, for different clusters, of nodes indexed in the same position by the sorting stage, whilst exhibiting considerably different centrality values. In order to mitigate such risk, we once again use the centrality measure to implement a gating mechanism on feature vectors during the aggregation phase. The underlying idea is to make the filtering operation invariant to different neighbours and let the gating mechanism address different cluster's structures and shapes. Roughly speaking, before applying the filtering operation described above, we are giving the centrality scores in input to a generic smoothed function $\sigma : \mathbb{R} \rightarrow (0, 1)$ (e.g. the sigmoid function). Once this has been done, we perform a point-wise multiplication on the feature vectors of each candidate node. The desired effect of this operation is to attenuate information coming from distant or non-central nodes and, at the same time, preserve signals coming from nodes that reside in the inner part of the cluster.

Lastly, our model computes the pooled feature vector as follows:

$$\mathcal{F}_{k,j}^{(m+1)} = \sum_{i=1}^{d_{IN}} \sum_{l=1}^{L} W_{l,i,j} \left(\sigma_{k,l} \cdot \overrightarrow{\mathcal{N}}_{k}^{(m+1)}(l,i)\right) + b_j, \tag{A.6}$$

where $W \in \mathbb{R}^{L \times d_{IN} \times d_{OUT}}$ and $b \in \mathbb{R}^{d_{OUT}}$ are learnable parameters of our CCP layer, whereas $\sigma_{k,l}$ refers to the gate's activation value computed at $Rank\left(\mathcal{V}_{\phi(l)}^{(m)} \rightarrow \mathcal{K}_{k}^{(m+1)}\right)$. As shown in Fig. A.3 (centre bottom), this operation is equivalent to a 1-d convolution, enabling weight sharing across clusters.

A.2.2 Experiments

In order to show the generality and effectiveness of our model for classification, we apply our architecture to three different domains. First, we train our model to classify human actions, given the 3D coordinates of each skeleton's joint: to this end, we evaluate it on NTU RGB-D dataset [166]. Secondly, we apply our solution on the 20NEWS dataset, where the goal is to address a text categorisation problem. Finally, we conduct experiments on image classification. More specifically, we use CIFAR-10 [88] as benchmark test, which is a challenging dataset for non-CNN architectures.

Implementation details. In each experiment, all parameters are learned using Adam [81] as an optimisation algorithm, with an initial learning rate fixed to 0.001. We use ELU [33] as activation function and Batch Normalization [69] in all layers to speed up the convergence. Moreover, we apply dropout and l_2 weight regularisation (with value 10^{-4}) to prevent overfitting, as well as standard data augmentation for CIFAR-10 and noise injection coupled with random 3d rotations for NTU RGB-D. In each experiment, we subsample the input graph until its cardinality becomes equal to one: afterwards, we feed its feature vector into two fully connected layers, followed by a softmax layer providing the target class predictions.

Action Recognition. The NTU RGB+D Human Activity Dataset [166] is one of the largest datasets for human action recognition. It contains 56,880 action samples for 60 different actions, captured by the Kinect v.2 sensors. Each sample, showing a daily action performed by one or two participants, is made available in 4 different modalities: RGB videos, depth map sequences, 3D skeletal data

Method	Cross Subject	Cross View
Lie Group [192]	50.1	52.8
HBRNN-L [41]	59.1	64.0
P-LSTM [166]	62.9	70.3
ST-LSTM+TS [103]	69.2	77.7
TGCNN [209]	71.4	82.9
Temporal Conv [80]	74.3	83.1
Deep STGC$_K$ [99]	74.9	86.3
C-CNN + MTLN [78]	79.6	84.8
CCP (our)	**80.1**	**86.8**

Table A.3: Summary of results in terms of classification accuracy for NTU RGB+D.

and infrared videos. However, we only rely on the 3D skeletal data, represented by a temporal sequence of 25 joints*. To this end, we model each sequence as a signal $\mathcal{F}^{(0)} \in \mathbb{R}^{(25 \cdot T) \times 3}$ defined on a single fixed spatio temporal graph, whose structure can be summarised as follows: a vertex set $\mathcal{V}_{ST} = \{v_{i,t} | i = 1, 2, \ldots, 25\ , t = 1, 2, \ldots, T\}$, which includes all joints captured in a fixed length sequence ($T = 80$). The edge set $\mathcal{E}_{ST}$ can be defined as the union of two distinct subsets: $\mathcal{E}_S$, which contains all edges within each frame according to the natural human-body connectivity, and $\mathcal{E}_T$, which includes all edges existing between the same joint in two adjacent frame.

In order to evaluate the model's performance, as described in [166], we run two different standard benchmarks: the cross-subject setting, in which the train/test split is based on two disjoint sets of actors; and the cross-view setting, where the test samples are captured from a different camera from those collecting the training sequences. We report in Tab. A.3 the top-1 classification accuracy on both settings, comparing it with other existing approaches. As illustrated, CCP outperforms previous state-of-the-art methods on this dataset, including other graph-oriented architectures [209, 99], despite being more general and not specifically designed to only address action recognition settings.

*The pre-processing step on skeleton data has been performed according to the guidelines provided in [166].

Method	Accuracy
Linear SVM †	65.9
Softmax †	66.3
Multinomial Naive Bayes †	68.5
FC2500-FC500 †	65.8
Chebyshev - GC32 [38] †	68.3
CCP (our)	**70.1**

† Baselines' results published in [38].

