Weakly supervised learning for image classification and potentially moving obstacles analysis

Florent Chiaroni

▶ **To cite this version:**

Florent Chiaroni. Weakly supervised learning for image classification and potentially moving obstacles analysis. Signal and Image processing. Université Paris-Saclay, 2020. English. NNT : 2020UPASC006 . tel-02881904

HAL Id: tel-02881904
https://theses.hal.science/tel-02881904

Submitted on 26 Jun 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Thèse de doctorat

NNT: 2020UPASC006

Weakly supervised learning for image classification and potentially moving obstacles analysis

Thèse de doctorat de l'Université Paris-Saclay

École doctorale n° 580, Sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat: Traitement du signal et de l'image
Unité de recherche: Université Paris-Saclay, CNRS, CentraleSupélec, Laboratoire des signaux et systèmes, 91190, Gif-sur-Yvette, France
Référent: CentraleSupelec

Thèse présentée et soutenue à Gif-sur-Yvette, le 3 février 2020, par

Florent CHIARONI

Composition du jury:

Jean-Luc Dugelay Professeur, EURECOM	Président du jury
Yap-Peng Tan Professeur, Nanyang Technological University (NTU), Singapore	Rapporteur
Hichem Sahbi Chargé de recherche CNRS (HDR), UPMC Sorbonne Université (LIP6)	Rapporteur
Samia Bouchafa Professeur, Université d'Evry Val-d'Essonne (IBISC)	Examinatrice
Camille Couprie Chercheur, FACEBOOK (FAIR)	Examinatrice
Frédéric Dufaux Directeur de recherche CNRS, CentraleSupelec, Université Paris-Saclay (L2S)	Directeur de thèse
Mohamed-Cherif Rahal Chercheur, Institut VEDECOM (Délégation de conduite, perception du véhicule)	Coencadrant
Nicolas Hueber Chercheur, Institut Saint-Louis franco-allemand ISL (ELSI)	Coencadrant
Féthi Ben Ouezdou Professeur, Université de Versailles Saint-Quentin en Yvelines	Invité

Acknowledgements

Premièrement, je remercie les entités sans lesquelles ce projet de thèse n'aurait pas pu aboutir. Je remercie l'institut VEDECOM pour avoir financé ce projet, ainsi que le L2S et l'ISL pour leurs collaborations. Plus précisément, je remercie le directeur scientifique de VEDECOM Féthi Ben Ouezdou pour avoir favorisé le bon commencement de ma thèse. Je remercie aussi Gilles Duc, le directeur de mon pôle d'affectation à l'école doctorale STIC, pour m'avoir orienté vers les bonnes personnes lors de ma recherche d'un directeur de thèse. Je remercie Frédéric pour avoir accepté de devenir mon directeur de thèse et pour avoir cru en mon projet doctoral dès le début. Je remercie mon encadrant Mohamed pour avoir motivé le lancement de ce sujet de thèse au sein de l'institut VEDECOM. Je remercie Nicolas et Pierre Raymond, respectivement mon co-encadrant et le responsable de l'équipe ELSI de l'ISL, pour avoir collaboré dans le cadre de ce projet.

Aussi, je remercie l'ensemble des membres de mon jury pour avoir accepté de participer à ma soutenance de thèse, en apportant leur esprit critique et expert. Je remercie Jean-Luc Dugelay pour avoir présidé la soutenance, Yap-Peng Tan et Hichem Sahbi pour leurs relectures et retours détaillés sur mon manuscrit, les examinatrices Samia Bouchafa et Camille Couprie pour leur expertise et remarques très pertinentes, ainsi que Féthi Ben Ouezdou pour avoir participé en tant qu'invité.

Ensuite, je souhaite grandement remercier mes trois encadrants, tant d'un point de vue professionnel que personnel. Je remercie mon directeur de thèse du L2S Frédéric. Il m'a permis de réaliser cette thèse dans les meilleures conditions d'encadrement possibles, en se rendant disponible du début jusqu'à la fin. Concernant la philosophie dans laquelle il m'a encadré, l'adage "Il vaut mieux apprendre à pêcher que donner du poisson." pourrait bien la résumer. En me transmettant ses méthodologies de travail et de recherche, il m'a permis de passer les différentes étapes de ma thèse avec confiance. Il m'a offert une grande liberté et autonomie dans les directions à prendre, tout en gardant une

vision bien définie sur plusieurs échelles de temps afin d'obtenir, ce qu'il faut, lorsqu'il le faut. Mais au-delà d'un directeur de thèse formateur et disponible, il a aussi été une superbe personne à la fois chaleureuse et prévenante. Je ne compte plus ses nombreux et riches conseils, ainsi que tous ces rires partagés. Tu es et resteras une référence pour la suite, tant pour ton encadrement que pour ta personne. Je remercie aussi mon encadrant Mohamed, pour avoir motivé et financé au travers de l'institut VEDECOM mon projet doctoral durant trois ans, et pour avoir si bien valorisé et défendu mes travaux de recherche du début jusqu'à la fin, d'un point de vue à la fois scientifique et industriel. Toujours très enthousiaste et positif, sa confiance n'a jamais failli, et il m'a ainsi permis de guider ma barque jusqu'à bon port. Enfin, je remercie mon encadrant de l'ISL Nicolas pour avoir suivi ma thèse de près malgré la distance. Son implication au travers de sa disponibilité, ses conseils techniques, et son soutien pendant ces trois années d'échanges m'ont été très enrichissants. Je le remercie pour m'avoir partagé son énergie débordante qui a été une bonne source de motivation. Prendre exemple sur sa proactivité me sera très bénéfique pour la suite. Et bien entendu, je le remercie aussi pour m'avoir fait découvrir avec passion sa belle région qu'est l'Alsace, ainsi que sa riche gastronomie.

Concernant l'institut VEDECOM, je remercie en premier lieu toute l'équipe, anciennement appelée VEH08 et appartenant aujourd'hui au domaine de délégation de conduite et connectivité dirigée par Guillaume, pour m'avoir bien accueilli durant ma thèse. J'ai eu la chance de voir pendant ces trois années une croissance continue de cette équipe. Cela m'a fait réaliser que, contribuer au véhicule autonome de demain n'ai pas une chose aisée, mais que cela reste possible lorsque la motivation et l'engagement de chacun perdurent. Je remercie chacune des personnes de cette équipe pour ces échanges continuels durant trois ans. Je remercie aussi les équipes de Sami, Bertrand, Jean Croque et Aziz pour toutes ces discussions, collaborations et entre aides, mais aussi l'ensemble des autres collaborateurs, stagiaires et doctorants, avec une attention particulière portée à Laurène et Alexis pour cette expérience de la thèse partagée. D'autres personnes qui ont été de passage à VEDECOM m'ont aussi beaucoup apporté pendant ce projet, telles que Hatem, Ghazalleh, Clément, Mathieu, et Sid Ali mon ancien super stagiaire.

Je remercie aussi l'ensemble de l'équipe ELSI, avec une particulière attention pour Pierre, son accueil chaleureux et sa répartie humoristique infaillible, pour m'avoir si bien intégré à chacune de mes venues malgré la distance. Ce fût

un grand plaisir de collaborer avec vous et de partager ensemble toutes ces flamenkuches à volonté. Je garderai un excellent souvenir de toute cette belle et dynamique équipe ELSI pleine de vie.

Concernant le L2S, je remercie l'ensemble de l'équipe de Frédéric et mes autres collègues de bureau. Commençons par Giuseppe, à la fois cuisinier, cultivateur de tomates cerises et organisateur journalier de la pause-café. Tout ces échanges avec lui m'ont beaucoup apporté sur le plan scientifique, par exemple pour ses conseils rédactionnels, ses idées et sa vision, mais aussi sur le plan personnel grâce à sa personnalité débordante d'enthousiasme et d'humour en toutes circonstances. Il y a aussi Maurice, un doctorant calme et posé investi dans son sujet de thèse, mais aussi expert en finances et investissements pendant son temps libre. Je suis toujours sorti de nos discussions avec des connaissances pratiques en plus. Il y a aussi Milan, un travailleur acharné *no pain, no gain*, féru de sport et testeur des diètes incongrues. Je le remercie pour ces échanges autour de sujets divers, en plus de nos discussions purement scientifiques dans l'objectif d'avancer respectivement sur notre thèse bien entendu. Je remercie aussi Li et Chen, dont j'ai eu le plaisir de découvrir la culture bien différente, au travers d'échanges très agréables et enrichissants qui nous ont aussi parfois amenés à des petits quiproquos bien rigolos. Enfin, je remercie aussi mes fidèles collègues de bureau Oumaima et Violetta, toujours présentes pour partager des petits goûters originaux, en parlant de photographie ou bien de clustering. Non loin de là, à la porte juste à côté, je remercie aussi Mourad et Younoussa pour leur constante bonne humeur. Je vous remercie chacun encore une fois pour m'avoir aidé à organiser mon pot de thèse.

Par ailleurs, je remercie ma famille et mes proches pour tout leur amour et leur soutien, en commençant par mes parents, mes deux oncles Angeot et Denis, mon cousin Laurent et ma cousine Maryn, mais aussi mon ami Yvan que je connais depuis aussi longtemps que je m'en souvienne. Et enfin, comment pourrais-je oublier de mentionner cette personne m'ayant accompagné au plus près durant ces trois années et bien plus ? Je remercie tout particulièrement Mercedes, pour son soutien et son dévouement continuels jusqu'à l'arrivée. Et c'est d'ailleurs en grande partie grâce à elle que beaucoup se sont régalés à mon pot de thèse, concluant si bien cette étape.

Abstract

In the context of autonomous vehicle perception applications, the interest of the research community for deep learning approaches has continuously grown since the last decade. This can be explained by the fact that deep learning techniques provide nowadays state-of-the-art prediction performances for several computer vision challenges. More specifically, deep learning techniques can provide rich semantic information concerning the complex visual patterns encountered in autonomous driving scenarios, from several kinds of input data and under various weather conditions.

However, such approaches require, as their name implies, to learn on data. In particular, state-of-the-art prediction performances on discriminative tasks often demand hand labeled data of the target application domain. Hand labeling has a significant cost, while, conversely, unlabeled data can be easily obtained in the autonomous driving context. It turns out that a category of learning strategies, referred to as weakly supervised learning, enables to exploit partially labeled data. Therefore, we aim in this thesis at reducing as much as possible the hand labeling requirement by proposing weakly supervised learning techniques.

We start by presenting a type of learning methods which are self-supervised. They consist of substituting hand-labels by upstream techniques able to automatically generate exploitable training labels. Self-supervised learning (SSL) techniques have proven their usefulness in the past for offroad obstacles avoidance and path planning through changing environments, by learning at the application time. More recently, they have also been applied for depth map estimation, asphalt road segmentation, and moving obstacles instance segmentation and tracking. However, SSL techniques still leave the door open for detection, segmentation, and classification of static potentially moving obstacles. For instance, the latter can be motionless cars at a road intersection, or pedestrians waiting to cross the street. Consequently, we propose in this thesis three novel weakly supervised learning methods with the final goal to deal with such road users through an SSL framework.

The first two proposed contributions of this work deal with partially labeled image classification datasets, such that the labeling effort can be focused only on our class of interest, the positive class. Then, we propose an approach which deals with training data containing a high fraction of wrong labels, referred to as noisy labels. Next, we demonstrate the potential of such weakly supervised image classification strategies for the two following real application tasks: detection and segmentation of potentially moving obstacles.

Contents

List of Figures

Chapter 1

Introduction

Contents

1.1 Context and Objectives: Towards a rich autonomous vehicle environment perception

The vision of urban autonomous driving applications has progressively become a reality since the last decades. For instance, several companies and research institutes are actively working on this novel type of technological mobility. Additionally, the potential introduction of autonomous vehicles in our daily life has also motivated recent studies on the societal impacts both in terms of rights and ethics [101].

1.1.1 Context: Autonomous driving implies autonomous perception

Before considering the potential autonomous driving technologies as mature, we need to build them up, showcase their usefulness, and more importantly, demonstrate their safety. The latter point includes a wide range variety of conditions. Generally speaking, a safe autonomous vehicle implies that it is able to undertake appropriate decisions on its own. In turn, this entails upstream to understand the given environment and situation. From a perception point of view, this task can be referred to as semantic scene understanding [32]. Scene understanding can be performed using different sources of information, including the sensor information of the ego-vehicle as illustrated in Fig. 1.1.

Figure 1.1: VEDECOM autonomous vehicle equipped with integrated lidar, radar and camera sensors.

Although fusing the information coming from several and varied sensor sources is in practice very impactful to improve perception performances, the contributions of this study focus on the use of a single monocular camera. This sensor presents, for potential industrial applications, a lower cost both in terms of fabrication and usage. Moreover, our goal is to provide semantic information and it turns out that image and video can provide dense visual information in comparison to low-cost sparse and dispersive LIDAR sensors.

1.1.2 Objectives

Potentially moving obstacles analysis using deep learning: Several semantic scene understanding tasks of broad interest include scene context classification, road segmentation, traffic sign and lane marking detection. In our case, we focus our attention on the urban road users that we refer to as *potentially moving obstacles*, for instance pedestrians, cars, or motorcyclists. It is of major importance to correctly analyze them, as they interact with the ego-vehicle and hence directly influence the ego-vehicle path planning decisions [30]. One of the main difficulty is to ensure that the perception model designed for this task can correctly analyze all encountered potentially moving obstacles, despite their complex and varied visual shape patterns, and under varying conditions such as moving observation point of view or changing weather.

Deep learning approaches [63] present nowadays state-of-the-art prediction performances for several semantic visual pattern analysis tasks [115]. They are essentially based on a high-level feature representation of the sensor data to analyze. Visual deep learning techniques generally use several consecutive layers of convolutional filters. Historically, these architectures have been inspired from [76]. First, they have been called neocognitron [55]

before being democratized and rendered functional thanks to backpropagation algorithms during the last two decades under the name convolutional neural networks (CNNs) [95].

Current CNN baseline methods are trained on fully labeled datasets to perform state-of-the-art prediction performances exposed on online available benchmark computer vision datasets such as ImageNet [93] for image classification, COCO dataset [102] for object detection, and ADE20K dataset [179] for semantic segmentation task [48]. Image classification associates a label class to a given image, object detection predicts bounding boxes of the objects to detect, and pixelwise semantic segmentation independantly classifies pixels of a given image. Fig. 1.2 illustrates these three key visual perception tasks by presenting real-time output predictions of the deep CNN model MultiNet [162], on Kitti dataset [59]. This vision dataset is specifically designed for autonomous vehicle perception.

Figure 1.2: MultiNet real-time predictions for road-type classification, vehicle detection and road segmentation.

Limitations of deep learning techniques: However, such baseline CNN methods also present the following drawback: They require to be trained on labeled data following the distribution of the target domain data to predict. If this requirement is not satisfied, then it is possible to obtain critical and unintended decreasing prediction performances as illustrated in Fig. 1.3 for MultiNet predictions. This phenomenon is also referred to as overfitting issue. We deduce that the learning model naturally specializes its generalization capacity of prediction on the training data distribution. As providing an adequate labeling effort has a significant cost, we propose to investigate solutions to overcome this problem.

The computational cost of deep learning techniques is another aspect of broad interest for real-time embedded applications. Computational cost of a given CNN architecture can be drastically reduced, for instance by applying pruning [69] or binarization [139] techniques on the deep learning model connections, weights parameters and predicted feature maps. However, this issue is not discussed in the scope of this thesis.

Improving the generalization model capacity using Transfer Learning: Nowadays, several strategies can reasonably attenuate the requirement of labeled data by improving the learning model capacity to generalize high-level feature concepts. For instance, a technique consists of pre-training the model on a generic dataset like ImageNet, before fine-tuning

(a) (b)

Figure 1.3: MultiNet overfitting issue for road segmentation task: A lack of generalization capacity when not trained on the target domain data. These images present MultNiet road segmentation pixel-level predictions on a Kitti dataset image (a) and on an image of a desert road (b).

on data of the target application domain, as proposed in [145] and [171]. While it enables to adapt a learning model to a novel data distribution in order to perform the same prediction task, transfer learning also allows adapting a pre-trained-model to a novel prediction task, as proposed in [157].

Reducing the need of hand labeled training data using Weakly Supervised Learning: Weakly Supervised Learning strategies have also been proposed to address the same lack of hand-labeled training data. Weakly supervised learning techniques focus on the ability to directly deal with a weakly labeled dataset of the target application domain, rather than on the ability to adapt a pre-trained model to these data. For instance, these approaches exploit partially labeled data (i.e. both labeled and unlabeled data) or noisy labeled data which contain a fraction of incorrect labels. However, existing methods still leave the door open to deal with complex shapes and high asymmetric fractions of corrupted labels. Moreover, despite being reduced, the labeling effort is still necessary.

Avoiding the need of hand labeled training data using Self-Supervised Learning: The need of hand labeled data can be avoided using some self-supervised learning (SSL) strategies. SSL techniques in the context of the autonomous vehicle perception propose to learn to extrapolate the available data information in order to fill the missing data information. In particular, SSL monocular camera vision approaches propose to exploit the temporal sequences of previously recorded images with the same camera. For instance, they can be trained to predict the next frame of a given video sequence based on the previously observed temporal sequences evolution [116]. However, existing state-of-the-art techniques are not able, to the best of our knowledge, to deal with partial or noisy labeled annotations in the context of discriminative tasks, such as binary image classification, detection and segmentation.

Selected direction: In our autonomous vehicle context, unlabeled training data can be easily acquired. It turns out that the above mentioned weakly supervised learning and SSL

techniques can exploit unlabeled data. Thus, based on these considerations, we propose to investigate weakly supervised learning techniques, in supplement to existing SSL techniques, with the final goal to envision novel computer vision framework perspectives for potentially moving obstacles analysis. Furthermore, in a real application context, it can be difficult to record all the varieties of urban scenario which can possibly be encountered. Thus, we propose to exploit in the meantime deep generative models for data augmentation in the context of the proposed weakly supervised learning frameworks.

Next, Sec. 1.2 presents the contributions of this thesis to address the mentioned goals.

1.2 Contributions

Scientific contributions presented through this thesis are the following:

- We propose a tutorial style overview of Self-Supervised Learning techniques for autonomous vehicles perception. More specifically, we identify relations between existing methods, advantages, drawbacks, and we highlight the current state-of-the-art limitations.

  [27] *Florent Chiaroni, Mohamed-Cherif Rahal, Nicolas Hueber, and Frederic Dufaux. Self-supervised learning for autonomous vehicles perception: A conciliation between analytical and learning methods. Accepted to IEEE Signal Processing Magazine, 2020. arXiv preprint arXiv:1910.01636.*

- We propose to regularize the classical biased Positive Unlabeled (PU) training by applying a Generative Adversarial Network [64] (GAN) on unlabeled data of Positive Unlabeled (PU) tiny image classification datasets.