Table A.4: Text categorisation accuracy on 20NEWS.

Text Categorisation. we apply our model on text categorisation by conducting experiments on the 20NEWS dataset [76], adhering to the guidelines described in [38] for the construction of the shared graph. To summarise, such protocol models each text as a graph which has a node for each common word in the document set. On the other hand, the pairwise connectivities of such graph are shared and obtained assessing the similarities inducted by word2vec embeddings [124], followed by a discretisation step computed through a K-NN pass (with $K = 16$). This way, each document $\mathcal{D}$ can be represented as a signal over a fixed graph, implemented as the word's distribution observed in $\mathcal{D}$.
As indicated in Tab. A.4, the discussed approach leads to good performances, defeating both baselines and the graph convolutional layer based on polynomial spectral filters. On this latter point, our architecture seems to take advantages of its depth and hierarchical nature, differently from [38] where a shallow graph convolutional network has been employed to categorise documents.

Image Classification. We conduct image recognition experiments on CIFAR-10. Each image, labeled into one of ten classes, can be treated as a signal defined on a graph, which can in turn be modeled as a 32×32 grid structure. In particular, every pixel is a vertex such a graph, linked to its neighbours following a 8-connectivity. The colour information is encoded as a signal $\mathcal{F}^{(0)} \in \mathbb{R}^{1024\times 3}$ over such vertexes. As shown in Tab. A.5, CCP obtains an encouraging performance in terms of classification accuracy on test set. Indeed, our method outperforms both the best reported fully connected (FC) network [102] and Graph-CNNs [172] - to the best of our knowledge, the only graph classification model in literature that reports results on CIFAR-10 - by a significant margin. To put our results into

Method	Accuracy
Graph-CNNs [172]	68.3
FC [102]	78.6
CCP (our)	**84.4**
Stochastic Pooling [205]	84.9
ResNet-50 [60]	93.6

Table A.5: Image classification accuracy on CIFAR-10.

perspective, we report the performance obtained by [205], which is the nearest score founded in the literature given by a deep CNN, as well as the results of a state-of-art CNNs like [60]. The gap with respect to the latter is still consistent, suggesting that there is still room for improvement in euclidean domains.

List of publications

- Davide Abati, Jakub Tomczak, Tijmen Blankevoort, Simone Calderara, Rita Cucchiara, and Babak Ehteshami Bejnordi. Conditional channel gated networks for task-aware continual learning. *IEEE International Conference on Computer Vision and Pattern Recognition*, 2020.

- Davide Abati, Angelo Porrello, Simone Calderara, and Rita Cucchiara. Latent space autoregression for novelty detection. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2019.

- Angelo Porrello, Davide Abati, Simone Calderara, and Rita Cucchiara. Classifying signals on irregular domains via convolutional cluster pooling. *International Conference on Artificial Intelligence and Statistics*, 2019.

- Davide Abati, Angelo Porrello, Simone Calderara, and Rita Cucchiara. And: Autoregressive novelty detectors. *arXiv preprint arXiv:1807.01653*, 2018.

- Stefano Alletto, Davide Abati, Simone Calderara, Rita Cucchiara, and Luca Rigazio. Self-supervised optical flow estimation by projective bootstrap. *IEEE Transactions on Intelligent Transportation Systems*, 2018.

- Marcella Cornia, Davide Abati, Lorenzo Baraldi, Andrea Palazzi, Simone Calderara, and Rita Cucchiara. Attentive models in vision: Computing saliency maps in the deep learning era. *Intelligenza Artificiale*, 2018.

- Andrea Palazzi, Davide Abati, Francesco Solera, Rita Cucchiara, et al. Predicting the driver's focus of attention: the dr (eye) ve project. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

- Marcella Cornia, Davide Abati, Lorenzo Baraldi, Andrea Palazzi, Simone Calderara, and Rita Cucchiara. Attentive models in vision: computing saliency

maps in the deep learning era. In *Conference of the Italian Association for Artificial Intelligence*, 2017.

- Andrea Palazzi, Guido Borghi, Davide Abati, Simone Calderara, and Rita Cucchiara. Learning to map vehicles into bird's eye view. In *International Conference on Image Analysis and Processing*, 2017.

- Stefano Alletto, Davide Abati, Giuseppe Serra, and Rita Cucchiara. Exploring architectural details through a wearable egocentric vision device. *Sensors*, 2016.

- Stefano Alletto, Davide Abati, Giuseppe Serra, and Rita Cucchiara. Wearable vision for retrieving architectural details in augmented tourist experiences. In *2015 7th International Conference on Intelligent Technologies for Interactive Entertainment (INTETAIN)*, 2015.