  [24] *Florent Chiaroni, Mohamed-Cherif Rahal, Nicolas Hueber, and Frederic Dufaux. Learning with a generative adversarial network from a positive unlabeled dataset for image classification. In IEEE, 25th International Conference in Image Processing (ICIP), October 2018.*

  [26] *Florent Chiaroni, Mohamed-Cherif Rahal, Frederic Dufaux, and Nicolas Hueber. Classification d´images en apprenant sur des échantillons positifs et non labelisés avec un reseau antagoniste génératif. In CNIA-RJCIA, 2018.*

- We propose to generate adversarially relevant counter-examples from PU tiny image datasets without prior knowledge. To do this, we have proposed the two following contributions:

- We propose to incorporate a biased PU learning loss function inside the original GAN [64] discriminator loss function. The intuition behind it is to have the generative model solving the PU learning problem formulated in the discriminator loss function. In this way, the generator learns the distribution of the examples which are both unlabeled and not positive, namely the negative ones included in the unlabeled dataset;
- In addition, we study normalization techniques compatibility with the proposed framework. A learning model which manipulates different minibatches distributions should not use batch normalization techniques [77]. Alternative normalization techniques are discussed and experimented.

[25] *Florent Chiaroni, Mohamed-Cherif Rahal, Nicolas Hueber, and Frederic Dufaux. Generating relevant counter-examples from a positive unlabeled dataset for image classification. Submitted to Pattern Recognition, 2019 (under revision). arXiv preprint arXiv:1910.01968.*

- We propose a novel GAN-based approach to tackle the noisy labeled learning task on small and complex datasets. The main contributions of this work consist of:
  - incorporating a noisy labeled risk inside the GAN discriminator loss function;
  - applying carefully regularization techniques during the GAN adversarial training. This addresses GAN mode collapse and discriminator overfitting issues;
  - exploiting prior knowledge of the corrupted labels fractions in order to estimate the most appropriate adversarial training labels.

  [23] *Florent Chiaroni, Mohamed-Cherif Rahal, Nicolas Hueber, and Frederic Dufaux. Hallucinating a cleanly labeled augmented dataset from a noisy labeled dataset using GANs. In IEEE, 26th International Conference on Image Processing (ICIP), September 2019.* ***Spotlight article (Top 10% of accepted papers).***

- We propose a new temporally self-supervised method for unsupervised classification of moving obstacles. The main contribution is to improve state-of-the-art unsupervised image classification methods in the context of autonomous vehicle perception by adding temporal information provided by videos.

  [68] *Sid Ali Hamideche and Florent Chiaroni and Mohamed-Cherif Rahal. Self-supervised classification of dynamic obstacles using the temporal information provided by videos. To be submitted in 2020. arXiv preprint arXiv:1910.09094.*

- As an application contribution, we propose for the first time to adapt PU learning strategies for potentially moving obstacles detection and segmentation.

  Best student poster award received at the 6th Budding Science ISL Colloquium for the presentation entitled: **See farther spatially and temporally using self-supervised learning, 2017.**

  Participation for a demonstrator for vehicle detection from aerial images, at VIVA Tech forum, 16-18 may 2019.

1.3 Overview of the thesis

The outline of this thesis is as follows:

Chapter 2 presents an overview of existing self-supervised learning (SSL) frameworks for autonomous vehicle perception. First, we motivate the interest for this specific type of unsupervised approaches. Then, we successively highlight the hand-crafted and learning tools that can cooperate into SSL systems. Next, we present SSL techniques respectively for high level and low level perception analysis of ego-vehicle sensor data. Then, we finish by a section discussing the remaining limitations of presented approaches in order to open up research perspectives in this aerea and to motivate the contributions presented in the next chapters.

Then the next three chapters 3, 4, 5 propose weakly supervised solutions for dealing with partially labeled and noisy labeled datasets for baseline binary image classification tasks as follow:

- Chapters 3 and 4 present two novel two-stage GAN-based frameworks for dealing with tiny image classification using a positive unlabeled training dataset. Positive Unlabeled learning related work is established and the proposed approaches are detailed. Then, their usefulness and competitiveness in terms of prediction scores are presented through qualitative and quantitative empirical experiments. Chapter 4 presents a solution solving the first-stage overfitting limitation of the approach proposed in chapter 3.

- Chapter 5 presents a novel GAN-based solution for dealing with small-scale noisy labeled data on tiny image datasets. The related work is discussed. Then, the proposed approach is presented and followed by empirical experiments simulating small-scale noisy labeled datasets.

Next, chapter 6 presents adaptations of previously presented PU techniques for potentially moving obstacle detection and segmentation in real-world applications. More specifically, the first section highlights the ability to reduce the need of hand labeled data for vehicle

detection on aerial images. Then, the second section proposes a theoretical and empirical study showing the proof of concept of applying PU learning principles for challenging real-world semantic image segmentation tasks. Then, we present a temporally self-supervised image clustering framework to classify without training labels the detected urban moving obstacles. This strategy improves the prediction performances of a current state-of-the-art image clustering technique. Then, based on the previous contributions and corresponding considerations, we highlight the potential to develop a future unified SSL framework enabling to jointly detect, segment, and classify potentially moving obstacles without supervision.

Finally, this thesis research work draws its overall conclusions and perspectives in chapter 7.

Chapter 2

Self-supervised learning for autonomous vehicle perception

Contents

This related work chapter mainly aims at motivating more investigations on self-supervised learning (SSL) perception techniques and their applications in autonomous driving. Such approaches are of broad interest as they can improve analytical methods performances, for example to perceive farther and more accurately spatially and temporally. In the meantime, they can also reduce the need of hand-labeled training data for learning methods, while offering the possibility to update the learning models through an online process. This can help an autonomous system to deal with unexpected changing conditions in its surrounding environment. In all, this chapter first highlights the analytical and learning tools which may be interesting for improving or developping SSL techniques. Second, it presents the insights and correlations between existing autonomous driving perception SSL techniques. Then, we draw some of their remaining limitations. This opens up some research perspectives which have motivated the contributions presented through the next following chapters of this thesis.

2.1 Introduction

The interest for autonomous driving has continuously increased during the last two decades. However, to be adopted, such critical systems need to be safe. Concerning the perception of the ego-vehicle environment, the litterature has investigated two different types of methods. On the one hand, analytical methods, also referred to as hand-crafted, are generally designed from end-to-end. On the other hand, learning methods aim to design their proper representation of the observed scene.

Analytical methods have demonstrated their usefulness for several tasks, including the keypoints detection [109], [84], optical flow, depth map estimation, background subtraction, geometric shape detection, tracking filtering, and simultaneous localization and mapping (SLAM) [16]. Those methods have the advantage to be explainable from end-to-end. However, it is difficult to apply them on high dimensional data for semantic scene analysis. For example, identifying the other users present in an urban scene requires to extract complex patterns from high dimensional data captured by camera sensors.

Learning methods are nowadays the most adapted in terms of prediction performances for complex pattern recognition tasks [89] implied in autonomous vehicles scene analysis and understanding. However, the state-of-the-art results are often obtained with large and fully labeled training datasets [33]. Hand-labeling a large dataset for a given specific application has a cost. Another difficulty is to aprehend from end-to-end the learned representations. To overcome the former limitation, transfer learning and weakly supervised learning methods have appeared. Some of them can exploit partially labeled datasets [126], [26], or noisy labeled datasets [112], [22]. Concerning the latter problem, under mild theoretical assumptions on the learning model, we can interpret the predicted outputs. For instance, it is possible to automatically detect the training overfitting [75], to estimate the fraction of mislabeled examples [78], or estimate the uncertainty in the prediction outputs [56].

Another challenge is to **prevent unpredictable events**. Indeed, some scenes unseen during the training can appear frequently in the context of the autonomous vehicle. For instance, an accident on the road can change drastically the appearance and the location of potential obstacles. Thus, even if it is possible to predict when the model does not know what it observes, it may be interesting to confirm it through an analytical process and to adapt the learning model to this novel situation.

It turns out that **self-supervised learning methods (SSL)** have shown in the litterature the ability to address such issues. For instance, the SSL system in [35] won the 2005 DARPA Grand Challenge thanks to its adaptability to changing environments. SSL for

autonomous driving vehicles perception is most often based on learning from data which is automatically labeled by an upstream method, similarly to feature learning in [80]. In this chapter, we discuss the following aspects of SSL:

- abilities such as sequential environment adaptation on the application time, referred to as online learning, self-supervised evaluation, unnecessity of hand-labeled data, fostering of multimodal techniques [35], and self-improvement. For example, iterative learning reduces progressively the corrupted predictions [177];
- applications enabled by those advantages such as depth map estimation [58], [177], temporal predictions [40], moving obstacles analysis [12], long range vision [35], [67]. For example, the SSL system in [67] learns to extrapolate the appearance of obstacles and traversable areas observable by stereo-vision in a short-range, to identify the long-range obstacles and traversable areas which cannot directly be detected by stereo-vision.

(a) (b) (c)

Figure 2.1: Some self-driving cars. (a) is the self-driving car *Stanley* that won the *DARPA Grand Challenge* using a SSL system equipped with a calibrated monocular camera and a LIDAR sensor [35]. (b) is the autonomous mobile robot *LAGR*. It integrates another SSL vision approach [67] able to identify online the obstacles and road segmentation from a short-range stereovision up to a long-range monocular vision. (c) is the car equipped with the perception sensors used to generate the KITTI dataset [59].

While the cited SSL techniques are respectively designed for a specific use case application, they present some similitudes. In particular, a shared underlying idea is to: *Learn to predict, from a given spatio-temporal information (e.g. a single camera frame [35], [67], [65], [58], [129]), something (e.g. traversable area segmentation [35], [67], depth estimation [58], or moving obstacles segmentation [65], [129]) that can be automatically labeled in another way using additional spatio-temporal information (e.g. stereo-vision camera [67], [58], a temporal sequence [128], or depth sensor [35]).*

We propose to highlight those interdependencies hereafter. In this way, we aim at providing to the reader some analytical, learning and hybrid tools which are transversal

to the final application use cases. In addition, the limitations of the presented frameworks are discussed and highlighted in Table 2.1, as well as the perspectives of improvement for self-evaluation, self-improvement, and self-adaptation, in order to address future autonomous driving challenges.

Table 2.1: Summary of global trends for advantages and drawbacks of current state-of-the-art Analytical, Learning and SSL methods for autonomous vehicle perception challenges.

Methodology	no hand-labeling	dense complex pattern analysis	online self-evaluation and adaptation	knowledge extrapolation	low-cost sensor requirements
Analytical	+++	+	++	+	+
Supervised learning	+	+++	+	+	+++
Self-Supervised Learning	+++	++	+++	+++	++

The outline of this chapter is as follows. After this introduction, we present in Sec. 2.2 and 2.3 some analytical and learning perception tools relevant to SSL. We follow in Sec. 2.4 by the presentation of existing SSL techniques for some autonomous driving perception applications. Finally we will end by a discussion focusing on limitations and future challenges in Sec. 2.5.

2.2 Analytical methods

Before the recent growing interest around deep learning methods, many analytical methods (without learning) have been proposed, bringing baseline reference tools for multiple challenging perception tasks in the context of autonomous driving. Some of the most investigated tasks considered in this chapter are briefly introduced hereafter:

- **Keypoints feature detection**. Before analyzing the sensor data from a relatively high point of view, analytical techniques often require to perform spatial or temporal data matching using **feature detection** methods. More specifically, these methods consist of detecting and extracting local features in the sensor data. These hand-crafted features can be small regions of interest [70]. In order to enable the matching of sensor data, captured from the same scene with different spatial or temporal points of view, such features need to be as invariant as possible to scale, translation, and rotation transformations. The most common sensor data is an image captured by a camera. In this case, competitive feature detectors include SIFT [109], SURF [6], ORB [149]. When a depth sensor is also available, the depth information can be exploited in order to further improve feature detection. For instance, the TRISK method [84] is specifically designed for RGB-D images. More recently, LIDAR has enabled to capture of point clouds. To tackle this new form of sensor data, some feature detection techniques are derived from image ones (e.g. Harris and SIFT). Alternatively, some new approaches such as as ISS [178] are exclusively designed for point clouds. From a

practical point of view, implementations of common image feature detectors can be found in image libraries as OpenCV[1], and in point clouds libraries as PCL[2]. Feature detectors are exploited by several autonomous driving perception techniques requiring matching of sensor data, including optical flow, disparity map, visual odometry, SLAM, tracking techniques.

- **Optical flow** is a dense [49] or sparse [111] motion pattern. It can be obtained by computing points or features transformations throughout a temporal images sequence captured from a static or mobile ego-camera point of view. In the context of autonomous driving perception, optical flow can be interesting for background subtraction, motion estimation of the ego-vehicle and surrounding moving obstacles as proposed by Menze et al. [117]. It can also be used, in the absence of additional information, for relative depth map estimation [134] of the surrounding static environment.

- **Depth map estimation** aims at providing image pixels depths, namely the relative or absolute distance between the camera and the captured objects. Several techniques exist to address this task. One of the most common and effective approaches is to compute a disparity map from a stereo-camera. Combined with the extrinsic cameras parameters, such as the baseline distance separating both cameras, the disparity map can be converted into an inversely proportional absolute depth map. Another approach is to project LIDAR points on some of the camera image pixels. It also requires extrinsic spatial and temporal calibrations between both sensors. As mentioned previously, a relative depth map can also be directly deduced on the move from the optical flow obtained with a moving camera in a static scene. Under some assumptions, the absolute depth map can then be obtained, for example with additional accurate GPS and IMU sensors information concerning the absolute pose transformations of the moving camera. The depth map can also be directly obtained with some RGB-D sensors. Depth map is interesting for identifying the 3D shape of objects in the scene. More specifically, in autonomous driving, an absolute depth map is relevant for estimating the distance between the ego-vehicle and detected obstacles. However, we can note that absolute depth map estimation is constraining compared to relative depth map, as at least two jointly calibrated sensors are necessary. Consequently, this has a relative higher financial cost in production. Moreover, extrinsic calibrations can be sensitive to the ego-vehicle physical shocks. Finally such sensor fusions can only offer limited long-range depth estimation, due to fixed baselines with stereo cameras,

[1] https://opencv.org/
[2] http://pointclouds.org/

or sparse point cloud projections with dispersive LIDAR sensors. Nevertheless, relative depth map can be sufficient to detect obstacles and traversable areas. For example, considering the traversable area as a set of planes in the depth map 3D point cloud projection, some template matching techniques can be used [67].

- **Geometric shape detection** techniques such as Hough transform and RANSAC [52] initially aimed at identifying some basic geometric shapes such as lines for lane marking detection, ellipses for traffic lights detection, or planes for road segmentation. In order to deal with sophisticated template matching tasks, techniques such as the hough transform have been generalized (GHT [5]) for arbitrary shape detection. Nonetheless, these techniques require an exact model definition of the shapes to detect. Consequently, they are sensitive to noisy data and are impractical for detection of complex and varying shapes such as obstacles encountered in the context of autonomous driving. Indeed, such objects typically suffer from outdoor illumination changes, background clutter, or non-rigid transformations.

- **Motion tracking** aims at following some data points, features or objects through time. Tracking filters, such as the Extended Kalman Filter (EKF), predict the next motion using the prior motion knowledge. Conversely, objects tracking can be achieved by features or template matching between consecutive video frames. Pixel points and features tracking is interesting for dense or sparse optical flow, as well as visual odometry estimation [155]. Obstacle objects tracking is very relevant in autonomous driving for modeling or anticipating their trajectories into the ego-vehicle environment. However, on the whole, while some techniques integrate uncertainty, they remain limited when dealing with complex real motion patterns. Pedestrians and drivers behaviour prediction typically requires knowledges about the context. Moreover, mobile obstacles appearence can drastically change depending on their orientation.

- **SLAM techniques.** The complementarity between the above enumerated concepts has been demonstrated through the problem of *simultaneously localizing* the ego-vehicle *and mapping* the surrounding environment (SLAM) [16]. Features matching provides the pose transformations of the moving ego-vehicle. In turn, 3D scaled projections of depth maps combined with the successive estimated poses provide the environment mapping. Tracking filters and template matching can offer some robustness against sensor data noise and drifting localization estimation, as respectively proposed in EKF SLAM [37] and SLAM++ [151] approaches.

To summarize, analytical methods can successfully deal with several perception tasks of significant interest in the context of autonomous driving. In particular, a self-driving vehicle embedding these techniques is able to carry out physical analysis such as the 3D reconstruction modelling of the environment, and dynamic estimations concerning the ego-vehicle and the encountered surrounding mobile obstacles. These techniques have the advantage to be end-to-end explainable in terms of design. This facilitates the identification and prevention of failure modes. However, some critical limitations persist nowadays:

- A lack of landmarks and salient features combined with the presence of dynamic obstacles may entail a severe degradation of the feature detection and matching.
- Severe noisy sensor data induces the same risks.
- It is impossible to achieve dense real-time semantic scene analysis of environments including a wide range of complex shape patterns.

Learning to recognize and predict complex patterns with generalization abilities aims at overcoming such issues, as developed in the next section.

2.3 Learning methods

Learning methods have demonstrated state-of-the-art prediction performances for semantic tasks during the last two decades. Autonomous driving is a key application which can greatly benefit from these recent developments. For instance, learning methods have been investigated in this context, for identifying the observed scene context using classification, for detecting the other road users surrounding the ego-vehicle, for delineating the traversable area surface, or for dynamic obstacles tracking.

- **Classification:** It aims at predicting, for a given input sensor sample, an output class label. In order to deal with high dimensional data containing complex patterns, the first stage is generally to extract relevant features using hand-crafted filters or learned feature extractors. For image feature extraction, the state-of-the-art techniques use Convolutional Neural Network (CNN) architectures. The latter are composed of a superposition of consecutive layers of trainable convolutional filters. Then, a second stage is to apply a learning classifier on the feature maps generated as output of these filters. Some commonly used classifiers are the Support Vector Machine (SVM) and the Multi-Layer Perceptron (MLP). Both require a training which is most of the time performed in a fully supervised way on labeled data. The CNN and MLP deep learning models are trained by backpropagating the output prediction error on the

trainable weigths up to the input. Concerning the evaluation of these models, a test dataset is required, which is labeled as well. The *Accuracy* metric is commonly used for evaluating the prediction performances, while the F1-Score, an harmonic mean of the precision and recall, is relevant for information retrieval. An image classification application example in autonomous driving is for categorizing the context of the driven road [162].