Bibliography

[1] R. Achanta, S. Hemami, F. Estrada, and S. Susstrunk. Frequency-tuned salient region detection. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1597–1604, June 2009. 8

[2] Amit Adam, Ehud Rivlin, Ilan Shimshoni, and Daviv Reinitz. Robust real-time unusual event detection using multiple fixed-location monitors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):555–560, 2008. 11

[3] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *European Conference on Computer Vision*, 2018. 12

[4] Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2019. 93

[5] Stefano Alletto, Andrea Palazzi, Francesco Solera, Simone Calderara, and Rita Cucchiara. Dr(eye)ve: a dataset for attention-based tasks with applications to autonomous and assisted driving. In *IEEE International Conference on Computer Vision and Pattern Recognition Workshops*, 2016. 19, 20, 122

[6] Borislav Antić and Björn Ommer. Video parsing for abnormality detection. In *IEEE International Conference on Computer Vision*, pages 2415–2422. IEEE, 2011. 12

[7] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv:1803.01271*, 2018. 69

[8] Johannes Ballé, Valero Laparra, and Eero P Simoncelli. End-to-end optimized image compression. *International Conference on Learning Representations*, 2017. 64

[9] Lorenzo Baraldi, Costantino Grana, and Rita Cucchiara. Hierarchical boundary-aware neural encoder for video captioning. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2017. 28

[10] Andrew Barto, Marco Mirolli, and Gianluca Baldassarre. Novelty or surprise? *Frontiers in psychology*, 4:907, 2013. 63

[11] Arslan Basharat, Alexei Gritai, and Mubarak Shah. Learning object motion patterns for anomaly detection and improved object detection. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008. 11

[12] Matthias Bauer and Andriy Mnih. Resampled priors for variational autoencoders. *International Conference on Artificial Intelligence and Statistics*, 2019. 11

[13] Loris Bazzani, Hugo Larochelle, and Lorenzo Torresani. Recurrent mixture density network for spatiotemporal visual attention. In *International Conference on Learning Representations*, 2017. 8, 37, 38

[14] Babak Ehteshami Bejnordi, Tijmen Blankevoort, and Max Welling. Batch-shaping for learning conditional channel gated networks. *International Conference on Learning Representations*, 2020. 13, 94

[15] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 97

[16] Guido Borghi, Marco Venturelli, Roberto Vezzani, and Rita Cucchiara. Poseidon: Face-from-depth for driver pose estimation. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2017. 9

[17] Ali Borji, Mengyang Feng, and Huchuan Lu. Vanishing point attracts gaze in free-viewing and visual search tasks. *Journal of Vision*, 16(14):18–18, 2016. 25

[18] Ali Borji and Laurent Itti. State-of-the-art in visual attention modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):185–207, 2013. 8

[19] Ali Borji and Laurent Itti. Cat2000: A large scale fixation dataset for boosting saliency research. *IEEE International Conference on Computer Vision and Pattern Recognition Workshops*, 2015. 9

[20] Ali Borji, Dicky N Sihite, and Laurent Itti. What/where to look next? modeling top-down visual attention in complex interactive environments. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 44(5):523–538, 2014. 8

[21] R. Brémond, J.-M. Auberlet, V. Cavallo, L. Désiré, V. Faure, S. Lemonnier, R. Lobjois, and J.-P. Tarel. Where we look when we drive: A multidisciplinary approach. In *Proceedings of Transport Research Arena*, Paris, France, 2014. 9, 10

[22] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann Lecun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations*, 2014. 125, 126

[23] Zoya Bylinskii, Tilke Judd, Ali Borji, Laurent Itti, Frédo Durand, Aude Oliva, and Antonio Torralba. Mit saliency benchmark, 2015. 9, 16, 20, 37, 39

[24] Zoya Bylinskii, Tilke Judd, Aude Oliva, Antonio Torralba, and Frédo Durand. What do different evaluation metrics tell us about saliency models? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 36

[25] Simone Calderara, Uri Heinemann, Andrea Prati, Rita Cucchiara, and Naftali Tishby. Detecting anomalies in people's trajectories using spectral graph analysis. *Computer Vision and Image Understanding*, 115(8):1099–1111, 2011. 12, 63

[26] Antoni Chan and Nuno Vasconcelos. Ucsd pedestrian database. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008. 77

[27] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *International Conference on Learning Representations*, 2019. 13, 95, 105

[28] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. In *Neural Information Processing Systems*, 2015. 17

[29] Zhiyuan Chen and Bing Liu. *Lifelong machine learning*. Morgan & Claypool Publishers, 2018. 93, 94

[30] Zhourong Chen, Yang Li, Samy Bengio, and Si Si. You look twice: Gaternet for dynamic filter selection in cnns. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2019. 13, 94

[31] M. M. Cheng, N. J. Mitra, X. Huang, P. H. S. Torr, and S. M. Hu. Global contrast based salient region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):569–582, March 2015. 8

[32] Hyunsun Choi and Eric Jang. Generative ensembles for robust anomaly detection. *arXiv preprint arXiv:1810.01392*, 2018. 87

[33] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *International Conference on Learning Representations*, 2016. 129

[34] Yang Cong, Junsong Yuan, and Ji Liu. Sparse reconstruction cost for abnormal event detection. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 3449–3456. IEEE, 2011. 10, 63, 64

[35] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2016. 115

[36] Marcella Cornia, Lorenzo Baraldi, Giuseppe Serra, and Rita Cucchiara. A Deep Multi-Level Network for Saliency Prediction. In *International Conference on Pattern Recognition*, 2016. 8, 37, 38

[37] Marcella Cornia, Lorenzo Baraldi, Giuseppe Serra, and Rita Cucchiara. Predicting human eye fixations via an lstm-based saliency attentive model. *IEEE Transactions on Image Processing*, 2016. 8