- **Detection:** It generally identifies in a visual sensor data the regions of interest, which in turn can be classified. A commonly used strategy invariant to scales and rotations applies an image classifier on sliding windows over an image pyramid. Then, several advanced competitive image detection techniques as Faster R-CNN [144] which improved Fast R-CNN [60] and R-CNN [61], SSD [106] and Yolo versions [141] [142] [143] have been more recently developed, and have been adapted for road users detection [162].

- **Segmentation:** As its name suggests, this task provides a segmentation of visual sensor data. Three distinct problems can be considered:

  - *Semantic segmentation* assigns a semantic class label to each pixel. An example is road segmentation [162]. State-of-the-art methods generally present a fully convolutional network (FCN) autoencoder (AE) architecture connecting in different ways the encoding part with the decoding part [3]. A standard AE is a generative model composed of an encoder and a decoder learning models which are jointly trained to reconstruct as output the input. State-of-the-art semantic segmentation methods can also use dilated [169] or atrous convolutions [20], as well as an image context modeling strategy [176], as reviewed in [57]. In the discussed image segmentation context, these models are trained to predict as output a per-pixel classification of the input image pixels.
  - *Instance segmentation* aims at detecting and segmenting each object instance. Examples include foreground segmentation and object detection of potentially moving obstacles [71].
  - *Panoptic segmentation* [89] is a unification of the two previously mentioned segmentation tasks.

  Some models dealing with these segmentation tasks have been adapted for performing per-pixel regression tasks such as dense optical flow estimation [41] or depth map estimation [104].

- **Temporal object tracking** follows the spatial location of selected objects along a temporal data sequence. State-of-the-art learning techniques use variants of the Recurrent Neural Network (RNN) model [119]. Compared to standard filtering techniques, RNNs have the ability to learn complex and relatively long-term temporal patterns in the context of autonomous driving.

These methods can be combined in a unified framework, for instance by sharing the same encoded latent feature maps, as proposed in MultiNet [162] for joint real-time scene classification, vehicle detection and road segmentation.

While demonstrating competitive prediction performances, the above mentioned learning techniques are fully supervised. In other words, they have in common the limitation to require large-scale fully annotated training datasets. In order to reduce this issue, some other learning strategies have been investigated:

- **Weakly supervised learning:** These techniques can be trained with a partially labeled dataset [126], and eventually with a fraction of corrupted labels [112], [22]. Advantageously, these approaches drastically reduce the need of labeled data.

- **Clustering**: These approaches can be defined as an unlabeled classification stategy, such that it aims at gathering without supervision the data depending on their features similarities. A huge advantage is that no labels are required. However, if it is necessary to associate the clusters obtained with humanly understandable semantic meanings, then a final step of ponctual hand-labeling per-cluster is required. State-of-the-art methods [18] dealing with complex real images mix trainable feature extractors with standard clustering methods such as a Gaussian Mixture Model (GMM) [122].

- **Pre-training:** Some relevant generic visual feature extractors can be obtained by performing a preliminary pre-training of the CNN model on unlabeled or labeled data coming from the target application domain [67] or even from a different one [62].

We note also that in order to aprehend from end-to-end the learned representations, it is possible to identify the training overfitting [75] of deep learning models without validation test supervision. Furthermore, some learning approaches can estimate the prior of a noisy labeled training dataset [78] or the model uncertainty [56], [86].

Now that some considered analytical and learning methods have been treated separately, the next section shows the complementarity between these two different types of approaches through several Self-Supervised Learning (SSL) systems developed in the context of the autonomous driving vehicle perception.

2.4 SSL Autonomous Driving Applications

In the context of autonomous driving applications, we can organize the Self-Supervised Learning (SSL) perception techniques in two main categories:

- High-level scene understanding:
  - road segmentation in order to discriminate the traversable path from obstacles to be avoided
  - dynamic obstacles detection and segmentation
  - obstacles tracking and motion anticipation predictions
- Low-level sensor data analysis, with a particular focus on:
  - dense depth map estimation, which is a potentially relevant input data information for dealing with the previously enumerated scene understanding challenges.

2.4.1 Scene understanding

In order to navigate safely, smoothly, or fast when it is required, a self-driving car must perform a path planning adapted to the surrounding environment. The planned trajectories must pass trough traversable areas, while ensuring that surrounding static and dynamic obstacles are avoided. For this purpose, it is necessary to detect and delineate them in advance, but also to anticipate future positions of the mobile ones.

2.4.1.1 Traversable area segmentation

A traversable area can be identified by performing its segmentation over the mapped physical environment. Two different strategies have been successively applied. The former is mainly dedicated to offroad unknown terrain crossing. It entails fully self-supervised training (i.e. without hand-labeled data) systems. The latter, that appeared more recently, is dedicated to urban road analysis. The main difference is that the SSL online systems developed are initialized with a supervised pre-training on hand-labeled data. This preliminary step aims at replacing the lack of landmarks on urban asphalt roads having uniform textures, by prior knowledge.

SSL offroad systems: a road segmentation is proposed in [100] by exploiting temporal past information concerning the road appearence on monocular camera images. It considers the close observable area on the current moncular camera frame in front of the car as a traversable road. Next, it propagates optical flow on this area from the current frame up to the past captured frames. Then, it can deduce this close area appearence when it was

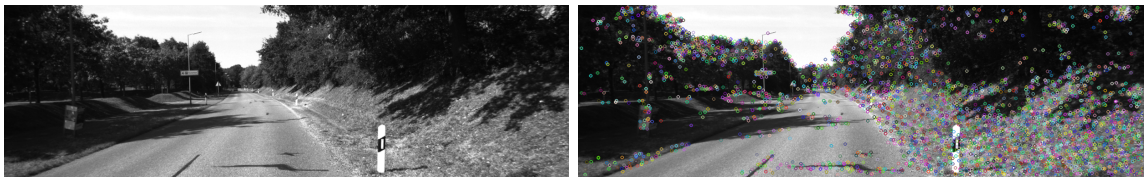
spatially farther in the past. This past appearence of the actual close traversable area is exploited for producing horizontal line templates using the SSD (sum of squared differences) matching measure. It is combined with a hough transform-based horizon detector to define the image horizontal lines of pixels on which to apply the horizontal 1-D template matching. Next, with the assumption that the actual distant traversable area has roughly the same appearence as the actual close area had in the past, the 1D templates are applied over the current frame to segment the distant traversable area. If the best template matching measure changes abruptly, then it is supposed that the ego-vehicle is going out of the road or that the road appearence has suddenly and drastically changed. The approach in [100] is relevant for providing a long-range road image segmentation using a monocular camera only. However, a major issue is the critical assumption considering the close area as always traversable. If the road aspect suddenly changes, then it is impossible with this SSL strategy to correctly segment the image pixels following an unknown distribution corresponding this novel road region.

Another SSL road segmentation approach is proposed in [35] dealing naturally with this issue. Instead of using temporal information with the assumption that the close area is always traversable, and in addition to the monocular camera, a LIDAR sensor is used for detecting the obstacles close to the ego-vehicle. Projected on the camera images, LIDAR depth points enable to automatically and sparsely labelize the close traversable area on images pixels. Then, a gaussian mixture model (GMM) is trained online to recognize the statistical appearence of these sparse analytically labeled pixels. Next, the learning model is applied on the camera pixels which cannot benefit from the sparse LIDAR points projection, in order to classify them as road pixels or not. In this way, the vehicle can anticipate the far obstacles observable in the monocular camera images, but not in the dispersive LIDAR data. This SSL system enabled the *Stanley* self-driving car, presented in Figure 2.1(a), to win the *DARPA Grand Challenge*[3] by smoothing the trajectories and increasing the vehicle speed thanks to the anticipation of distant obstacles. This highlighted the interest of combining multiple sensors in a self-driving car.

More recently, with the growing interest for deep learning methods, Hadsell et al. [67] propose to use a CNN classifier model instead of the earlier template matching or GMM learning techniques. Moreover, an additional paired camera (i.e. stereo-camera) replaces the LIDAR sensor as in [35]. As offroad terrain traversable areas are not always completely flat, a multi-ground plane segmentation is performed in [67], on the short-range point cloud projection, obtained with the stereo-vision depth map, by using a hough transform plane detector. This technique provides several automatic labels for image patches which are

[3]https://www.darpa.mil/about-us/timeline/-grand-challenge-for-autonomous-vehicles

observable in the short-range region. Then, addressing the long-range vision segmentation, the authors firstly train a classifier to predict patches labels automatically estimated within the short-range region. Next, the trained classifier predicts the same labels on the long-range observable image region patches by using a sliding window classification strategy. Concerning the prediction performances, the authors have demonstrated that the online fine tuning of the classifier and the offline pre-taining of its convolutional layers using an unsupervised autoencoder architecture can improve prediction performances. Moreover, an interesting point to note is that instead of using uncertainty or noisy labeled learning techniques, the authors created transition class labels for the boundary image surfaces separating the obstacles from the traversable area. Finally, from an initial 11-12 meters short range stereo-vision, the developed SSL system was able to extrapolate a long-range vision up to 50-100 meters. Nonetheless, in order to estimate the short-range stereo 3D reconstruction, including planar sets of points corresponding to the offroad traversable area, this approach requires the presence of salient visual features in the road regions. This may be impractical for instance on the uniform visual texture of asphalt roads commonly encountered in urban scenarios, as illustrated in Fig. 2.2.



(a) (b)

Figure 2.2: Salient features location on urban ego-vehicle environment. (a) is an arbitrary frame, extracted from the KITTI dataset [59], illustrating an urban asphalt road with the surrounding environment. (b) shows keypoints detected on the left input image using SIFT detector. Keypoints distribution is dense on the offroad region observable in the image right side, and sparse on the asphalt road observable in the image center.

Pre-trained SSL urban road systems: Some other techniques, that we propose to consider as SSL only during the online process, deal with this issue by exploiting a classifer pre-trained offline on hand-labeled data [180], [147].

The automatic labeling step previously performed with analytical methods is replaced in [180] by an SVM classifier pre-trained offline using a human annotated dataset. In this way, this approach is also compatible with uniform asphalt road surfaces. However, compared to the previously presented SSL offroad approaches, it requires hand-labeled data.

A hybrid path segmentation technique is proposed in [147]. It combines a 3D traversability cost map obtained by stereo-vision, and an SVM classifier pre-trained offline over a human

annotated dataset. Six different ground surfaces are considered to train the classifier: asphalt, big gravel, small gravel, soil, grass, bushes and stones. The strategy is as follows. SVM predictions refine online the cost map concerning the flat regions. In turn, the 3D traversability cost map obtained without supervision is exploited to update online some mis-classifications of the pre-trained classifier.

To sum up regarding these road segmentation SSL systems, we can notice that while the sensor data and the analytical and learning models are different for each approach, the online process remains essentially the same. The first stage always consists of generating automatic labels by using additional temporal [100], sensor [35], [67], or prior knowledge information [180], [147]. Then, a second stage trains or updates online a classifier, such that it can be used to provide a long-range or refine road segmentation. Overall, while the short-range visions based on depth sensors aims at ensuring the reliable detection of close obstacles, using such SSL vision techniques in static environments directly enables to anticipate the path planning evolution. Consequently, it is possible to increase the maximum speed velocity of the self-driving car [35], while preserving smooth trajectories [67].

Now that we have presented some SSL techniques dealing with limited depth sensors in static environments, we focus on dynamic obstacles, as they represent the other potential road users interacting with the ego-vehicle in the shared surrounding environment.

2.4.1.2 Dynamic obstacles analysis

We start by presenting an SSL approach [65] based on a binary per-pixel segmentation of dynamic obstacles. Then, we present its extension [12] for dynamic obstacles instance segmentation, such that the different road users can be separated.

SSL for dynamic obstacles per-pixel segmentation: a per-pixel binary segmentation of dynamic obstacles is proposed in [65], using temporal image sequences captured with a monocular camera installed on a mobile urban vehicle. The approach firstly separates sparse dynamic keypoints features from the static ones, by applying a RANSAC technique over the optical flow between consecutive frames. Then, the automatically produced per-pixel dynamic labels are transferred as input of a learning Gaussian Process (GP) model. Next, the learned model extrapolates this knowledge to label as dynamic the pixels following the same visual propoerties than the ones previously automatically identified as dynamic. The whole process is achieved during an online procedure. The system is evaluated on a hand-labeled dataset. This SSL strategy has the advantage to provide the background subtraction from a moving camera, while extrapolating a dense per-pixel segmentation of the dynamic obstacles from sparse detected keypoints only. However, this technique cannot

provide per-obstacles analysis as it merely predicts a binary mask of pixels corresponding to dynamic obstacles.

The technique in [12] extends the previous approach for SSL multi-instance segmentation by using temporal image sequences captured with a monocular camera installed on a mobile urban vehicle. The authors apply, over the mobile keypoints detected by [65], a clustering method using the tracked keypoints information such as their spatial location and motion pattern features. The multi-instance segmentation of dynamic obstacles is evaluated on a hand-labeled video sequence of the KITTI dataset [59].

Overall, the authors state that some issues shared with analytical methods persist in their approach. If the dynamic obstacles shadows are projected on the background, then the latter are considered as dynamic as well. Moreover, the segmentation of distant dynamic obstacles can be missed if the corresponding keypoints variations are considered as noise due to the difficulty to detect the corresponding slight optical flow variations. Furthermore, if a dynamic obstacle, large or close to the sensor, represents the majority of the image keypoints, then this given obstacle is likely to be treated as the static background scene.

Nonetheless, it is important to bear in mind that these approaches present state-of-the-art competitive performances for dynamic obstacles detection and segmentation without training or pre-training on annotated data. In addition, the method in [12] provides interesting tools to analyze on the move the dynamic obstacles, for example to separately track them and learn to predict their intention.

The next focus is on SSL techniques designed for object tracking and temporal predictions in urban road scene evolution, including dynamic obstacles.

2.4.1.3 Temporal tracking predictions

In order to deal with object appearence changes, a competitive SSL tracking technique [83] proposes an online adaptive strategy combining tracking, learning, and object detector real-time modules. However, in the context of autonomous driving, it may be often necessary to simultaneously track, and even anticipate the trajectories of several surrounding road users. Moreover, being able to consider the interactions between each road user requires under particular circumstances some complex motion pattern analysis.

It turns out that some SSL approaches propose to deal with this challenge by focusing the prediction effort on the entire scene in a unified way, rather the on every obstacles independently. The *deep tracking* system [128][4] learns to predict the future state of a 2D LIDAR occupancy grid. This is achieved by training an RNN on the latent space of a CNN

[4]Such an approach could be categorized as unsupervised. However, exploiting during the training an additional future temporal information, not available during the prediction step, is a type of self-supervision.

autoencoder (AE) which is applied on the input occupancy grid considered as an image. Each cell of the grid is represented by a pixel, which can be color-coded as occluded, void, or as an obstacle surface. Consequently, the model can be trained from end-to-end by learning to predict the next occupancy grid states using the past and current grid states. Solely the prediction output error of non occluded cells is backpropagated during the training. By definition, this system can perform a self-evaluation by computing a per-pixel photometric error between the predicted occupancy grid and the real future observed occupancy grid at the same temporal instant. This technique has the advantage of being compatible with complex motion patterns compared to Bayesian and Kalman tracking techniques. In addition, the training process enables to predict the obstacles trajectories even during occlusions. The major interest of *deep tracking* is that, as the model learns to predict a complete scene, it naturally considers interactions between each dynamic obstacle present in the scene. In [40], the *deep tracking* model is extended for a real mobile LIDAR sensor by adding a spatial transformer module in order to take into consideration the displacements of the ego-vehicle with respect to its environment during objects tracking.

In turn, these tracking approaches provide the tools to collect motion pattern information of surronding dynamic obstacles such that this information may help to classify obstacles depending on their dynamic properties [51].

2.4.2 Low-level sensor data analysis

We address now the sensor data analysis for low-level information estimation in the context of autonomous driving. Compared to the previous methods, the attention has mainly focused recently on SSL depth map estimation from monocular or stereo cameras.

2.4.2.1 SSL Depth map estimation

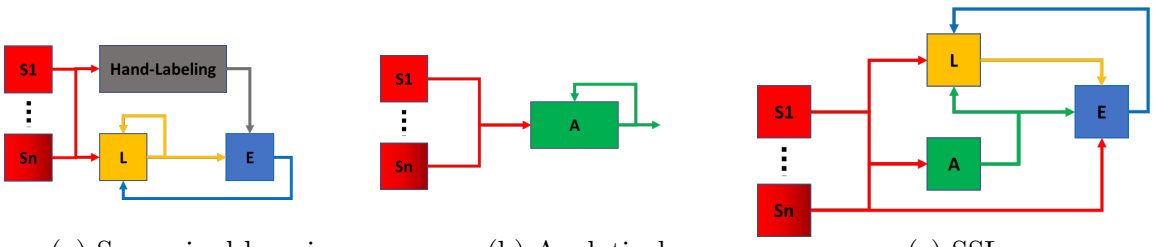
The self-supervised depth map estimation approach presented in [58] predicts a depth map from a monocular camera without relying on annotated depth maps. The pose transformation between both left and right cameras is known. The SSL strategy is as follows. First, the left camera frame is provided as input to a CNN model trained from scratch to predict, the corresponding depth map. Second, an inverse warping is performed by combining the predicted left depth map with the right camera frame in order to output a synthesized frame similar to the input left frame. In this way, an SSL photometric reconstruction error can be computed in output of the decoder part. Next, this per-pixel error is directly used to train the encoder weights using an SGD optimization technique. While not requiring pre-training, nor annotated ground-truth depths, this approach presents prediction preformances comparable with the state-of-the-art fully supervised monocular techniques. However, the

ground truth pose transformation, related to the inter-view displacement between both cameras, is required.

Following a similar idea, another technique is proposed in [177]. It is trained to reconstruct, from a given frame, the second frame taken from a different point of view. It generates a depth map using a stereo camera during the training step, but also during the prediction step. This makes the approach more robust, such that it becomes competitive with standard stereo matching techniques. Moreover, the constraint of preserving two cameras and the pose transformation ground truth for predictions, enables in counterpart to perform online learning. This may be interesting for dealing with novel ego-vehicle environments unseen during the training.

In order to overcome the necessity of the pose transformation ground-truth, Zhou et al. [181] propose to predict, from a temporal sequence of frames, the depth map with a learning model, and the successive camera pose transformations with another learning model. Both models are trained together from end-to-end for making the novel view synthesis of the next frame. However, such a pose transformation estimation implies that the predicted depth map is defined up to a scale factor.

A more modular technique [62] exploits either temporal monocular sequences of frames as in [181], the paired frames of a stereo camera as in [177], or to jointly exploit both temporal and stereo information. This framework also deals with the false depth estimation of moving obstacles by ignoring, during training, the pixels not varying between two consecutive temporal frames. It also deals with occluded pixels when the captured point of view changes by using a minimum reprojection loss.



(a) Supervised learning (b) Analytical (c) SSL

Figure 2.3: Function diagrams showing the common connections depending on the strategy. Functional blocks represent single monocular camera frame S_1, additional sensor data (e.g. temporal frame sequence, stereo-camera, or lidar data) S_n, a Learning model L, an Analytical method A, and Evaluation method E.

To summarize, low-level analysis techniques for depth map estimation have demonstrated that SSL strategies without using ground truth labels can bring state-of-the-art solutions competitive with fully supervised techniques.

SSL Methodologies	$S_1 \rightarrow L$	$S_n \rightarrow L$	$S_1 \rightarrow A$	$S_n \rightarrow A$	$S_n \rightarrow E$	$A \rightarrow L$	$L \rightarrow E$	$A \rightarrow E$	$E \rightarrow L$	datasets	performances
(Off)road segmentation [100], [35], [67], [180], [147]	√		√	√		√	√	√	√	-	-
Dynamic obstacles analysis [65], [12]	√		√	√		√				KITTI [59] Sidney [65]	-
Temporal tracking predictions [128], [40]	√	√			√		√		√	Oxford Robotcar dataset [113]	-
Depth map estimation [58], [181], [177][1], [62][1]	√	√[1]			√		√		√	KITTI Make3D [154]	[54]*>[62]>[58]> [181]>[45]*

Table 2.2: Functional block connections of presented SSL methodologies depending on the application. Experimental datasets exploited and relative prediction performances are reported whenever available. *refers to supervised methods.

Overall, the SSL techniques presented in this section support the following conclusion. By exploiting the complementarity between analytical and learning methods, it is possible to address several low-level and challenging autonomous driving perception tasks, without necessarily requiring an annotated dataset. Presented methodologies are summarized in Fig. 2.3 along with Table 2.2.

The next section presents self-supervised learning limitations and future challenges for autonomous driving perception applications.

2.5 Limitations and future challenges

In the context of autonomous driving, some limitations remain in the presented SSL perception systems and open future research perspectives:

- *Catastrophic forgetting:* During the online learning procedure, the trainable weights of the model may require unnecessary repetitive updates for detecting a given pattern throughout the environment exploration. In fact, when a learning model is continuously specialized for dealing with the latest data, the likelihood increases that the model simultaneously forget the potentially relevant formerly learned patterns. It turns out that it is possible to deal with this *catastrophic forgetting* issue when using neural networks [90]. For future research directions, it may be interesting to combine such incremental learning techniques with the presented SSL frameworks.

- Concerning the scene depth map estimation solely based on temporal analysis:

  - the presence of dynamic obstacles in the scene during the learning stage can result in poor estimates of the observed scene. As discussed in [65], further research on SSL for potentially dynamic obstacles delineations on the sensor data may help to deal with this issue.

  - the current state-of-the-art techniques cannot estimate the real depth map without requiring a supervised scaling factor. The latter is generally obtained by estimating the real metric values of the pose transformation between two consecutive camera viewpoints. As proposed in the supervised detector *Deep MANTA* [19], it may be interesting to recover automatically this scale factor by using some template matching techniques on the observable objects of the scene.

- Concerning the online self-evaluation, some of the presented systems require a baseline reference obtained analytically [67]. However, if we consider that the analytical processes, considered as ground-truth labeling techniques, are likely to generate some noisy labels, it may be interesting to investigate some future reasearch on how to evaluate this prior noise from the learning model point of view [78], and how to deal with it as proposed in chapter 5 for image classification using noisy labels [22].

- Concerning potentially moving obstacles analysis, the presented SSL state-of-the-art strategies [65] and [12] propose to exclusively segment and detect the moving instances of this class. Hence, static potentially moving obstacles, like a car waiting in front of a street intersection, a pedestrian waiting to cross the street, or any of them standing on the road in an accident backdrop, cannot be detected. What is more, to the best of our knowledge, there is no current existing SSL framework enabling to automatically allocate these moving obstacles into semantically interpretable different sub-categories like pedestrian, motocyclist, car. Yet, in a real autonomous driving application context, such static potentially moving obstacles can prospectively interact in the near future with our ego-vehicle, and consequently influence the actions to be undertaken into the given situation. Therefore, they necessarily deserve the same rights than the currently moving ones to be considered by the ego-vehicle.

Next chapters propose weakly supervised learning solutions envisioned to be complementary with existing presented SSL frameworks to address the last point: Detect, segment, and classify static potentially moving obstacles from a monocular camera by reducing as far as possible the need of hand-labeled training data.

Chapter 3

Positive unlabeled learning using unlabeled data generation

Contents

In this chapter, we propose a novel approach for image classification task from a positive unlabeled dataset. It is based on generative adversarial networks (GANs) abilities. These allow us to generate fake images whose distribution is close to the distribution of the negative samples included in the unlabeled dataset available, while remaining different from the distribution of positive samples that are not labeled. Then, we train a CNN classifier with the positive samples and the fake generated samples, as with a classical Positive Negative dataset. Although very different, this method is empirically competitive with state-of-the-art PU learning on complex RGB images while keeping a steady behaviour.

This chapter is organized as follows. We start by introducing in Sec. 3.1 the motivations of this work before presenting the proposed method in Sec. 3.2. Then, its corresponding experimentations and results are presented in Sec. 3.3. At the end, a conclusion is drawn on our approach.

3.1 Motivation

Deep learning methods using convolutional kernel filters have demonstrated good prediction performance in the field of computer vision and especially for the task of image classification despite the high dimensionality of these data. To achieve such performance, large fully labeled datasets are generally required. To reduce this need, some semi-supervised learning techniques exist [137]. However, if an image object, not belonging to any labeled class of the training dataset, must be treated, then it is difficult to predict the behaviour of the learning model.

To mitigate this problem, an idea is to mainly focus on data of interest. This is the case in One-Class Classification methods (OCC) [87], [131] where samples of the class of interest only, namely the positive class, are used during training. However, in the autonomous vehicle context, it is often easy to acquire unlabeled samples that may contain relevant information especially about the counter-examples (i.e. examples of the negative class) of the class of interest (i.e. examples of the positive class). In this way, we address a Positive Unlabeled (PU) learning problem.

Another interest for PU learning is to combine it with Self-Supervised Learning (SSL). SSL purpose is to learn to extrapolate automatically labeled data using another method. In such a context, PU techniques may bring adaptability for applying SSL in every problems where one is only able to automatically label examples of the class of interest, but not what is not included in this positive class (i.e. counter-examples). Using PU learning for potentially moving obstacles analysis through an SSL framework is a potential application explored in Chapter 6. Beyond the scope of this thesis, other PU learning application examples are presented in [153].

3.1.1 Related Work for Positive Unlabeled learning

According to [126], PU learning techniques become competitive when the number of unlabeled examples in the training set considerably increases. This is an advantage when it is easy to get unlabeled data. Moreover, some PU approaches have demonstrated the ability to deal with very small-scale unlabeled samples proportions of the class to detect, as proposed in [175], for anomaly detection by using multi-features on normal and unlabeled data. It turns out that PU learning methods have been recently applied to image [43], [91], [74], [127]. They can be classified in two subcategories from a functionnal point of view, and in two others from a practical point of view.

From a functional point of view, it is possible to perform the binary classification by directly training a classifier with the PU dataset without any pre-processing, as suggested

in uPU [43] and nnPU [91]. The second kind of approaches consists of first applying an upstream process on the initial PU dataset, to exploit during the second stage a subset which is considered by the classifier as a standard Positive Negative (PN) dataset (i.e. including only labeled positive and negative examples). For instance, Rank Pruning method (RP) [127] consists of consecutively carrying out several training samples of the classifier, by removing the least relevant samples after each training stage before performing the final training where it is considered that only correctly labeled images are preserved. However, this technique has a high computational cost due to its multiple consecutive trainings.

From a practical point of view, uPU and nnPU methods require to know the fraction π of positive samples present in the unlabeled set. This can be limiting for real applications. In contrast, to our knowledge and according to [130], RP method is the best state-of-the-art method when we do not have prior knowledge on this samples fraction π. For these reasons, experimental results compare our approach with RP.

3.1.2 Generative Adversarial Networks

Generative adversarial networks (GANs) have drawn our attention because of their ability to generate fake samples x_F that have a distribution p_F that tends towards the distribution p_{data} of the real samples x_R used during its training. The original GAN [64] contains a generative model G and a discriminative model D. Both models have a multilayer perceptron structure. A noise vector z with a distribution p_z, composed of continuous random variables, is placed at the input of G. D is trained to distinguish real samples from fake samples generated by G, while the latter is trained to produce fake samples that seem as real as possible for D. This adversarial training consists of using a minimax function value $V(G, D)$:

$$\begin{aligned} \min_G \max_D V(D,G) = \mathbb{E}_{x_R \sim p_{data}}[logD(x_R)] \\ + \mathbb{E}_{z \sim p_z}[log(1 - D(G(z)))]. \end{aligned} \tag{3.1}$$

When D can no longer distinguish real samples from fake samples, we have the following property for its scalar predicted output y_D:

$$p_F(x_F) \xrightarrow[y_D \to 0.5]{} p_{data}(x_R). \tag{3.2}$$

Other variants of GAN have emerged such as the DCGAN [136], which adapts its structure to image processing by incorporating convolutional layers. The Wasserstein GAN (WGAN) [1], on one hand integrates the $Earth - Mover$ distance (EM) into its cost function. On the other hand, it limits the weights values of its model over a pre-defined interval, in order to rectify the instability, the mode collapse problem of these early versions of GANs.

Because of their ability to learn relevant semantic representations and their already demonstrated interest for semi-supervised learning [152], their advantages have been exploited for a PU learning application.

The GenPU [74] approach also appeared during the same period than the proposed PGAN approach to answer the same PU problematic by the use of a GAN learning model. GenPU trains simultaneously five neural networks using prior knowledge during the first-stage to generate both positive and negative samples.

Here, our proposed approach that we called Positive-GAN (hereafter *PGAN*) has been tested on four different datasets and whose results are competitive with the state-of-the-art and very promising in terms of prediction performance and stability for complex images analysis. It outperforms RP on CIFAR-10 and enables to outperform as well a fully supervised training on STL-10, when using a huge amount of unlabeled data.

In the context of image classification, we have drawn the comparative Table 3.1 highlighting the proposed contribution advantages comparing to previous state-of-the-art techniques mentioned.

Table 3.1: Table summarizing contributions of the Positive-GAN compared to the state-of-the-art PU learning for image classification.

Methods	PGAN	RP	GenPU	nnPU	uPU
No use of prior knowledge	✓	✓	×	×	×
Reproducibility	✓	✓	×	✓	✓
Stability	✓	×			
Appropriate for complex image analysis	✓	×			

3.2 Proposed approach

In this section, we describe our PU learning framework as generically as possible and focus the description on the training process. The Positive-GAN learning method (PGAN) consists of substituting the absence of labeled negative samples x_N with fake samples x_F generated by our GAN, whose distribution is as close as possible to that of x_N, while being different from that of positive samples x_P. Fig. 3.1 illustrates the structure of the system.

During the Step 1, the GAN is trained with the unlabeled samples x_U following a distribution p_U from the PU training dataset that contains an unknown fraction $\pi \in (0, 1)$ of positive samples x_P following a distribution p_P and a fraction $1 - \pi$ of negative samples x_N following a distribution p_N, such that:

$$p_U = \pi \cdot p_P + (1 - \pi) \cdot p_N. \tag{3.3}$$

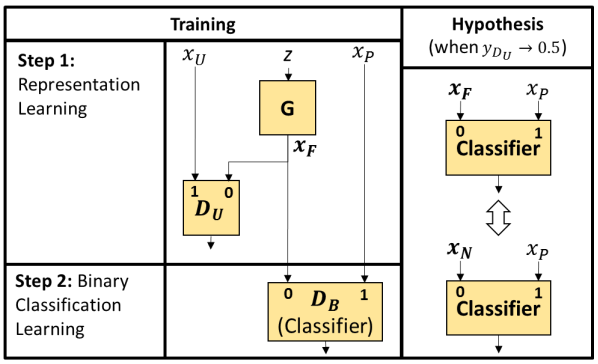


Figure 3.1: Proposed two-stage PU system using unlabeled GAN generated samples: Positive-GAN (PGAN) learning model.

The Positive Unlabeled framework includes three convolutional neural network models with different roles:

- The discriminative model D_U is trained to distinguish real unlabeled samples x_U from fake generated unlabeled samples x_F following a distribution p_F, with $y_{D_U} \in (0, 1)$ its scalar output prediction.
- The generative model G takes in input a noise vector z of continuous random variables with a uniform distribution in this case and outputs, in the same format as x_U, the fake image samples $x_F = G(z)$. G is trained in an adversarial way with D_U in order to generate fake samples such that their distribution p_F converges towards p_U.
- In Step 2, once the GAN training is considered as completed, the convolutional classifier D_B, designed for binary image classification task, is trained to distinguish the real positive samples x_P from fake samples x_F.

The next explanations aim at developping the intuition behind the proposed system. First, we recall that the untagged dataset is composed of a fraction π of positive samples x_P and a fraction $1 - \pi$ of negative samples x_N. If the GAN is correctly trained on the unlabeled samples x_U following the distribution p_U detailed in Eq. (3.3), then the convergence proposed in [64] as follows

$$p_F \xrightarrow[y_{D_U} \to \frac{1}{2}]{} p_U \tag{3.4}$$

can be developed in our context as

$$p_F \xrightarrow[y_{D_U} \to \frac{1}{2}]{} \pi p_P + (1 - \pi) p_N. \tag{3.5}$$

As a consequence, we can make the hypothesis that

$$p_F = \pi p_{FP} + (1 - \pi) p_{FN}, \tag{3.6}$$

with respectively p_{FP} and p_{FN} the distributions of generated positive samples x_{FP} and negative samples x_{FN}, such that

$$\begin{cases} p_{FP} \xrightarrow[y_{D_U} \to \frac{1}{2}]{} p_P \\ p_{FN} \xrightarrow[y_{D_U} \to \frac{1}{2}]{} p_N. \end{cases} \tag{3.7}$$

On another note, it is also known that a GAN is not perfect in its operation when it is applied to high dimensional data, therefore

$$\begin{cases} p_{FP} \neq p_P \\ p_{FN} \neq p_N. \end{cases} \tag{3.8}$$

To go even further, we also suppose that if the processed images have a high complexity of information, then the intersection between distributions p_U and p_F can remain zero in practice, such that

$$supp(p_U) \cap supp(p_F) = \emptyset, \tag{3.9}$$

with *supp* the support function. This phenomenon is intuitively illustrated in Fig. 3.2 and experimentally illustrated in Fig. 3.3.

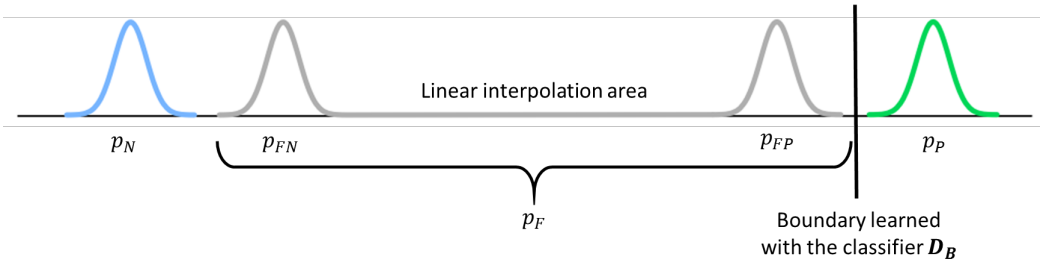


Figure 3.2: One dimensional representation of the proposed PGAN method. From the classifier point of view during the second stage, p_P is separable from the generated distribution p_F composed of p_{FP} and p_{FN}. In the meantime, p_N is semantically closer to generated p_{FN} samples than to p_P real samples. Consequently, learning to separate p_P from p_F can be relevant for learning to separate p_P from p_N.

We can observe that the distributions of the discriminator predictions for real and generated samples are separated after the interruption of the adversarial training with G. This is due to the fact that the interruption of G training enables the discriminator D_U to *go ahead of* G learning, and to determine more accurately the boundary between the real and generated distribution as the latter is frozen, such that it no longer evolves. The discriminator becomes then like a classifier which is trained on fixed distributions. This separation of p_U from p_F is possible as well during the second stage of the proposed framework if the classifier

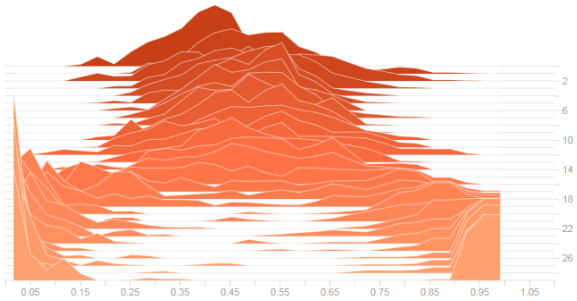


Figure 3.3: Histograms of the discriminator predictions depending on the epoch iteration, over a minibatch containing a first half of unlabeled samples x_U and a second half of generated samples x_F, during the interruption of the adversarial training. The generator G training is stopped at the fifteenth epoch while the D_U discriminator training continues during the next epochs. This experiment is realized on the dataset MNIST using the DCGAN architecture. Orange surfaces respectively represent a histogram of D_U predictions at a given epoch iteration. Horizontal lines represent epoch iterations. The bottom horztontal axis represents D_U output predictions y_{D_U} between 0 and 1.

architecture of D_B provides a capacity to encode complexity similar or greater than the D_U discriminator architecture enables during the first-stage adversarial training.