[38] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Neural Information Processing Systems*, 2016. 125, 131

[39] Allison Del Giorno, J Andrew Bagnell, and Martial Hebert. A discriminative framework for anomaly detection in large videos. In *European Conference on Computer Vision*, pages 334–349. Springer, 2016. 12

[40] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2009. 101

[41] Yong Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2015. 130

[42] Lior Elazary and Laurent Itti. A bayesian model for efficient visual search and recognition. *Vision Research*, 50(14):1338 – 1352, 2010. 8

[43] Philipp Fischer, Alexey Dosovitskiy, Eddy Ilg, Philip Häusser, Caner Hazırbaş, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. *IEEE International Conference on Computer Vision*, 2015. 121, 123

[44] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 19

[45] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 1999. 12, 93

[46] Yoav Freund and David Haussler. A fast and exact learning rule for a restricted class of boltzmann machines. *Neural Information Processing Systems*, 4:912–919, 1992. 11

[47] Lex Fridman, Philipp Langhans, Joonbum Lee, and Bryan Reimer. Driver gaze region estimation without use of eye movement. *IEEE Intelligent Systems*, 31(3):49–56, 2016. 9, 10, 19

[48] Simone Frintrop, Erich Rome, and Henrik I Christensen. Computational visual attention systems and their cognitive foundations: A survey. *ACM Transactions on Applied Perception*, 7(1):6, 2010. 7

[49] Björn Fröhlich, Markus Enzweiler, and Uwe Franke. Will this car change the lane?-turn signal recognition in the frequency domain. In *IEEE Intelligent Vehicles Symposium Proceedings*, pages 37–42. IEEE, 2014. 15

[50] A Gaidon, Q Wang, Y Cabon, and E Vig. Virtual worlds as proxy for multi-object tracking analysis. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2016. 119

[51] Dashan Gao, Vijay Mahadevan, and Nuno Vasconcelos. On the plausibility of the discriminant centersurround hypothesis for visual saliency. *Journal of Vision*, pages 1–18, 2008. 8

[52] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2012. 122

[53] Mathieu Germain et al. Made: Masked autoencoder for distribution estimation. In *International Conference on Machine Learning*, pages 881–889, 2015. 11, 12, 66

[54] Theodosios Gkamas and Christophoros Nikou. Guiding optical flow estimation using superpixels. In *International Conference on Digital Signal Processing*, pages 1–6. IEEE, 2011. 34, 36

[55] Stas Goferman, Lihi Zelnik-Manor, and Ayellet Tal. Context-aware saliency detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(10):1915–1926, October 2012. 8

[56] Rudolf Groner, Franziska Walder, and Marina Groner. Looking at faces: Local and global aspects of scanpaths. *Advances in Psychology*, 22:523–533, 1984. 20

[57] Stephanie Grubmüller, Jiri Plihal, and Pavel Nedoma. *Automated Driving from the View of Technical Standards*, pages 29–40. Springer International Publishing, Cham, 2017. 15

[58] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Neural Information Processing Systems*, 2017. 100, 106

[59] Mahmudul Hasan et al. Learning temporal regularity in video sequences. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2016. 10, 63, 64, 80

[60] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2016. 81, 101, 132

[61] John M Henderson. Human gaze control during real-world scene perception. *Trends in cognitive sciences*, 7(11):498–504, 2003. 20

[62] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Neural Information Processing Systems*, pages 6626–6637, 2017. 87

[63] Ryota Hinami et al. Joint detection and recounting of abnormal events by learning deep generic knowledge. In *IEEE International Conference on Computer Vision*, 2017. 11, 64, 80

[64] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Brian Kingsbury, and Tara Sainath. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Magazine*, 2012. 125

[65] X. Huang, C. Shen, X. Boix, and Q. Zhao. Salicon: Reducing the semantic gap in saliency prediction by adapting deep neural networks. In *IEEE International Conference on Computer Vision*, pages 262–270, Dec 2015. 8

[66] David I Shuman, Sunil K. Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 2012. 125

[67] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. *IEEE International Conference on Computer Vision and Pattern Recognition*, 2017. 123, 124

[68] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *International Conference on Machine Learning*, 2015. 97

[69] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, 2015. 129

[70] Radu Tudor Ionescu, Sorina Smeureanu, Bogdan Alexe, and Marius Popescu. Unmasking the abnormal events in video. *IEEE International Conference on Computer Vision*, 2017. 12, 80

[71] Laurent Itti and Pierre Baldi. Bayesian surprise attracts human attention. *Vision research*, 49(10):1295–1306, 2009. 64

[72] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Neural Information Processing Systems*, 2015. 120, 121

[73] Ashesh Jain, Hema S Koppula, Bharad Raghavan, Shane Soh, and Ashutosh Saxena. Car that knows before you do: Anticipating maneuvers via learning temporal driving models. In *IEEE International Conference on Computer Vision*, pages 3182–3190, 2015. 15

[74] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *International Conference on Learning Representations*, 2017. 97

[75] Ming Jiang, Shengsheng Huang, Juanyong Duan, and Qi Zhao. Salicon: Saliency in context. In *IEEE International Conference on Computer Vision and Pattern Recognition*, June 2015. 9, 16

[76] Thorsten Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. Technical report, Cornell University, 1996. 131