Thus, during the second stage starting when $y_{D_U} \to \frac{1}{2}$, it is possible to discriminate real positive samples x_P from all generated unlabeled samples x_F using an arbitrary cost function l in the training loss function L_{D_B} of the classifier D_B, defined as follows

$$\begin{aligned} L_{D_B} =& \mathbb{E}_{x_P, x_F \sim p_P, p_F}[l(x_P, x_F)] \\ =& \pi \cdot \mathbb{E}_{x_P, x_{FP} \sim p_P, p_{FP}}[l(x_P, x_{FP})] + (1-\pi) \cdot \mathbb{E}_{x_P, x_{FN} \sim p_P, p_{FN}}[l(x_P, x_{FN})]. \end{aligned} \tag{3.10}$$

It turns out that the training cost function term $\mathbb{E}_{x_P, x_{FP} \sim p_P, p_{FP}} l(x_P, x_{FP})$ does not influence the final predictions of the classifier D_B on the test dataset as the latter only treats real test samples. In other words, this enables to avoid the positive bias training issue previously discussed in the PU learning state-of-the-art literature. Moreover, when $p_{FN} \xrightarrow[y_{D_U} \to \frac{1}{2}]{} p_N$, and D_B has been correctly trained, p_{FN} can be considered as p_N, such that the difference between the distributions p_P and p_N can be approximated using the term $\mathbb{E}_{x_P, x_{FN} \sim p_P, p_{FN}}[l(x_P, x_{FN})]$ which is introduced in Eq. (3.10). We are thus able to calculate the distance that interests us. By transposing this reasoning in the second stage of the proposed PU framework, this amounts to asserting the following equality at the output loss function L_{D_B} of the classifier D_B when $y_{D_U} \to \frac{1}{2}$:

$$\begin{aligned} L_{D_B} &= \pi \cdot \mathbb{E}_{x_P, x_{FP} \sim p_P, p_{FP}}[l(x_P, x_{FP})] + (1-\pi) \cdot \mathbb{E}_{x_P, x_{FN} \sim p_P, p_{FN}}[l(x_P, x_{FN})] \\ &= \pi \cdot \mathbb{E}_{x_P, x_{FP} \sim p_P, p_{FP}}[l(x_P, x_{FP})] + (1-\pi) \cdot \mathbb{E}_{x_P, x_N \sim p_P, p_N}[l(x_P, x_N)]. \end{aligned} \tag{3.11}$$

With l the binary cross-entropy loss function, we obtain

$$L_{D_B} = \mathbb{E}_{x_P \sim p_P}[logD_B(x_P)] + \mathbb{E}_{z \sim p_z(z)}[log(1 - D_B(G(z)))] \tag{3.12}$$

that we consider as

$$\begin{aligned} L_{D_B} =& \mathbb{E}_{x_P \sim p_P}[logD_B(x_P)] + \pi \cdot \mathbb{E}_{x_{FP} \sim p_{FP}}[log(1 - D_B(x_{FP}))] \\ &+ (1 - \pi) \cdot \mathbb{E}_{x_N \sim p_N}[log(1 - D_B(x_N))]. \end{aligned} \tag{3.13}$$

Thus, from the assumptions made above, we can assume that the PGAN method is getting closer to a Positive Negative training while moving away from a Positive Unlabeled training despite the fact that the training dataset contains only labeled positive samples and unlabeled samples. In addition, the estimation of $l(x_P, x_{FP})$ has the potential to foster the learning of p_P boundaries. However, two risks can occur with this method:

- If the untagged samples x_U contain mostly positive samples x_P, then it is possible that G no longer generates enough fake samples similar to the samples x_N.
- If G generates false samples with a distribution equal to that of the real samples, unlike the inequalities presented in Eq. (3.8), then PGAN would become equivalent in terms of performance to a classical Positive Unlabeled training.

But, when the dimensionality of images to be processed is large, the second risk disappears as empirically shown in Sec. 3.3. Moreover, it can be reduced using a D_B classifier structure for which prediction performances are better than those of the D_U discriminator. What is more, we recall that the discriminator is adversarially trained to separate unlabeled samples including positive and negative samples from generated samples. For its part, the classifier D_B is trained to distinguish labeled samples representing only the positive class from generated samples. Given that the unlabeled set contains samples from the positive and negative classes, the classification task that the classifier D_B has to perform can be considered as a subset of the task previously performed by the discriminator D_U. This suggests that the task of the classifier D_B requires a smaller capacity for encoding the information complexity than the task of the discriminator D_U. Consequently, when the discriminator is no longer suitable for separating real from generated unlabeled samples (i.e. when its classification error has sufficiently converged towards 0.5), the classifier D_B is still able to perform its task correctly.

To sum up, the ideal operation is obtained when p_F has sufficiently converged towards p_U, while remaining different. This is possible in practice as empirically demonstrated in Sec. 3.3. The PGAN thus consists of simultaneously exploiting strengths and weaknesses of the generative model. Nonetheless, finding a solution dealing simultaneously with the

Algorithm 1 Training of the Positive-GAN

Initialize $epochs_{GAN} = 10$ (10 for MNIST, 20 for Fashion-MNIST and 100 for CIFAR-10).
Initialize $epochs_{Classifier} = 20$.
GAN training (Step 1):
for $epoch = 0$ **to** $epochs_{GAN}$ **do**
  for $minibatch = 0$ **to** $int(\frac{nbrImagesDatasetU}{m})$ **do**
    Minibatch sample of m unlabeled images $\{x_U^{(1)}, ..., x_U^{(m)}\}$ from the distribution p_U.
    Minibatch sample of m noise vectors $\{z^{(1)}, ..., z^{(m)}\}$ from the distribution $p_z(z)$.
    Update of the trainable weights θ_{D_U} of D_U by performing a stochastic gradient descent:

$$\nabla_{\theta_{D_U}} \frac{1}{m} \sum_{i=1}^{m} log[D_U(x_U^{(i)})] + log[1 - D_U(G(z^{(i)}))]$$

    Update of the trainable weights θ_G of G by performing a stochastic gradient descent:

$$\nabla_{\theta_G} \frac{1}{m} \sum_{i=1}^{m} log[1 - D_U(G(z^{(i)}))]$$

  end for
end for
Training of the classifier D_B (Step 2):
for $epoch = 0$ **to** $epochs_{Classifier}$ **do**
  for $minibatch = 0$ **to** $int(\frac{nbrImagesSetP}{m_P})$ **do**
    Minibatch sample of m_P positive images $\{x_P^{(1)}, ..., x_P^{(m_P)}\}$ from the distribution p_P.
    Minibatch sample of $m_z = m_P$ noise vectors $\{z^{(1)}, ..., z^{(m_z)}\}$ from the distribution $p_z(z)$.
    Update of the trainable weights θ_{D_B} of D_B by performing a stochastic gradient descent computed using the cost function l:

$$\nabla_{\theta_{D_B}} \frac{1}{m_P + m_z} \Big[\sum_{i=1}^{m_P} l(D_B(x_P^{(i)}), y = [0; 1]) + \sum_{i=1}^{m_Z} l(D_B(G(z^{(i)})), y = [1; 0]) \Big]$$

  end for
end for
The stochastic gradient descent (SGD) method used is Adam. The cost function l can be the binary-cross entropy or the mean-squared error (MSE).

above mentioned risks is an interesting question for improving the proposed approach trustworthiness. Thus, this PhD work leads to propose in chapter 4 a novel GAN-based PU approach addressing this second risk.

From a practical point of view, the implementation of the proposed framework is detailed with Algorithm 1.

3.3 Experiments

3.3.1 Settings

Experiments have been realized on datasets MNIST [96], Fashion-MNIST [166], CIFAR-10 [92], and STL-10 [31]. We have compared our approach to RP [127], which is to the best

of our knowledge the best asymetric noisy labeled learning and PU learning method not using ground truth prior knowledge concerning the fraction π. Author's implementation is available [1]. We also report the performance of the classifier trained on the entire initial fully labeled Positive Negative training dataset, naturally referred to as PN. We also compare our method to a training referred to as PU, which is equivalent to PN but with a substitution of the labeled negative samples by unlabeled samples.

For comparative experiments on MNIST, Fashion-MNIST and CIFAR-10, PN, PU, RP and proposed PGAN methods are tested with exactly the same convolutional neural network (CNN) classifier in order to stay impartial. In order to remain generic, the classifier has the same structure as the model[2] proposed by tensorflow that is devoted to classify tiny images. It contains two successive convolutional layers, two corresponding maxpooling steps, and then two consecutive fully connected layers. The activation function after each layer is ReLU except the last one where softmax is applied. We only changed its last top layer from 10 to 2 neurons to adapt it for binary classification. The classifier is trained on 20 epochs iterations. For the CIFAR-10 dataset with images of size $32 \times 32 \times 3$, the input and output tensors of the two convolutional layers are adapted and the depth of the first convolutional kernel filters is 3 to correspond to the 3 channels of the RGB images. But the number of kernel filters and their height and width remain unchanged.

Thanks to the WGAN [1] abilities, we combine its training method with the DCGAN architecture for these experiments. Note that with the distance *Earth Mover*, p_F converges towards p_U when y_{D_U} converges towards 0. In these experiments, the input noise vector z is a continuous random variable with uniform distribution. The training duration for the generative model depends on the dataset complexity: 10 epochs for MNIST, 20 for Fashion-MNIST, and 100 for CIFAR-10. For the latter, we do the same modifications in the structures of D_U and G as explained before for the classifier.

Regarding the PU training dataset, ρ corresponds to the fraction of positive samples from the total of positive samples in the initial training dataset which contains n_P positive samples. Collected $\rho \cdot n_P$ samples are then introduced into the U_{train} unlabeled dataset, which initially contains only negative samples whose total number is n_N. π is the fraction of positive samples in the unlabeled training dataset U_{train}. To achieve this, we remove some negative samples from U_{train}, such that U_{train} contains both negative and positive samples according to the ρ and π parameters. We establish that with $\pi \in [\frac{1}{\frac{n_N}{\rho \cdot n_P}+1}, 1)$ and $\rho \in (0, 1)$, we can obtain consecutively, with P_{train} the set of positive labeled samples, the

[1] *https://github.com/cgnorthcutt/rankpruning*

[2] *https://github.com/tensorflow/tensorflow/blob/master/tensorflow/examples/tutorials/mnist/mnist_softmax.py*

two following training sets:

$$\begin{aligned} P_{train} &= \{(1-\rho)n_P \ P \ ; \ 0 \ N\}, \\ U_{train} &= \{\rho \cdot n_P \ P \ ; \ \frac{1-\pi}{\pi}\rho \cdot n_P \ N\}, \end{aligned} \tag{3.14}$$

where notations a P and b N respectively refer to a positive and b negative items (i.e. images or examples).

To find equations making U_{train} depending on parameters ρ and π, and the interval of possible values for the latter, we use n_U which represents the total number of unlabeled samples contained in the set U_{train} such that

$$\begin{aligned} U_{train} &= \{\pi \cdot n_U \ P \ ; \ (1-\pi)n_U \ N\} \\ &= \{\rho \cdot n_P \ P \ ; \ (1-\pi)\frac{\rho \cdot n_P}{\pi} \ N\}, \end{aligned} \tag{3.15}$$

as one imposes $\rho \cdot n_P = \pi \cdot n_U$. However, in order to satisfy this constraint, the term $(1-\pi)\frac{\rho \cdot n_P}{\pi}$ must be less than or equal to n_N. This implies that $\pi \in [\frac{1}{\frac{n_N}{\rho \cdot n_P}+1}, 1)$. The results presented below are all performed with $\rho = 0.5$ and for several values of π.

Concerning our comparative experiments with RP for the *One-versus-Rest* task, and as previously proposed in RP article [127], we use the F1-Score metric for its relevance in such information retrieval and binary classification tasks. As highlighted in [103], the F1-score measures the positive examples retrieval. More specifically, the F1-Score is the harmonic mean of the precision and recall such that we have:

$$F1 - Score = 2 \cdot \frac{precision \cdot recall}{precision + recall}, \tag{3.16}$$

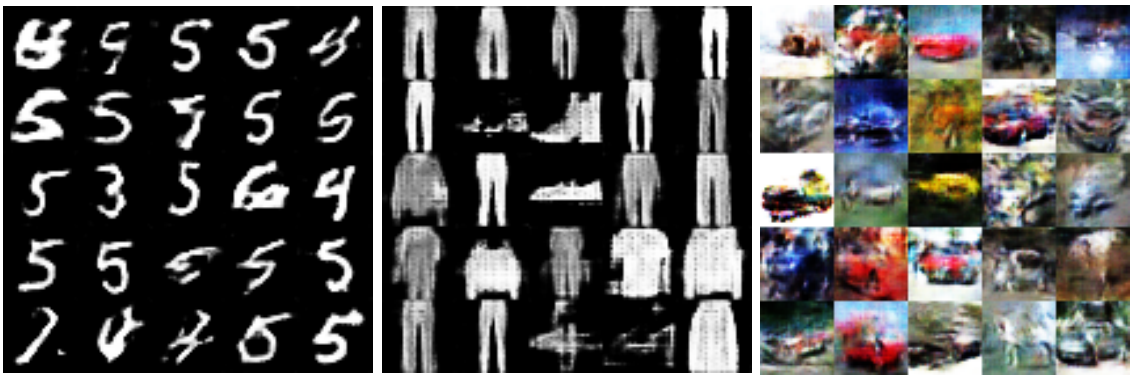
with $precision = \frac{TP}{TP+FP}$ and $recall = \frac{TP}{TP+FN}$, with TP, FP and FN the true positive rate, the false positive rate and the false negative rate, respectively. Concerning the other experiments, we use the Accuracy metric for prediction performance evaluation of the proposed approach. In contrast to the F1-Score metric, the Accuracy metric exploits in addition the true negative rate TN in its formula defined as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{3.17}$$

To compute the F1-Score and the Accuracy, the ArgMax function is applied to the two output neurons of the classifier. Thus, if the index of the first neuron is returned by ArgMax, then the current sample is considered as negative. Otherwise, the sample is considered as positive. In addition, the *One-versus-Rest* test datasets have nine times more negative examples than positive ones. Thus, once every test prediction is performed, we upsample the proportion of test ground truth positive examples up to the size of test ground truth counter-examples set, in order to measure balanced scores.

3.3.2 Results

In Fig. 3.4, we present some of the fake images generated from MNIST, Fashion-MNIST and CIFAR-10 respectively. We can notice that images generated by G seem visually real, which indicates from a qualitative point of view the effectiveness of the generative model. In order to get such a rendering, more complex and larger images imply that the generative model requires an increased number of training epochs.



(a) (b) (c)

Figure 3.4: Images generated by G trained with $\rho = 0.5$ and $\pi = 0.5$ after (a) 10 epoch iterations on MNIST, (b) 20 epoch iterations on Fashion-MNIST, and (c) 100 epoch iterations on CIFAR-10. Respective positive classes are "5", "*trouser*" and "*automobile*".

Table 3.2 and Fig. 3.5 show F1-Score comparative results for every tested method previously mentioned in Sec. 3.3.1, depending on π. It can be observed that the PN method is a good reference for MNIST and Fashion-MNIST datasets. We find that the efficiency of the PGAN learning method is equivalent to that of the RP method up to $\pi = 0.5$ on MNIST and $\pi = 0.3$ on Fashion-MNIST. Its efficiency then declines slightly faster than for RP, while keeping a correct F1-Score. On CIFAR-10 from end to end, the average F1-Score is better for PGAN than for RP. Note that the PGAN method also presents better results than the reference PN up to $\pi = 0.8$, which is quite surprising. This is probably due to the fact that the generated samples represent a larger field of negative sample distributions than the real negative samples present in the initial Positive Negative dataset. Moreover, the PGAN F1-Score is consistently higher than the PU method on the three datasets, even with only 10% of positive samples among unlabeled samples (i.e. with $\pi = 0.1$).

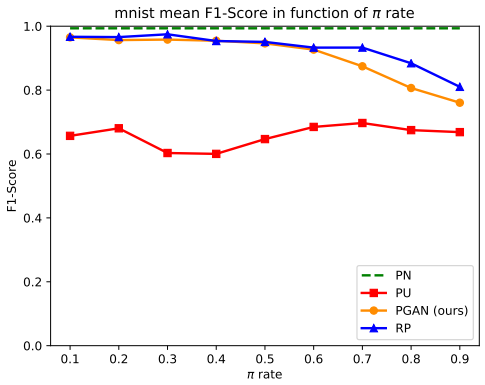
Figure 3.6 presents the study of the steadiness of PGAN. Figures 3.6(a) and 3.6(b) show that PGAN has a comparatively steadier behavior such that we can predict more easily the F1-Score evolution as a function of π for each dataset class. Figure 3.6(e) shows that the classifier stabilizes and converges after about 10 training epochs. To generate histograms

Table 3.2: F1-Score results comparisons on MNIST, Fashion-MNIST and CIFAR-10 after 20 epochs of the classifier.