[77] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014. 28

[78] Qiuhong Ke, Mohammed Bennamoun, Senjian An, Ferdous Ahmed Sohel, and Farid Boussaïd. A new representation of skeleton sequences for 3d action recognition. *IEEE International Conference on Computer Vision and Pattern Recognition*, 2017. 130

[79] Jaechul Kim and Kristen Grauman. Observe locally, infer globally: a space-time mrf for detecting abnormal activities with incremental updates. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 2921–2928. IEEE, 2009. 11, 80

[80] Tae Soo Kim and Austin Reiter. Interpretable 3d human action analysis with temporal convolutional networks. In *IEEE International Conference on Computer Vision and Pattern Recognition Workshops*, 2017. 130

[81] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2014. 36, 74, 106, 122, 129

[82] Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Neural Information Processing Systems*, pages 4743–4751, 2016. 12, 87

[83] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *International Conference on Learning Representations*, 2014. 66, 74, 85

[84] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. 125, 127

[85] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 2017. 12, 94

[86] Teuvo Kohonen. *Self-organization and associative memory*, volume 8. Springer Science & Business Media, 2012. 64

[87] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009. 87

[88] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009. 129

[89] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 101

[90] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Neural Information Processing Systems*, 2012. 125

[91] Ajay Kumar. Computer-vision-based fabric defect detection: A survey. *IEEE Transactions on Industrial Electronics*, 55(1):348–363, 2008. 63

[92] Puneet Kumar, Mathias Perrollaz, Stéphanie Lefevre, and Christian Laugier. Learning-based approach for online lane change intention prediction. In *IEEE Intelligent Vehicles Symposium Proceedings*, pages 797–802. IEEE, 2013. 15

[93] Matthias Kümmerer, Lucas Theis, and Matthias Bethge. Deep gaze I: boosting saliency prediction with feature maps trained on imagenet. In *International Conference on Learning Representations Workshops*, 2015. 8

[94] Matthias Kümmerer, Thomas SA Wallis, and Matthias Bethge. Information-theoretic model comparison unifies saliency metrics. *Proceedings of the National Academy of Sciences*, 112(52):16054–16059, 2015. 36

[95] Junseok Kwon and Kyoung Mu Lee. A unified framework for event summarization and rare event detection from multiple views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1737–1750, 2015. 11

[96] Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. In *International Conference on Artificial Intelligence and Statistics*, pages 29–37, 2011. 11, 66

[97] Adam M Larson and Lester C Loschky. The contributions of central versus peripheral vision to scene gist recognition. *Journal of Vision*, 9(10):6–6, 2009. 43

[98] Yann LeCun, Corinna Cortes, and Christopher J.C. Burges. The MNIST database of handwritten digits, 1998. 100

[99] Chaolong Li, Zhen Cui, Wenming Zheng, Chunyan Xu, and Jian Yang. Spatio-temporal graph convolution for skeleton based action recognition. In *AAAI Conference on Artificial Intelligence*, 2018. 130

[100] Weixin Li, Vijay Mahadevan, and Nuno Vasconcelos. Anomaly detection and localization in crowded scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(1):18–32, 2014. 11

[101] Zhizhong Li and Derek Hoiem. Learning without forgetting. In *European Conference on Computer Vision*. Springer, 2016. 12, 94, 103

[102] Zhouhan Lin, Roland Memisevic, and Kishore Reddy Konda. How far can we go without convolution: Improving fully-connected networks. *IEEE International Conference on Computer Vision and Pattern Recognition Workshops*, 2016. 131, 132

[103] Jun Liu, Amir Shahroudy, Dong Xu, and Gang Wang. Spatio-temporal lstm with trust gates for 3d human action recognition. In *European Conference on Computer Vision*, 2016. 130

[104] Nian Liu, J. Han, D. Zhang, Shifeng Wen, and T. Liu. Predicting eye fixations using convolutional neural networks. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 362–370, June 2015. 8

[105] Wen Liu et al. Future frame prediction for anomaly detection – a new baseline. *IEEE International Conference on Computer Vision and Pattern Recognition*, 2018. 10, 63, 64, 80

[106] Ziwei Liu, Raymond Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *IEEE International Conference on Computer Vision*, 2017. 123, 124

[107] Gucan Long, Laurent Kneip, Jose M Alvarez, Hongdong Li, Xiaohu Zhang, and Qifeng Yu. Learning image matching by simply watching video. In *European Conference on Computer Vision*, 2016. 123

[108] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Neural Information Processing Systems*, 2017. 13, 94

[109] David G Lowe. Object recognition from local scale-invariant features. In *IEEE International Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1150–1157. Ieee, 1999. 19

[110] Cewu Lu, Jianping Shi, and Jiaya Jia. Abnormal event detection at 150 fps in matlab. In *IEEE International Conference on Computer Vision*, pages 2720–2727. IEEE, 2013. 10

[111] Pauline Luc, Natalia Neverova, Camille Couprie, Jacob Verbeek, and Yann LeCun. Predicting deeper into the future of semantic segmentation. *IEEE International Conference on Computer Vision*, 2017. 66

[112] Weixin Luo et al. A revisit of sparse coding based anomaly detection in stacked rnn framework. *IEEE International Conference on Computer Vision*, 2017. 10, 63, 77, 80