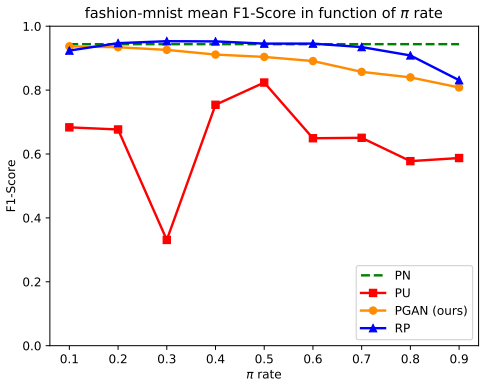
	ref	$\rho = 0.5$, $\pi = 0.1$			$\rho = 0.5$, $\pi = 0.3$			$\rho = 0.5$, $\pi = 0.5$			$\rho = 0.5$, $\pi = 0.7$		
Dataset	PN	PU	PGAN	RP	PU	PGAN	RP	PU	PGAN	RP	PU	PGAN	RP
0	0.997	0.633	0.974	**0.992**	0.445	**0.973**	0.955	0.320	0.973	**0.991**	0.689	**0.902**	0.880
1	0.998	0.774	0.971	**0.995**	0.642	0.979	**0.996**	0.851	0.958	**0.994**	0.884	0.863	**0.993**
2	0.990	0.395	0.972	**0.975**	0.658	**0.959**	0.923	0.795	**0.947**	0.936	0.692	0.914	**0.987**
3	0.996	0.716	0.963	**0.991**	0.620	0.953	**0.991**	0.766	**0.934**	0.882	0.729	**0.885**	0.829
4	0.997	0.512	0.964	**0.972**	0.802	0.952	**0.995**	0.717	**0.945**	0.933	0.551	0.914	**0.977**
5	0.993	0.701	0.974	**0.985**	0.725	**0.950**	0.943	0.799	**0.949**	0.910	0.626	0.873	**0.973**
6	0.992	0.708	**0.962**	0.928	0.758	0.959	**0.992**	0.699	0.971	**0.993**	0.613	0.944	**0.990**
7	0.995	0.603	**0.962**	0.947	0.433	0.960	**0.991**	0.620	0.926	**0.988**	0.783	0.737	**0.979**
8	0.995	0.741	**0.949**	0.929	0.506	0.941	**0.982**	0.339	0.922	**0.941**	0.651	**0.849**	0.818
9	0.981	0.785	**0.959**	0.954	0.442	0.956	**0.979**	0.561	0.939	**0.941**	0.750	0.865	**0.904**
AVG_{MNIST}	0.993	0.657	0.965	**0.967**	0.603	0.958	**0.975**	0.647	0.946	**0.951**	0.697	0.875	**0.933**
T-shirt/top	0.908	0.724	**0.926**	0.899	0.206	0.91	**0.937**	0.821	0.873	**0.947**	0.695	0.802	**0.91**
Trouser	0.993	0.815	0.983	**0.993**	0.247	0.969	**0.989**	0.938	0.953	**0.99**	0.681	0.911	**0.984**
Pullover	0.932	0.635	**0.9**	0.887	0.29	0.885	**0.925**	0.695	0.865	**0.917**	0.657	0.842	**0.888**
Dress	0.952	0.601	0.941	**0.948**	0.312	0.925	**0.955**	0.852	0.893	**0.914**	0.631	0.853	**0.882**
Coat	0.882	0.614	**0.909**	0.847	0.252	0.889	**0.942**	0.788	0.845	**0.92**	0.686	0.83	**0.918**
Sandal	0.995	0.793	0.945	**0.977**	0.444	0.964	**0.98**	0.819	0.923	**0.985**	0.67	0.919	**0.981**
Shirt	0.818	0.446	**0.852**	0.758	0.398	0.846	**0.847**	0.797	0.819	**0.873**	0.554	0.792	**0.853**
Sneaker	0.983	0.73	**0.973**	**0.973**	0.271	0.952	**0.979**	0.865	0.943	**0.967**	0.64	0.922	**0.977**
Bag	0.989	0.772	**0.978**	0.976	0.536	0.947	**0.99**	0.837	0.96	**0.965**	0.685	0.757	**0.977**
Ankle boot	0.985	0.704	0.964	**0.979**	0.354	0.973	**0.986**	0.824	0.963	**0.976**	0.609	0.942	**0.976**
$AVG_{\text{F-MNIST}}$	0.944	0.683	**0.937**	0.924	0.331	0.926	**0.953**	0.824	0.904	**0.945**	0.651	0.857	**0.935**
Plane	0.727	0.341	**0.818**	0.669	0.557	0.784	**0.795**	0.295	**0.758**	0.743	0.621	**0.731**	0.718
Auto	0.78	0.506	**0.801**	0.695	0.492	0.737	**0.829**	0.414	0.789	**0.798**	0.521	0.734	**0.783**
Bird	0.447	0.175	**0.688**	0.56	0.439	**0.744**	0.68	0.184	**0.694**	0.644	0.359	**0.688**	0.542
Cat	0.5	0.125	**0.658**	0.384	0.272	**0.722**	0.651	0.249	**0.718**	0.67	0.446	0.69	**0.698**
Deer	0.698	0.272	**0.68**	0.605	0.232	**0.708**	**0.708**	0.3	**0.708**	0.64	0.43	**0.633**	0.602
Dog	0.567	0.2	**0.632**	0.539	0.37	**0.756**	0.648	0.258	**0.746**	0.733	0.514	0.678	**0.712**
Frog	0.691	0.35	**0.837**	0.666	0.418	0.793	**0.794**	0.256	**0.788**	0.769	0.693	**0.75**	0.749
Horse	0.786	0.373	**0.693**	0.653	0.515	**0.757**	0.723	0.26	0.751	**0.759**	0.611	0.675	**0.711**
Ship	0.832	0.313	**0.821**	0.764	0.565	0.809	**0.831**	0.324	0.775	**0.785**	0.623	0.716	**0.755**
Truck	0.771	0.462	**0.822**	0.685	0.367	**0.786**	0.637	0.272	**0.754**	0.617	0.539	**0.724**	0.564
$AVG_{\text{CIFAR-10}}$	0.680	0.312	**0.745**	0.622	0.423	**0.760**	0.730	0.281	**0.748**	0.716	0.536	**0.702**	0.684

in Figures 3.6(c) and 3.6(d), we have retrieved the scalar of the second output neuron of the classifier which corresponds to the predicted probability for an input image belonging to the positive class. It can be seen that the prediction distribution of the negative and positive test samples processed by the PGAN is of the Gaussian type, which is an interesting characteristic for real applications. Compared to a biased PU training (see Fig. 3.6(c)), PGAN plays the role of a regularizer.

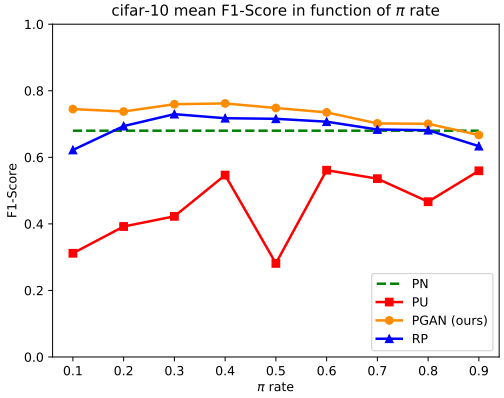
Moreover, Table 3.3 presents the prediction performance stability for RP and PGAN, depending on π, for the three datasets. To this end, we have respectively smoothed the curves $s(\pi)$ of each class representing the F1-Score evolution as a function of π. Smoothed curves $\tilde{s}(\pi)$ have been obtained using a mean filter with a kernel size of 3. Then, the mean



(a)



(b)



(c)

Figure 3.5: Average F1-Scores after 20 training epoch iterations of the classifier depending on the rate π that is varied between 0 and 1 with a step of 0.1, for PU (red), RP (blue) and PGAN (orange) on (a) MNIST, (b) Fashion-MNIST and (c) CIFAR-10.

squared error MSE for every class is estimated between $\tilde{s}(\pi)$ and $s(\pi)$, such that

$$MSE = \frac{1}{k}\sum_{i=1}^{k}(\tilde{s}^{(i)} - s^{(i)})^2, \tag{3.18}$$

with k the number of samples of $\tilde{s}(\pi)$. Then, for every dataset, we can compute the mean errors E_{PGAN} and E_{RP} over their 10 One-vs-Rest tasks respectively, as well as the ratio $E_{RP} : E_{PGAN}$. In this way we can observe that the proposed approach has a systematically steadier behaviour, by a factor four on MNIST and CIFAR-10.

In view of the interesting results obtained on CIFAR-10, we have also compared the PGAN with a PN training on the STL-10 dataset [31]. The main specificity of this dataset is that it provides only 500 labeled images for each class and 1 000 000 unlabeled images. This enables to test our method under conditions which are similar to a real application case,

Table 3.3: F1-Score predictions stability of PGAN and RP on MNIST, Fashion-MNIST and CIFAR-10 datasets.

Datasets	E_{PGAN} ($\times 10^{-3}$)	E_{RP} ($\times 10^{-3}$)	$\frac{E_{RP}}{E_{PGAN}}$
MNIST	**0.39**	1.72	4.410
Fashion-MNIST	**0.16**	0.25	1.563
CIFAR-10	**0.44**	1.82	4.136

and to highlight its relevance from performance and stability perspectives. Fig. 3.7 presents the training evolution of the PGAN classifier Accuracy depending on the first-stage. For this purpose, we have trained entirely the classifier D_B during 35 epochs from stratch for every GAN first stage epoch. Concerning this first stage, the WGAN-GP [66][3] framework is used. Concerning the second step, the classifier D_B maintains the same structure as the discriminator D_U used during the first stage. This empirically demonstrates that the proposed approach is as well valuable when the classifier D_B has a capacity to encode complexity equivalent to the discriminator D_U. In Table 3.4, we observe that the proposed approach significantly outperforms a PN training using the same classifier architecture. We can also observe that there is no first-stage overfitting issue occuring. Based on generated images presented in Fig. 3.8, it is also interesting to observe that it is not necessary to obtain generated images which are visually acceptable from a qualitative human point of view for the proper operation and convergence of the proposed system.

Table 3.4: Accuracy test of the classifier D_B after 35 epochs, comparing a PN training which only uses 500 positive and 4500 negative labeled images, versus a PGAN using 500 labeled positive images and one million of unlabeled images.

STL-10 (One vs. Rest)	PGAN	PN
training settings	{500 P ; 1 000 000 U}	{500 P ; 4 500 N}
airplane	**0.805**	0.674
bird	**0.64**	0.574
car	**0.794**	0.645
cat	**0.614**	0.521
deer	**0.65**	0.561
dog	**0.605**	0.529
horse	**0.721**	0.632
monkey	**0.674**	0.539
ship	**0.787**	0.633
truck	**0.724**	0.594
average	**0.701**	0.59

[3]Official implementation available at: *https* : *//github.com/igul*222/*improved_wgan_training*.

3.4 Conclusion

In this chapter, we demonstrated that the proposed PGAN PU learning approach provides state-of-the-art prediction performances and has a steady behavior on complex image datasets up to an acceptable fraction π of positive samples in the unlabeled training dataset. Moreover, the results obtained on natural image datasets as CIFAR-10 and STL-10 are consistent with the proposed framework reasoning and thus enable to consider PU applications on high dimensional data without using prior knowledge concerning the unlabeled set.

However, an issue of the proposed framework is the first-stage overfitting which can appear on simple datasets as MNIST. As the semantic complexity of images to be processed is a notion very difficult to empirically check upstream, we propose in the next chapter a second GAN-based PU approach which overcomes the PGAN first-stage overfitting, such that its usefulness does not depend on the data complexity to deal with.

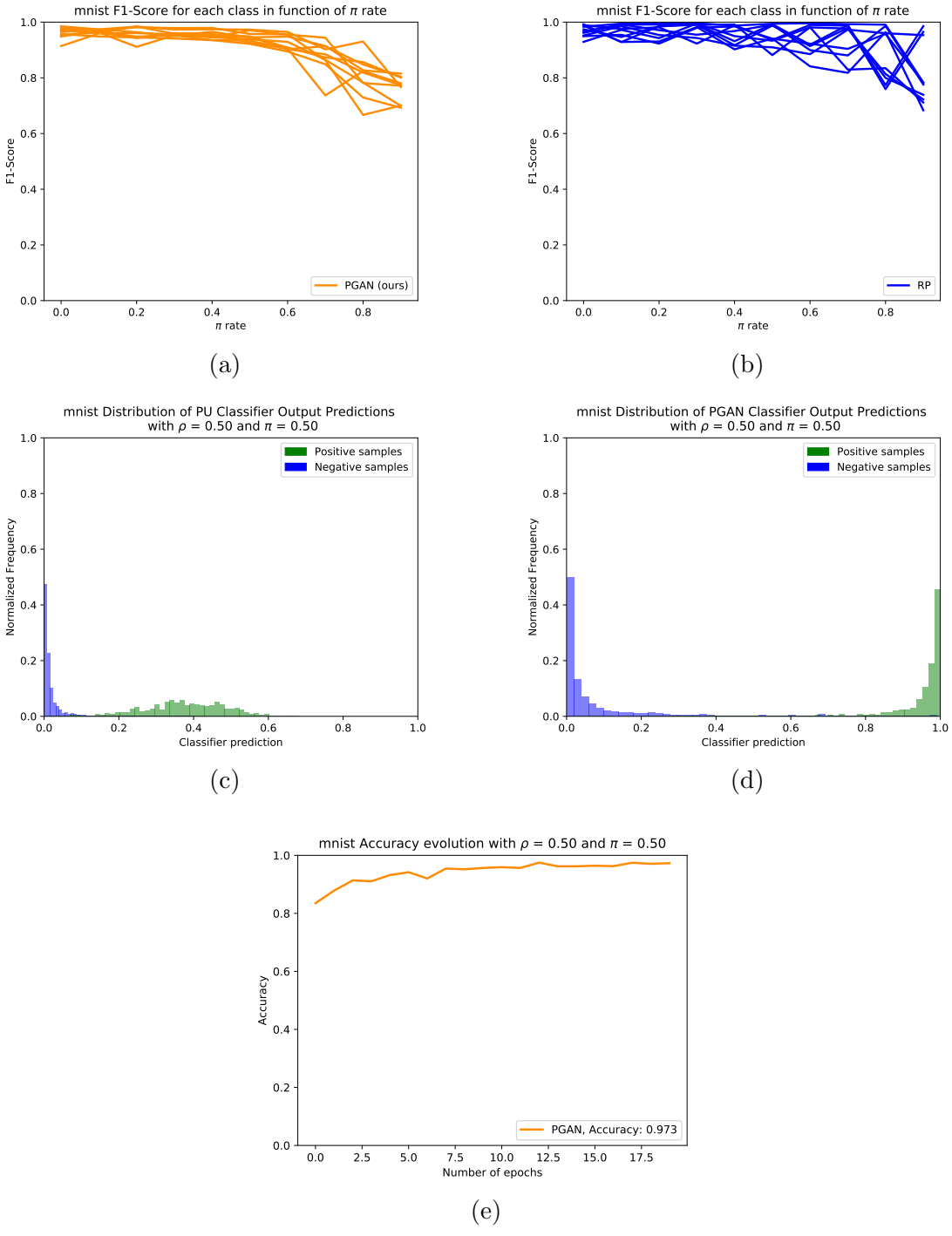


Figure 3.6: Stability analysis on MNIST. F1-Score evolution for each class as a function of π for (a) PGAN, and (b) RP [127]. (c) and (d) are the histogram of the output predictions of the classifier trained in PU and PGAN modes at its 20th epoch iteration for positive (green) and negative (blue) test samples. The positive class is "5" and $\pi = 0.5$. (e) shows the Accuracy evolution during the PGAN training for the same *One-vs-Rest* task.

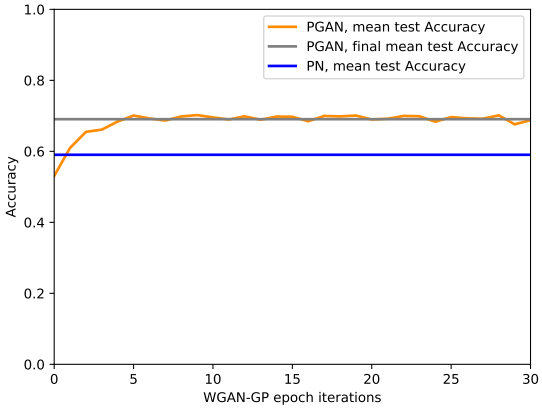


Figure 3.7: Evolution of the average of the test Accuracy for every class, depending on training iterations of the WGAN-GP on unlabeled images of the dataset STL-10.

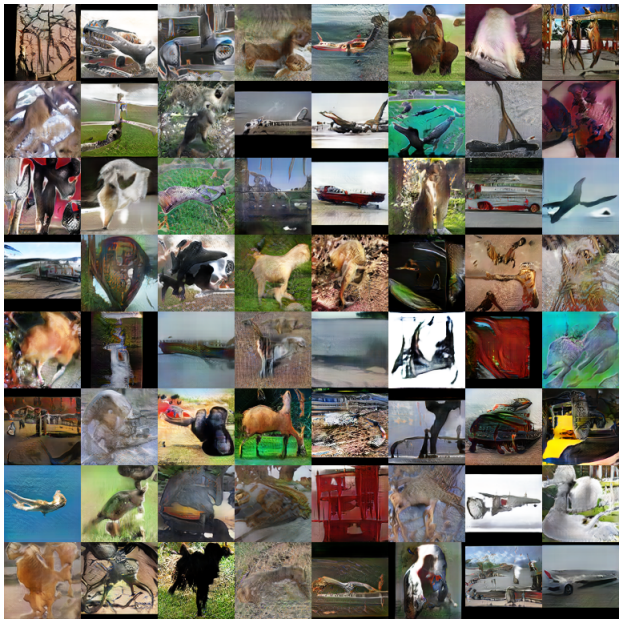


Figure 3.8: Generated images (96*96*3) with the WGAN-GP after 25 epochs on the 1 000 000 unlabeled images on the dataset STL-10. Although visually unacceptable for a human analysis, these images are relevant from the D_B classifier point of view, as shown quantitatively in Table 3.4.

Chapter 4

Counter-examples generation from a Positive Unlabeled dataset

Contents

The previously presented GAN-based two-stage approach can deal with a Positive Unlabeled (PU) dataset without a cumbersome architecture or requiring prior knowledge. However, this strategy may suffer from a first-stage overfitting depending on the learning abilities of the models to encode the information complexity of the image data to process.

To overcome this first-stage overfitting issue, while also demonstrating prior knowledge insensitivity, we propose in this chapter to incorporate a biased PU risk within the standard GAN discriminator loss function. In this manner, the discriminator is constrained to request the generator to converge towards the unlabeled samples distribution while diverging from the positive samples distribution. This enables the proposed model, referred to as D-GAN, to exclusively learn the counter-examples distribution without prior knowledge. Experiments demonstrate that our approach outperforms state-of-the-art PU methods.

4.1 Introduction

Nowadays, the number of available labeled datasets dedicated to perception applications such as image classification [150] and semantic scene understanding [32] has considerably augmented. However, when learning methods trained on these datasets are applied on real data, their performances are likely to deteriorate. Consequently, it is necessary to use a dataset specialized for the given target application. It turns out that it can be easy to get unlabeled data in some applications domains such as autonomous driving. Positive Unlabeled (PU) learning, also called *partially supervised classification* [103], enables to use these unlabeled data in combination with labeled samples of our class of interest: the positive class. The interest is that unlabeled data can contain relevant counter-examples, also called negative examples[1]. The difficulty is that unlabeled data can also contain a fraction π_p of unlabeled positive examples. [153] enumerates several learning problems which can be addressed in this way such as the challenging information retrieval task.

Several PU learning methods exist. While some of them are adapted to time series [38] or text classification, for instance by using non-negative matrix factorization [99], we focus in this chapter on those that are applied to image classification using deep neural networks. They are generally classified into two categories. The former is censoring PU learning, formalized in [47] and recently improved in [127]. The latter is case-control PU learning, introduced in [164], formalized in [44], and then consecutively improved in [43] and [91] to reduce the training computational cost and alleviate the overfitting issue. In the context of the proposed approach, we focus our attention in this chapter on the recently presented GAN-based PU approaches. Thus, we classify PU learning approaches into the two following groups suggested in [91]: one-stage and two-stage PU methods.

One-stage PU methods, such as the unbiased PU method (uPU) [43] and the non-negative PU method (nnPU) [91], consist of training a classifier using an unbiased risk directly on the PU dataset. These methods have the advantage to need only one training of the classifier. However, they require dataset prior knowledge and consequently uPU and nnPU need to be combined with an approach estimating the prior knowledge [29]. Consequently, they are critically sensitive to slight prior variations per minibatch, as shown experimentally in Section 4.4.3.1.