[113] Weixin Luo, Wen Liu, and Shenghua Gao. Remembering history with convolutional lstm for anomaly detection. In *IEEE International Conference on Multimedia and Expo*, pages 439–444. IEEE, 2017. 80

[114] Yu-Fei Ma and Hong-Jiang Zhang. Contrast-based image attention analysis by using fuzzy growing. In *ACM International Conference on Multimedia*, MULTIMEDIA '03, pages 374–381, New York, NY, USA, 2003. ACM. 8

[115] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *International Conference on Learning Representations*, 2017. 97

[116] Vijay Mahadevan, Weixin Li, Viral Bhalodia, and Nuno Vasconcelos. Anomaly detection in crowded scenes. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1975–1981. IEEE, 2010. 11, 80

[117] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *European Conference on Computer Vision*, 2018. 13

[118] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2018. 13, 94, 98

[119] SK Mannan, KH Ruddock, and DS Wooding. Fixation sequences made during visual examination of briefly presented 2d images. *Spatial vision*, 11(2):157–178, 1997. 20

[120] Nicolas Y Masse, Gregory D Grant, and David J Freedman. Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. *Proceedings of the National Academy of Sciences*, 2018. 13

[121] Thomas Mauthner, Horst Possegger, Georg Waltner, and Horst Bischof. Encoding based saliency detection for videos and images. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 2494–2502, 2015. 8

[122] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*. Elsevier, 1989. 12, 93

[123] Ramin Mehran, Alexis Oyama, and Mubarak Shah. Abnormal crowd behavior detection using social force model. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 935–942. IEEE, 2009. 12

[124] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *International Conference on Learning Representations*, 2013. 131

[125] Brendan Morris, Anup Doshi, and Mohan Trivedi. Lane change intent prediction for driver assistance: On-road design and evaluation. In *IEEE Intelligent Vehicles Symposium Proceedings*, pages 895–901. IEEE, 2011. 15

[126] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2017. 17

[127] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative models know what they don't know? *International Conference on Learning Representations*, 2019. 87

[128] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. *Neural Information Processing Systems Workshops*, 2011. 101

[129] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *IEEE International Conference on Computer Vision*, 2017. 115

[130] Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. *International Conference on Learning Representations*, 2018. 12

[131] David Nilsson and Cristian Sminchisescu. Semantic video segmentation by gated recurrent flow propagation. *IEEE International Conference on Computer Vision and Pattern Recognition*, 2018. 17

[132] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *International Conference on Machine Learning*, 2017. 112

[133] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016. 66

[134] Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Patrick Jahnichen, and Moin Nabi. Learning to remember: A synaptic plasticity driven framework for continual learning. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2019. 13, 93, 101, 106, 109

[135] Andrea Palazzi, Francesco Solera, Simone Calderara, Stefano Alletto, and Rita Cucchiara. Where should you attend while driving? In *IEEE Intelligent Vehicles Symposium Proceedings*, 2017. 37, 38, 39

[136] Pingbo Pan, Zhongwen Xu, Yi Yang, Fei Wu, and Yueting Zhuang. Hierarchical recurrent neural encoder for video representation with application to captioning. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1029–1038, 2016. 28

[137] George Papamakarios, Iain Murray, and Theo Pavlakou. Masked autoregressive flow for density estimation. In *Neural Information Processing Systems*, pages 2335–2344, 2017. 12

[138] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, 2017. 63

[139] Jeffrey S Perry and Wilson S Geisler. Gaze-contingent real-time simulation of arbitrary visual fields. In *Human vision and electronic imaging*, volume 57, 2002. 43, 45

[140] Robert J Peters and Laurent Itti. Beyond bottom-up: Incorporating task-dependent influences into a computational model of spatial attention. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007. 8

[141] Robert J Peters and Laurent Itti. Applying computational tools to predict gaze direction in interactive visual environments. *ACM Transactions on Applied Perception*, 5(2):9, 2008. 8

[142] Oluwatoyin P Popoola and Kejun Wang. Video-based abnormal human behavior recognition—a review. *IEEE Transactions on Systems, Man and Cybernetics*, 42(6):865–878, 2012. 12

[143] Michael I Posner, Robert D Rafal, Lisa S Choate, and Jonathan Vaughan. Inhibition of return: Neural basis and function. *Cognitive Neuropsychology*, 2(3):211–228, 1985. 20

[144] N Pugeault and RICHARD Bowden. How much of driving is preattentive? *IEEE Transactions on Vehicular Technology*, 2015. 9, 10, 19

[145] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2017. 123, 124

[146] Mahdyar Ravanbakhsh, Enver Sangineto, Moin Nabi, and Nicu Sebe. Training adversarial discriminators for cross-channel abnormal event detection in crowds. *IEEE Winter Conference on Applications of Computer Vision*, 2019. 11

[147] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2017. 13, 94, 95, 105

[148] Scott Reed, Yutian Chen, Thomas Paine, Aäron van den Oord, SM Eslami, Danilo Rezende, Oriol Vinyals, and Nando de Freitas. Few-shot autoregressive density estimation: Towards learning to learn distributions. *International Conference on Learning Representations*, 2018. 12

[149] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2015. 123, 124

[150] Stephan R Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for benchmarks. In *IEEE International Conference on Computer Vision*, 2017. 119

[151] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *International Conference on Learning Representations*, 2019. 13

[152] Mark B Ring. CHILD: A first step towards continual learning. *Machine Learning*, 1997. 93

[153] Joceline Rogé, Thierry Pébayle, Elina Lambilliotte, Florence Spitzenstetter, Daniele Giselbrecht, and Alain Muzet. Influence of age, speed and duration of monotonous driving task in traffic on the driver's useful visual field. *Vision Research*, 44(23):2737–2744, 2004. 26

[154] Mehrsan Javan Roshtkhari and Martin D Levine. Online dominant and anomalous behavior detection in videos. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 2611–2618. IEEE, 2013. 12

[155] Dmitry Rudoy, Dan B Goldman, Eli Shechtman, and Lihi Zelnik-Manor. Learning video saliency from human gaze using candidate selection. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2013. 8

[156] R. Rukšėnas, J. Back, P. Curzon, and A. Blandford. Formal modelling of salience and cognitive load. *Electronic Notes in Theoretical Computer Science*, 208:57 – 75, 2008. 20

[157] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016. 12, 98

[158] Mohammad Sabokrou, Mohsen Fayyaz, Mahmood Fathy, Zahra. Moayed, and Reinhard Klette. Deep-anomaly: Fully convolutional neural network for fast anomaly detection in crowded scenes. *Computer Vision and Image Understanding*, 2018. 11

[159] Mohammad Sabokrou, Mohammad Khalooei, Mahmood Fathy, and Ehsan Adeli. Adversarially learned one-class classifier for novelty detection. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 3379–3388, 2018. 10, 63, 64

[160] Jill Sardegna, Susan Shelly, and Scott Steidl. *The encyclopedia of blindness and vision impairment*. Infobase Publishing, 2002. 25

[161] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International Conference on Information Processing in Medical Imaging*, pages 146–157. Springer, 2017. 11, 63, 75

[162] Bernhard Schölkopf, Robert C Williamson, Alex J Smola, John Shawe-Taylor, and John C Platt. Support vector method for novelty detection. In *Neural Information Processing Systems*, 2000. 74

[163] Jonathan Schwarz, Jelena Luketina, Wojciech M Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. *International Conference on Machine Learning*, 2018. 103

[164] B. Schölkopf, J. Platt, and T. Hofmann. *Graph-Based Visual Saliency*, pages 545–552. MIT Press, 2007. 8

[165] Joan Serrà, Dídac Surís, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. *International Conference on Machine Learning*, 2018. 13, 94, 103, 106

[166] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+d: A large scale dataset for 3d human activity analysis. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2016. 129, 130

[167] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *International Conference on Learning Representations*, 2017. 13

[168] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Neural Information Processing Systems*, 2017. 13

[169] L. Simon, J. P. Tarel, and R. Bremond. Alerting the drivers about road signs with poor visual saliency. In *IEEE Intelligent Vehicles Symposium Proceedings*, pages 48–53, June 2009. 9, 10, 19

[170] Paul Smolensky. Information processing in dynamical systems: Foundations of harmony theory. Technical report, Department of Computer Science, Colorado University, 1986. 11

[171] Cristian Sminchisescu Stefan Mathe. Actions in the eye: Dynamic gaze datasets and learnt saliency models for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37, 2015. 8, 9, 37, 38

[172] Felipe Petroski Such, Shagan Sah, Miguel Domínguez, Suhas Pillai, Chao Zhang, Andrew Michael, Nathan D. Cahill, and Raymond W. Ptucha. Robust spatial filtering with graph convolutional neural networks. *IEEE Journal of Selected Topics in Signal Processing*, 2017. 131, 132

[173] Benjamin W Tatler. The central fixation bias in scene viewing: Selecting an optimal viewing position independently of motor biases and image feature distributions. *Journal of Vision*, 7(14):4–4, 2007. 24

[174] Benjamin W. Tatler, Mary M. Hayhoe, Michael F. Land, and Dana H. Ballard. Eye guidance in natural vision: Reinterpreting salience. *Journal of Vision*, 11(5), May 2011. 10, 17

[175] A. Tawari and M. M. Trivedi. Robust and continuous estimation of driver gaze zone by dynamic analysis of multiple face videos. In *IEEE Intelligent Vehicles Symposium Proceedings*, 2014. 9

[176] Graham W. Taylor, Rob Fergus, Yann LeCun, and Christoph Bregler. Convolutional learning of spatio-temporal features. In *European Conference on Computer Vision*, 2010. 125

[177] Jan Theeuwes. Top–down and bottom–up control of visual selection. *Acta Psychologica*, 135(2):77–99, 2010. 8

[178] Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. *International Conference on Learning Representations*, 2016. 76

[179] Jakub M Tomczak and Max Welling. Vae with a vamp prior. *International Conference on Artificial Intelligence and Statistics*, 2018. 11, 87

[180] Antonio Torralba, Aude Oliva, Monica S Castelhano, and John M Henderson. Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search. *Psychological review*, 113(4):766, 2006. 8

[181] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *IEEE International Conference on Computer Vision*. IEEE, 2015. 28, 29, 30

[182] Anne M Treisman and Garry Gelade. A feature-integration theory of attention. *Cognitive Psychology*, 12(1):97–136, 1980. 7

[183] Myron Tribus. *Thermostatics and thermodynamics: an introduction to energy, information and states of matter, with engineering applications*. van Nostrand, CS7, 1961. 63, 64