Two-stage PU methods prepare during the first stage a Positive Negative (PN) dataset. Some noisy labeled learning strategies consist of detecting automatically the most plausible mislabeled examples [46] through an iterative process in order to accelerate the manual PN labeling. Some other methods can prepare automatically a PN dataset without human

[1] We use the term *example* to design a single instance (i.e. item, observation) included in a sample set of data following a given distribution.

supervision from end-to-end. For instance, Rank Pruning method (RP) [127] firstly estimates the prior such that it can select only the examples considered as the most confident, in order to substitute the unlabeled samples for the second-stage training of the classifier. RP achieves two-stage state-of-the-art performances without prior knowledge. However, it can only exploit a sub part of the training PU dataset. This can curb its prediction performances on complex datasets like CIFAR-10. Recently, a new subcategory of two-stage PU methods appeared: GAN-based PU methods. They address the PU learning challenge by producing, thanks to an adversarial training [64], generated samples from a PU dataset during the first step. Then, they are used to train a standard Positive Negative (PN) classifier during the second step.

We discuss in more details the previously introduced PU methods (see chapter 3) uPU [43], nnPU [91], RP [127], and GenPU [74], and we also recall PGAN [26] proposed approach (see chapter 3), in the related work Section 4.2.

We can nonetheless already make the following remarks, motivating the design of the proposed approach. Unbiased methods [91], [43], and GenPU [74] are by definition sensitive to the prior knowledge in order to deal with a PU dataset. Conversely, whereas the two-stage censoring methods, such as RP [127], do not require prior information, they suffer from generalization and unstability problems due to their selective process. We recall that PGAN method, previously proposed in chapter 3, is the first one that does not need prior knowledge nor a selective process, thus preserving a low sensitivity to prior knowledge combined with a training stability. However, as mentioned in chapter 3, PGAN inherently suffers from first-stage overfitting. Based on these considerations, we propose in this chapter a novel GAN-based model, referred to as Divergent-GAN (D-GAN), to overcome the latter issue while preserving the PGAN advantages. To the best of our knowledge, we are the first to propose a GAN-based method to capture exclusively the unlabeled negative samples distribution from a PU dataset without prior knowledge. More specifically, our contributions are the following:

- We propose to incorporate a biased PU learning loss function inside the original GAN [64] discriminator loss function. The intuition behind it is to have the generative model solving the PU learning problem formulated in the discriminator loss function. In this way, the generator learns the distribution of the examples which are both unlabeled and not positive, namely the negative ones included in the unlabeled dataset;

- In addition, we study normalization techniques compatibility with the proposed framework. A learning model which manipulates different minibatches distributions should

not use batch normalization techniques [77]. Alternative normalization techniques are discussed and experimented.

Consequently, the proposed D-GAN framework compares favorably with state-of-the-art PU learning performances on simple MNIST [96] and complex CIFAR-10 [92] image datasets. The proposed framework code is available.

The remaining of this chapter is structured as follow. Section 4.2 presents previous PU learning approaches. Section 5.2 describes the proposed method. Section 5.3 presents the corresponding experimental results. Finally, in Section 4.5, we draw conclusions and discuss perspectives.

4.2 Related work

The PU learning problem consists of trying to distinguish positive samples from negative ones by using a PU dataset. Let $X \in \mathbb{R}^m$ be the input random variable and $Y \in \{0, 1\}$ its associated label. X can be a positive X_P, negative X_N or unlabeled X_U sample which respectively follow the distributions $p_P = p(X|Y = 0)$, $p_N = p(X|Y = 1)$ and $p_U = (1 - \pi_P) \cdot p_N + \pi_P \cdot p_P$. The unknown prior $\pi_P \in (0, 1)$ represents the fraction of unlabeled positive examples included in the unlabeled dataset.

Previous works on PU learning [39] consider the entire distribution of the unlabeled examples as negative. In this way, all the negative examples, present in the unlabeled dataset, are always considered as negative. However, concerning the positive examples, it implies associating two contradictory labels to the distribution of positive examples in unknown proportions depending on the π_P value. Thus, training directly a classifier with positive and unlabeled data provokes a bias in the training estimator, which is not present during a standard positive negative training. This bias can limit prediction performances of the learning model.

Several strategies have been proposed to solve this drawback such as unbiased methods [44], [43], [91], pruning method [127], and more recently GAN based methods [74], [26]. However, those strategies still present some issues including prior knowledge sensitivity, training unsteadiness, or overfitting problems.

We present in this section different state-of-the-art methods and their respective drawbacks that we aim at overcoming with the proposed GAN-based PU framework.

4.2.1 Unbiased methods

In order to palliate a biased training, the authors of unbiased techniques [44], [43], [91] suggest to avoid the estimator bias by adding some terms in the training loss function.

Then, the classifier behaves as if it is trained with a positive negative dataset. The authors firstly used a non convex loss function [44], which then has been reformulated as convex loss functions [43] in order to reduce the computational burden. Subsequently, it was proposed to overcome the training overfitting by adding a binary condition (an "if" condition) in the training loss function [91].

These methods exploit the prior π_P in the empirical training loss function. However, we observe that the empirical prior value $\hat{\pi}_P$ per batch of small size (minibatch) is slightly different to π_P, as its standard deviation depends on the minibatch size, such that:

$$\hat{\pi}_P = \pi_P + \alpha, \tag{4.1}$$

with $\alpha \sim p_\alpha(m)$, where p_α is the probability distribution of the noise α depending on the minibatch size m, as shown in Figure 4.1. We observe that the worst case scenario is when π_P is close to the value 0.5, combined with a small batch size. The cases where π_P is higher than 0.5 behave symmetrically to the cases where π_P is smaller than 0.5.

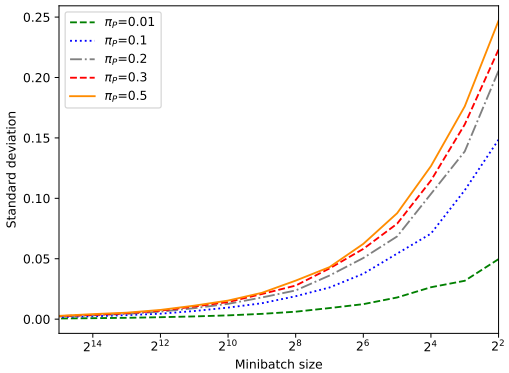


Figure 4.1: Standard deviation per minibatch of the global prior π_P in function of the minibatch size, for a given uniformly mixed dataset composed by 60000 examples.

In our case, we want to train a deep learning model using the stochastic gradient descent (SGD) optimization technique, which is known to be relevant with batches of small size. So the theoretical formulation of unbiased techniques cannot be maintained using SGD with small batch sizes. We will show empirically that in practice, unbiased techniques are highly sensitive to the minibatch size in terms of prediction performances, as they are theoretically sensitive to the prior π_P.

It turns out that it is possible to avoid this limitation with two-stage approaches.

4.2.2 Two-stage approaches

Two-stage approaches mainly consist of preparing during the first-stage a positive negative (PN) training dataset which then will be used to directly train a standard classifier during the second stage. One interest of those approaches is that they are not sensitive to the prior knowledge variation. Consequently, they are compatible with the use of minibatches, and thus are suitable when applying SGD optimization.

4.2.2.1 Pruning approach

Rank Pruning (RP) method [127] is a two-stage technique. It first estimates the prior π_P and exploits it to prune the dataset in order to capture only a subset corresponding to the most confident positive and negative samples. Then, during the second stage it considers this subset as a cleanly labeled positive negative dataset to train a classifier. While not requiring prior knowledge in input, RP achieved state-of-the-art results for information retrieval in the One-vs-Rest task on simple datasets such as MNIST. However, by using a pruning strategy, RP can miss some relevant training examples not included in the selected subset of training. As a consequence, this can limit its generalization, as will be shown experimentally in Table 4.4, where RP is shown to be relatively unstable when compared to GAN-based approaches in terms of prediction performances. Using only a training subset is also a weakness on complex datasets like CIFAR-10, where a large training dataset is preferable to obtain better results.

Some approaches have been more recently proposed by exploiting generative adversarial networks (GANs) benefits, maintaining or increasing the prediction scores over the same PU learning tasks.

4.2.2.2 GAN-based approaches

GAN-based PU approaches represent a recent subcategory of two-stage PU methods, as proposed in GenPU [74] and PGAN [26] chapter 3. The interest of using GANs is twofold. First, GANs enable relevant data augmentation, as will be experimentally demonstrated on Table 4.4. Second, it allows for the use of high-level feature metrics to evaluate generated samples quality, thanks to the adversarial training. This can ease to capture a target distribution in a meaningful manner.

In this PU learning context, the generated samples replace the unlabeled ones by learning on the latter as PGAN [26], or on both unlabeled and positive labeled ones as GenPU [74]. Both methods exploit GANs benefits, but their functioning is different and they are not suitable under the same datasets conditions.

GenPU [74] is based on the original GAN convergence [64], such that: $\pi_P \cdot p_{G_P} + (1 - \pi_P) \cdot p_{G_N} \rightarrow p_U$, with p_{G_P} the distribution of positive samples generated by the generator G_P, p_{G_N} the distribution of the negative samples generated by the generator G_N, and p_U the distribution of real unlabeled samples. In practice, GenPU is an interesting PU method on simple datasets with few positive labeled samples, and it generates relevant counter-examples. However, training adversarially five learning models instead of two as in the original GAN framework [64] to address *standard* PU learning challenge[2] is more computational demanding and not necessary to generate relevant counter-examples. Moreover, using five models amplifies the mode collapse issue, and the corresponding training optimization functions need three additional hyper-parameters combined with prior knowledge. This is impractical in the context of real applications where hyper-parameters tuning may be required on limited computational resources to adapt the model for a given application dataset.

Concerning the previously proposed approach **PGAN** [26] (see chapter 3), we recall that it is trained to converge towards the unlabeled dataset distribution during the first step. During the second step, it exploits GANs imperfections for capturing the unlabeled distribution, such that the generated distribution at the adversarial equilibrium is still separable from the unlabeled samples distribution by a classifier. It presents a relatively steadier behaviour and better prediction performances than the two-stage baseline RP method on the complex RGB image dataset CIFAR-10 without prior knowledge. However, it is less suitable for relatively simpler datasets like MNIST. The problem is that the generated samples are all considered as negative samples by the classifier. But this is possible only if the generated samples distribution converges close enough towards the unlabeled samples distribution, while not matching it. If the PGAN first-stage performs as expected theoretically by [64], then the PGAN classification second stage falls back into the initial PU learning problem.

Our proposed approach in this chapter, presented in Sec. 5.2, overcomes previously enumerated PU methods shortcomings, as summarized in Table 4.1.

4.3 Proposed Approach

In this section, we first briefly recall the main reasoning which motivated our research work. Next, we discuss some features of a biased PU risk. We then propose to incorporate this risk into a generic GAN framework in order to guide the generator convergence towards the negative samples distribution, denoted as p_N, included inside the unlabeled dataset

[2]We use the term *standard* to refer to the case where we have enough positive labeled examples (at least 100), such that the difficulty is mainly the ability to exploit counter-examples included in the unlabeled set.

Methods	D-GAN (proposed)	PGAN (chapter 3)	GenPU [74]	RP [127]	nnPU [91]
No need of priori knowledge	√	√		√	
No first-stage overfitting	√		√	√	√
Generalizable over complex datasets	√	√			
Able to generate relevant counter-examples	√		√		
Training stability using SGD	√	√		√	
Original GAN architecture	√	√			
Code availability	√			√	√

Table 4.1: Summary of presented state-of-the-art methods advantages and drawbacks compared to the proposed D-GAN approach. A void cell means that the mentioned criterium is not applicable with the corresponding method.

distribution, denoted as p_U. Furthermore, we study regularization techniques to manipulate three distinct types of minibatches: positive, unlabeled and generated ones.

4.3.1 Motivation

In PU learning, if a classifier associates a given expected label value with positive examples, and in parallel associates a second distinct label value with unlabeled examples, then it is proven that the negative non-labeled examples are exclusively associated with the label of non-labeled examples [39]. Concerning GANs, it has been shown that the discriminator learning task influences directly the adversarial generator behaviour [114]. Based on these considerations, this work aims at incorporating a biased PU risk inside the traditional GAN discriminator cost function. This compels the discriminator D, to separate negative from positive distributions, which in turn guides the generator G, to exclusively learn the unlabeled counter-examples distribution from a PU dataset. As a matter of the fact, the proposed method is novel in the way it exclusively generates relevant counter-examples without prior knowledge information, while preserving a standard GAN architecture.

Thereafter, we present the biased PU risk that we incorporate in the proposed GAN PU discriminator training loss function.

4.3.2 Biased PU risk to incorporate

In what follows, we first explain the expected PU functionality to be incorporated into the GAN discriminator loss function.

Biased PU risk setting: Let $D : \mathbb{R}^m \to [0, 1]$ be the decision function which is, later on, considered as the discriminator D, of the proposed framework network. We have $l(\hat{y}, y)$ such that $l : [0, 1] \times [0, 1] \to \mathbb{R}$ is the arbitrary cost function with the predicted output $\hat{y}$ of D for a given example and the corresponding label y as input. D is trained with a PU risk R_{PU} to predict the label value 1 for the unlabeled examples, and the label value 0 for the

positive labeled ones such that:

$$R_{PU}(D) = \mathbb{E}_{x_U \sim \boldsymbol{p_U}}[l(D(x_U), 1)] + \mathbb{E}_{x_P \sim p_P}[l(D(x_P), 0)]. \tag{4.2}$$

Given the composition of the distribution p_U, we develop:

$$\begin{aligned} R_{PU}(D) =& (1 - \pi_P) \cdot \mathbb{E}_{x_N \sim p_N}[l(D(x_N), 1)] + \pi_P \cdot \mathbb{E}_{x_P \sim p_P}[l(D(x_P), 1)] \\ &+ \mathbb{E}_{x_P \sim p_P}[l(D(x_P), 0)]. \end{aligned} \tag{4.3}$$

Counter-examples are correctly labeled: Decomposed in this way, the negative examples included in the unlabeled dataset are associated exclusively to the label value 1 for any π_P value, such that the negative training examples are all correctly labeled. When there is no overfitting on training positive examples, then one can assume that labeled and unlabeled positive examples follow the same distribution p_P, as mentioned in [91]. Since expectations are linear, p_P is associated to both contradictory labels 0 and 1 as below:

$$R_{PU}(D) = \mathbb{E}_{x_N \sim p_N}[(1 - \pi_P) l(D(x_N), 1)] + \mathbb{E}_{x_P \sim p_P}[\pi_P l(D(x_P), 1) + l(D(x_P), 0)]. \tag{4.4}$$

Positive samples distribution p_P is shifted away from the counter-examples distribution p_N: When defining the cost-function l as the binary cross-entropy H (Eq. 4.5) such that $l = H$, then we can demonstrate that the second term in the Eq. (4.4) is equivalent to associating the positive distribution p_P with a unified biased intermediate label value δ. The binary cross-entropy H is defined as:

$$H(D(X), Y) = -Y log(D(X)) - (1 - Y) log(1 - D(X)), \tag{4.5}$$

where Y is the label value associated with the input X of D. If $l = H$, then concerning the second term of the Eq. (4.4), we can demonstrate that:

$$\begin{aligned} \pi_P H(D(x_P), 1) + 1H(D(x_P), 0) =& -\pi_P log(D(x_P)) - 1 log(1 - D(x_P)) \\ =& -\pi_P log(D(x_P)) - (1 + \pi_P - \pi_P) log(1 - D(x_P)) \\ =& (1 + \pi_P) \cdot \Big[-\frac{\pi_P}{1 + \pi_P} log(D(x_P)) \\ & - (1 - \frac{\pi_P}{1 + \pi_P}) log(1 - D(x_P)) \Big] \\ =& (1 + \pi_P) \cdot H\big(D(x_P), \frac{\pi_P}{1 + \pi_P}\big) \\ =& (1 + \pi_P) \cdot H(D(x_P), \delta), \end{aligned} \tag{4.6}$$

with $\delta = \pi_P / (1 + \pi_P)$. Consequently, the PU risk becomes:

$$R_{PU}(D) = \mathbb{E}_{x_N \sim p_N}[(1 - \pi_P) H(D(x_N), 1)] + \mathbb{E}_{x_P \sim p_P}[(1 + \pi_P) H(D(x_P), \delta)]. \tag{4.7}$$

Such a PU risk has been previously called biased or constrained in the literature [103]. The equivalence between Eqs. (4.4) and (4.7) makes it possible to estimate the restricted interval of possible values for δ without using prior such that if $\pi_P \in (0, 1)$ then:

$$0 < \pi_P < 1 \Leftrightarrow 0 < \delta < \frac{1}{1+1}. \tag{4.8}$$

In other words, $\delta \in (0, 1/2)$. This confirms that for any π_P value between 0 and 1, labeled and unlabeled positive examples are associated with a label value δ comprised between 0 and 1/2. Therefore, when training D with the risk R_{PU}, the D prediction related to the unlabeled positive examples is shifted away from the label value 1. From D prediction output point of view, this risk makes the positive distribution p_P *diverging* from the negative distribution p_N. Thus, D is trained to predict the label value 1 exclusively for the counter-examples.

4.3.3 Proposed generative model

The insight in the proposed D-GAN model can be expressed as follows: D addresses to G the riddle: *Show me what IS unlabeled AND NOT positive.* It turns out that negative examples included in the unlabeled dataset are both unlabeled and not positive. Consequently, G addresses this riddle by learning to show the negative samples distribution to D.

GAN background: We first give a short recall of the original GAN discriminator. It is trained to distinguish real unlabeled samples distribution p_U from generated samples distribution p_G with the loss function $L_{D_{GAN}}$ defined as:

$$L_{D_{GAN}}(G, D) = \mathbb{E}_{x_U \sim p_U}[-logD(x_U)] + \mathbb{E}_{z \sim p_z}[-log(1 - D(G(z)))], \tag{4.9}$$

where z stands for the input random vector of the generative model G such that $G(z)$ is a generated sample. z follows a uniform or normal distribution. It turns out that the binary cross-entropy formulation (Eq. 4.5) implies $H(D(X), 1) = -log(D(X))$ and $H(D(X), 0) = -log(1 - D(X))$. Consequently, $L_{D_{GAN}}$ can be expressed as follows:

$$L_{D_{GAN}}(G, D) = \mathbb{E}_{x_U \sim p_U}[H(D(x_U), 1)] + \mathbb{E}_{z \sim p_z}[H(D(G(z)), 0)]. \tag{4.10}$$

Towards a GAN biased discriminator loss function: The proposed approach aims at training G to learn the negative samples distribution p_N instead of learning the distribution p_U. This replaces the standard GAN task *"Show me what is unlabeled "*, by the task *"Show me what is both unlabeled and not positive"*. We now propose to incorporate the benefits of a biased PU risk (Eq. 4.2) into the original GAN discriminator loss function (Eq. 4.9). To this end, we define the D-GAN discriminator loss function L_D by adding the

term $\mathbb{E}_{x_P \sim p_P}[H(D(x_P), 0)]$ to $L_{D_{GAN}}$. Consequently, in the proposed D-GAN framework, the training discriminator loss function L_D of D becomes:

$$L_D(G, D) = L_{D_{GAN}}(G, D) + \mathbb{E}_{x_P \sim p_P}[H(D(x_P), 0)]. \tag{4.11}$$

If we develop the term $L_{D_{GAN}}$, we then obtain:

$$\begin{aligned} L_D(G, D) =& \mathbb{E}_{x_U \sim p_U}[H(D(x_U), 1)] + \mathbb{E}_{z \sim p_z}[H(D(G(z)), 0)] + \mathbb{E}_{x_P \sim p_P}[H(D(x_P), 0)] \\ =& R_{PU}(D) + \mathbb{E}_{z \sim p_z}[H(D(G(z)), 0)]. \end{aligned} \tag{4.12}$$

In other words, the R_{PU} risk (Eq. 4.2) is incorporated inside the D-GAN discriminator loss function. To this extent, D can be trained to only consider the counter-examples as the most real examples by associating to them exclusively the label value 1. This can be considered as applying a constrained optimization.