[184] Yoshiyuki Ueda, Yusuke Kamakura, and Jun Saiki. Eye movements converge on vanishing points during visual search. *Japanese Psychological Research*, 59(2):109–121, 2017. 25

[185] Geoffrey Underwood, Katherine Humphrey, and Editha van Loon. Decisions about objects in real-world scenes are influenced by visual saliency

before and during their inspection. *Vision Research*, 51(18):2031 – 2038, 2011. 9, 10, 16, 19

[186] Benigno Uria, Iain Murray, and Hugo Larochelle. Rnade: The real-valued neural autoregressive density-estimator. In *Neural Information Processing Systems*, pages 2175–2183, 2013. 11, 66

[187] Benigno Uria, Iain Murray, and Hugo Larochelle. A deep and tractable density estimator. In *International Conference on Machine Learning*, pages 467–475, 2014. 11, 12, 66

[188] Gido M van de Ven and Andreas S Tolias. Three scenarios for continual learning. *Neural Information Processing Systems Workshops*, 2018. 94, 96, 100

[189] A. van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, and A. Graves. Conditional image generation with pixelcnn decoders. In *Neural Information Processing Systems*, 2016. 11, 12, 66, 75

[190] A. van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *International Conference on Machine Learning*, 2016. 11, 12, 66, 68

[191] Andreas Veit and Serge Belongie. Convolutional networks with adaptive inference graphs. In *European Conference on Computer Vision*, 2018. 13

[192] Raviteja Vemulapalli, Felipe Arrate, and Rama Chellappa. Human action recognition by representing 3d skeletons as points in a lie group. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2014. 130

[193] Francisco Vicente, Zehua Huang, Xuehan Xiong, Fernando De la Torre, Wende Zhang, and Dan Levi. Driver gaze tracking and eyes off the road detection system. *IEEE Transactions on Intelligent Transportation Systems*, 16(4), 2015. 9

[194] Panqu Wang, Pengfei Chen, Ye Yuan, Ding Liu, Zehua Huang, Xiaodi Hou, and Garrison Cottrell. Understanding convolution for semantic segmentation. *IEEE Winter Conference on Applications of Computer Vision*, 2018. 17

[195] Panqu Wang and Garrison W Cottrell. Central and peripheral vision for scene recognition: A neurocomputational modeling explorationwang & cottrell. *Journal of Vision*, 17(4):9–9, 2017. 43, 44

[196] Wenguan Wang, Jianbing Shen, and Fatih Porikli. Saliency-aware geodesic video object segmentation. In *IEEE International Conference on Computer Vision and Pattern Recognition*, 2015. 8, 37, 38

[197] Wenguan Wang, Jianbing Shen, and Ling Shao. Consistent video saliency using local gradient flow optimization and global refinement. *IEEE Transactions on Image Processing*, 24(11):4185–4196, 2015. 8, 37, 38

[198] Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In *European Conference on Computer Vision*, 2018. 13

[199] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. DeepFlow: Large displacement optical flow with deep matching. In *IEEE International Conference on Computer Vision*, 2013. 123, 124

[200] Jeremy M Wolfe. Visual search. *Attention*, 1:13–73, 1998. 7

[201] Jeremy M. Wolfe, Kyle R. Cave, and Susan L. Franzel. Guided search: an alternative to the feature integration model for visual search. *Journal of Experimental Psychology: Human Perception & Performance*, 1989. 8

[202] Chenshen Wu, Luis Herranz, Xialei Liu, Joost van de Weijer, Bogdan Raducanu, et al. Memory replay gans: Learning to generate new categories without forgetting. In *Neural Information Processing Systems*, 2018. 13

[203] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *International Conference on Learning Representations*, 2016. 17, 26, 34, 36, 115

[204] Jason J Yu, Adam W Harley, and Konstantinos G Derpanis. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. *European Conference on Computer Vision Workshops*, 2016. 123

[205] Matthew Zeiler and Robert Fergus. Stochastic pooling for regularization of deep convolutional neural networks. In *International Conference on Learning Representations*, 2013. 132

[206] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833. Springer, 2014. 80

[207] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2017. 12

[208] Yun Zhai and Mubarak Shah. Visual attention detection in video sequences using spatiotemporal cues. In *ACM International Conference on Multimedia*, MM '06, pages 815–824, New York, NY, USA, 2006. ACM. 8

[209] Tong Zhang, Wenming Zheng, Zhen Cui, and Yang Li. Tensor graph convolutional neural network. *arXiv preprint arXiv:1803.10071*, 2018. 130

[210] Bin Zhao, Li Fei-Fei, and Eric P Xing. Online detection of unusual events in videos via dynamic sparse coding. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 3313–3320. IEEE, 2011. 10

[211] Hua Zhong, Jianbo Shi, and Mirkó Visontai. Detecting unusual activity in video. In *IEEE International Conference on Computer Vision and Pattern Recognition*, volume 2, pages II–II. IEEE, 2004. 12

[212] Sheng-hua Zhong, Yan Liu, Feifei Ren, Jinghuan Zhang, and Tongwei Ren. Video saliency detection via dynamic consistent spatio-temporal attention modelling. In *AAAI Conference on Artificial Intelligence*, 2013. 8

[213] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International Conference on Learning Representations*, 2018. 11, 64