The generator generates the counter-examples distribution: In contrast, the role of G during the adversarial training is to generate samples considered by D as 1. As proposed in [64], the training loss function L_G of G is such that:

$$\begin{aligned} L_G(G, D) &= \mathbb{E}_{z \sim p_z}[-log(D(G(z)))] \\ &= \mathbb{E}_{z \sim p_z}[H(D(G(z)), 1)]. \end{aligned} \tag{4.13}$$

We recall that D exclusively considers the negative examples as 1 thanks to the R_{PU} risk introduced in Eq. (4.2). Thus, if D trainable weights are fixed in the proposed framework, then we propose to reinterpret in L_G the label value 1 as $D(x_N)$, as follows:

$$\begin{aligned} L_G(G, D) &= \mathbb{E}_{z \sim p_z, x_N \sim p_N}[H(D(G(z)), D(x_N))] \\ &= \mathbb{E}_{z \sim p_z, x_N \sim p_N}[-D(x_N)log(D(G(z)))], \end{aligned} \tag{4.14}$$

such that the distance between the generated samples distribution and p_N is minimized. Consequently, this justifies the convergence of G in the proposed D-GAN framework towards the negative samples distribution p_N, for any $\pi_P \in (0, 1)$.

Implementation: The corresponding implementation Algorithm 2 of the proposed first-stage D-GAN approach enables to adversarially train D and G to respectively minimize loss functions L_D and L_G.

Second-stage: Positive-Generative learning. Once the D-GAN training is completed, the second step can be carried out. It consists of training a classifier C to distinguish fake generated examples $x_{FN} = G(z)$, which are ideally equivalent to the real negative samples, from real positive labeled samples as illustrated in Figure 4.2.

In practice, the worst-case scenario is when D overfits the positive examples during the adversarial training. Another pitfall is when D cannot encode the complexity of the boundary between positive and negative examples included in the unlabeled dataset. In such

Algorithm 2 Minibatch SGD training of the D-GAN

GAN training (1^{st} step)

for number of training iterations **do**

Sample minibatch of m noise samples $\{z^{(1)}, ..., z^{(m)}\}$ from noise prior p_z.

Sample minibatch of m unlabeled examples $\{x_U^{(1)}, ..., x_U^{(m)}\}$ from data distribution p_U.

Sample minibatch of m positive labeled examples $\{x_P^{(1)}, ..., x_P^{(m)}\}$ from data distribution p_P.

Update D by descending its stochastic gradient:

$$\nabla_{\theta_D} \frac{1}{m} \sum_{i=0}^{m} \Big[-logD(x_U^{(i)}) - log\big[1 - D(G(z^{(i)}))\big] - log\big[1 - D(x_p^{(i)})\big] \Big]$$

Sample minibatch of m noise samples $\{z^{(1)}, ..., z^{(m)}\}$ from noise prior p_z.

Update G by descending its stochastic gradient:

$$\nabla_{\theta_G} \frac{1}{m} \sum_{i=0}^{m} -log\big[D(G(z^{(i)}))\big]$$

end for

Classifier training (2^{nd} step):

for number of training iterations **do**

Sample minibatch of m positive labeled examples $\{x_P^{(1)}, ..., x_P^{(m)}\}$ from data distribution p_P.

Sample minibatch of m noise samples $\{z^{(1)}, ..., z^{(m)}\}$ from data distribution p_z.

Update C by descending its stochastic gradient:

$$\nabla_{\theta_C} \frac{1}{2 \cdot m} \sum_{i=1}^{m} \Big[l(C(x_P^{(i)}), 1) + l(C(G(z^{(i)})), 0) \Big]$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We use Adam in our experiments.

cases, D will consider some unlabeled positive examples as negative ones. As a consequence, this implies that G will also generate some examples following a subset of the positive samples distribution. Thus, the D-GAN will tend to behave as the PGAN [26], which seems to be the best solution in this situation.

The next section presents effective regularization techniques to overcome these issues in the context of the proposed GAN-based PU framework.

4.3.4 Discriminator regularizations

Nowadays, Batch Normalization (BN) [77] is considered as a one of the most relevant regularization techniques commonly used in deep neural networks architectures. Its utility for GANs training has been highlighted in [136] for the DCGAN architecture in order to stabilize the adversarial training. Other variants like the Wasserstein-GAN [1] or the Loss-Sensitive GAN [135] confirmed its interest. As developed in [77], BN addresses issues like vanishing or exploding gradient problems, as well as the risk of getting stuck in a poor

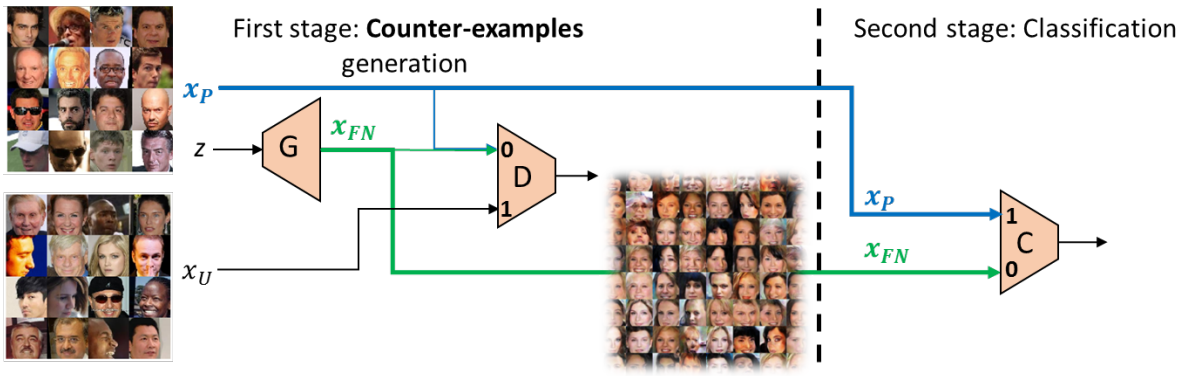


Figure 4.2: Proposed GAN-based PU approach, x_{FN} represents the generated samples which are similar to real negative samples x_N, G is the generative model, D is the discriminator, C is the classifier used to perform the binary Positive-Negative (PN) classification.

local minima, by reducing the internal covariate shift problem of the learning model. A higher learning rate can be used and it can significantly improve the training speed.

Multiple minibatch manipulation incompatibility. BN regularizes the model, in such a way that a training example (i.e. single instance) from a given minibatch sample is considered in conjunction with other examples of this minibatch sample. It is the consequence of estimating the mean and variance normalization parameters one time per minibatch, and then applying them on each example in the minibatch. When positive examples x_P and unlabeled examples x_U are not in the same training minibatch, as this is the case in our discriminator loss function, this does not allow to link labeled positive examples with the unlabeled positive ones. Consequently, this cannot produce a distance between positive and negative examples predictions. To counter this problem, we could imagine to apply BN on a unified minibatch which contains a fraction of each distribution x_P, x_U and x_F. But the BN effect is greatly influenced by the content of the minibatch on which it is applied. Therefore, the fraction π_P of positive examples included in x_U will negatively impact the BN outcome.

Compatible normalization techniques: However, BN benefits in a more traditional training are not negligible. Hence, we propose to use two alternative techniques in order to replace the BN role in the proposed GAN-based PU framework. On the one hand, **Layer Normalization** (LN) [2] is a frequently used technique with sequential networks, as it can be applied for each sequential example independently. With LN, the normalization for a given example is computed on its resulting output feature map layers, and the mean and variance are computed independently for each example of a minibatch. On the other hand, **Spectral Normalization** (SN) [120] is a recent competing technique for GANs [120] training which can stabilize the training of D against input perturbations [50] by perfoming a weight normalization. In this way, a training manipulating multiple types of minibatch

distributions preserves SN effectiveness. For these reasons, we propose to apply LN or SN instead of BN inside our discriminative model structure. The use of these normalization techniques will be validated in Sec. 5.3.

Dropout alleviates the positive overfitting problem: As mentioned in the previous section, we can only deduce Eq. (4.4) if we consider that the positive samples distribution is the same for both labeled and unlabeled ones. In practice, this assumption holds in the case of a large dataset, such that this overfitting problem concerning the positive examples disappears. While some model averaging strategies such as bootstrap aggregating techniques have been previously combined with Support-Vector Machines (SVMs) in order to deal with PU learning [123], the dropout [158] generalization technique is also a solution concerning the deep neural networks. Consequently, in the context of the proposed D-GAN training, we propose to introduce dropout in the top fully connected layer of D. We enable it during D training steps, and conversely disable it during G training steps. This improves the evaluation of generated samples which is transmitted from D to G by back-propagation. In the next section, we will show that dropout alleviates the positive examples overfitting during long D-GAN trainings. This insures to exclusively generate counter-examples.

The next section presents experimental results demonstrating the usefulness of the proposed approach.

4.4 Experimental Results

In this section, we assess the performance of the proposed approach. We first experimentally validate the expected discriminator prediction behaviour when it is applied on a positive unlabeled dataset (Sec. 4.4.2.1), and study the impact of regularization (Sec. 4.4.2.2). Then, we show the ability of the generator to generate counter-examples for different types of PU datasets, including two-dimensional points and natural RGB images (Sec. 4.4.2.3). Finally, we evaluate the proposed model prediction robustness and compare it with state-of-the-art PU learning methods in terms of prior noise (Sec. 4.4.3.1) and first-stage overfitting (Sec. 4.4.3.3).

4.4.1 Settings

We detail in this section the settings of the experiments. We have adapted the first-stage discriminator and generator architectures of the proposed GAN based PU framework depending on the dataset on which they are applied, as follows:

- **2D point dataset**: In order to deal with 2D point datasets, we use a GAN architecture composed of fully connected layers (FullyConnected). The generator and discriminator architectures are summarized in Figure 4.3.
- **MNIST** [96]: In order to deal with grayscale images of dimension 28*28 pixels from the MNIST dataset, we use a deep convolutional GAN architecture (DCGAN) such that the generator contains transposed convolutional (DeConv2D) top layers, and the discriminator contains convolutional (Conv2D) bottom layers as illustrated in Figures 4.4 (a) and (b).
- **CIFAR-10** [92]: In order to deal with RGB images of size 32*32 pixels from the CIFAR-10 dataset, we use the same DCGAN architecture presented in Figures 4.4 (a) and (b). We only adapt the feature maps size depending on the width (w), the height (h), and the number of channels (ch) of input RGB images.
- **celebA** [107]: In order to deal with RGB images of size 64*64 from the celebA dataset, we use a deeper convolutional GAN architecture presented in Figure 4.5.

Input: $z \in \mathbb{R}^8 \sim \mathcal{N}(0,1)$
FullyConnected (128) eLU
FullyConnected (2) Sigmoid
Output: 2D point $x \in \mathbb{R}^2$

(a) Generator

Input: Image $x \in \mathbb{R}^2$
FullyConnected (128) eLU
FullyConnected (1) Sigmoid
Output: Scalar $y \in (0,1)$

(b) Discriminator

Figure 4.3: Fully connected GAN model architecture used for two dimensional points datasets. Minibatch size 64, optimizer Adam. We trained the model during 100 epochs on 2D point datasets.

Concerning the PU dataset initialization from a standard PN dataset, in all the experiments, except the ones in Sec. 4.4.3.1, we use the methodology proposed in chapter 3. More specifically, we set $\rho = 0.5$ which is the fraction of positive labeled examples of the initial PN dataset that we unlabel such that they are included into the unlabeled dataset. Then, we set π_P which is the fraction that represents these unlabeled positive examples among the unlabeled dataset. This method is interesting for testing an approach depending on π_P, independantly of the selected fraction $1 - \rho$ of positive labeled samples.

4.4.2 Qualitative analysis

We start by studying qualitatively whether the discriminator behaves as expected in practice. More precisely, we need to verify that it exclusively associates the counter-examples distri-

Input: $z \in \mathbb{R}^{100} \sim \mathcal{N}(0,1)$
FullyConnected (1024) BN ReLU
FullyConnected ($128 \times \frac{h}{4} \times \frac{w}{4}$) BN ReLU
DeConv2D (64 filters 4×4) BN ReLU
DeConv2D (ch filters 4×4) Sigmoid
Output: Image $x \in \mathbb{R}^{h \times w \times ch}$

(a) Generator

Input: Image $x \in \mathbb{R}^{h \times w \times ch}$
Conv2D (64 filters 4×4) SN LeakyReLU
Conv2D (128 filters 4×4) SN LeakyReLU
FullyConnected (1024) SN LeakyReLU
Dropout (0.5) FullyConnected (1) Sigmoid
Output: Scalar $y \in (0,1)$

(b) Discriminator

Input: Image $x \in \mathbb{R}^{h \times w \times ch}$
Conv2D (32 filters 5×5) ReLU Maxpooling (2×2)
Conv2D (64 filters 5×5) ReLU Maxpooling (2×2)
FullyConnected (1024) ReLU
Dropout (0.5) FullyConnected (2) Softmax
Output: One hot vector $y \in (0,1)^2$

(b) Classifier

Figure 4.4: Convolutional GAN model architecture used for 28*28 grayscale MNIST and 32*32 RGB CIFAR-10 image datasets. For MNIST we set h=28, w=28, ch=1. For CIFAR-10 we set h=32, w=32, ch=3. Minibatch size: 64, optimizer: Adam, strides of 2×2 for the generator Deconv2D and the discriminator Conv2D layers, strides of 1×1 for the classifier Conv2D layers. We trained the model during 40 epochs and 1000 epochs respectively on MNIST and CIFAR-10 datasets.

Input: $z \in \mathbb{R}^{100} \sim \mathcal{N}(0,1)$
FullyConnected (512) BN ReLU
DeConv2D (256 filters 5×5) BN ReLU
DeConv2D (128 filters 5×5) BN ReLU
DeConv2D (64 filters 5×5) BN ReLU
DeConv2D (3 filters 5×5) tanh
Output: Image $x \in \mathbb{R}^{64 \times 64 \times 3}$

(a) Generator

Input: Image $x \in \mathbb{R}^{64 \times 64 \times 3}$
Conv2D (64 filters 5×5) SN LeakyReLU
Conv2D (128 filters 5×5) SN LeakyReLU
Conv2D (256 filters 5×5) SN LeakyReLU
Conv2D (512 filters 5×5) SN LeakyReLU
Dropout (0.5) FullyConnected (1) LeakyReLU
Output: Scalar $y \in (0,1)$

(b) Discriminator

Figure 4.5: Convolutional GAN model architecture used for 64*64 RGB images of celebA dataset. Minibatch size: 64, optimizer: Adam, 2D stride of 2×2. We trained the model during 100 epochs on the celebA dataset.

bution with the label value 1, and the positive samples distribution with an intermediate label value between 0 and 1/2.

In Sec. 4.4.2.1, we start by showing the relation between the PU loss function and the proposed equivalent PN loss function including a biased label for positive examples, as mentioned in Sec. 4.3.2. Then, in Sec. 4.4.2.2, we investigate which regularization techniques

enable to preserve the same behaviour on an image dataset such that the discriminator does not suffer from overfitting during the epoch training iterations.

4.4.2.1 Empirical Positive Unlabeled risk analysis

We have previously demonstrated (Eq. 4.6) that we can reformulate the discriminator PU training loss function R_{PU} into a PN training loss function, referred to as R_{PN}, by replacing the two opposite labels 0 and 1 associated to positive samples distribution p_P by an intermediate label value δ depending on π_P, such that we obtain:

$$R_{PU}(D) = R_{PN}(D), \tag{4.15}$$

with:

$$\begin{cases} R_{PU}(D) = & \mathbb{E}_{x_U \sim \boldsymbol{p_U}}[H(D(x_U), \mathbf{1})] + \mathbb{E}_{x_P \sim \boldsymbol{p_P}}[H(D(x_P), \mathbf{0})], \\ R_{PN}(D) = & \mathbb{E}_{x_N \sim \boldsymbol{p_N}}[(1-\pi_P)H(D(x_N), \mathbf{1})] + \mathbb{E}_{x_P \sim \boldsymbol{p_P}}[(1+\pi_P)H(D(x_P), \boldsymbol{\delta})]. \end{cases} \tag{4.16}$$

It turns out that we can verify the same relation empirically. As illustrated in Figure 4.6 with 2D point samples following gaussian distributions, if we train the discriminator D with a multilayer perceptron structure using the PU loss function R_{PU}, then its predictions outputs for an unlabeled batch sample are partitioned in the vicinity of two different labels. Positive examples are centered around an intermediate label value corresponding to δ. Conversely, D output predictions for the negative examples are centered around the label value 1. In addition, we have also computed the approximated PN risk $\hat{R}_{PN}$ using negative labeled and positive labeled samples, for several δ values between 0 and 1. We can observe that the global minimum of the PN approximated risk $\hat{R}_{PN}$ as a function of δ corresponds graphically to the global maximum of the density function corresponding to D output predictions for a positive set. This coincides also with the equality presented in Eq (4.15).

To sum up, this illustrates experimentally that if D is trained with the R_{PU} loss function, then it should predict the label value 1 exclusively for the negative samples, which is the necessary condition to guide the generator during the adversarial training to learn exclusively the counter-examples distribution.

However, this behaviour is only possible if D does not overfit labeled and unlabeled positive samples. In other words, D should be able to discriminate unlabeled positive examples from the unlabeled negative ones. Therefore, in order to generalize the proposed GAN framework to image datasets, we compare in the next section some state-of-the-art regularization techniques commonly used in deep learning models, in order to select the most appropriate one.

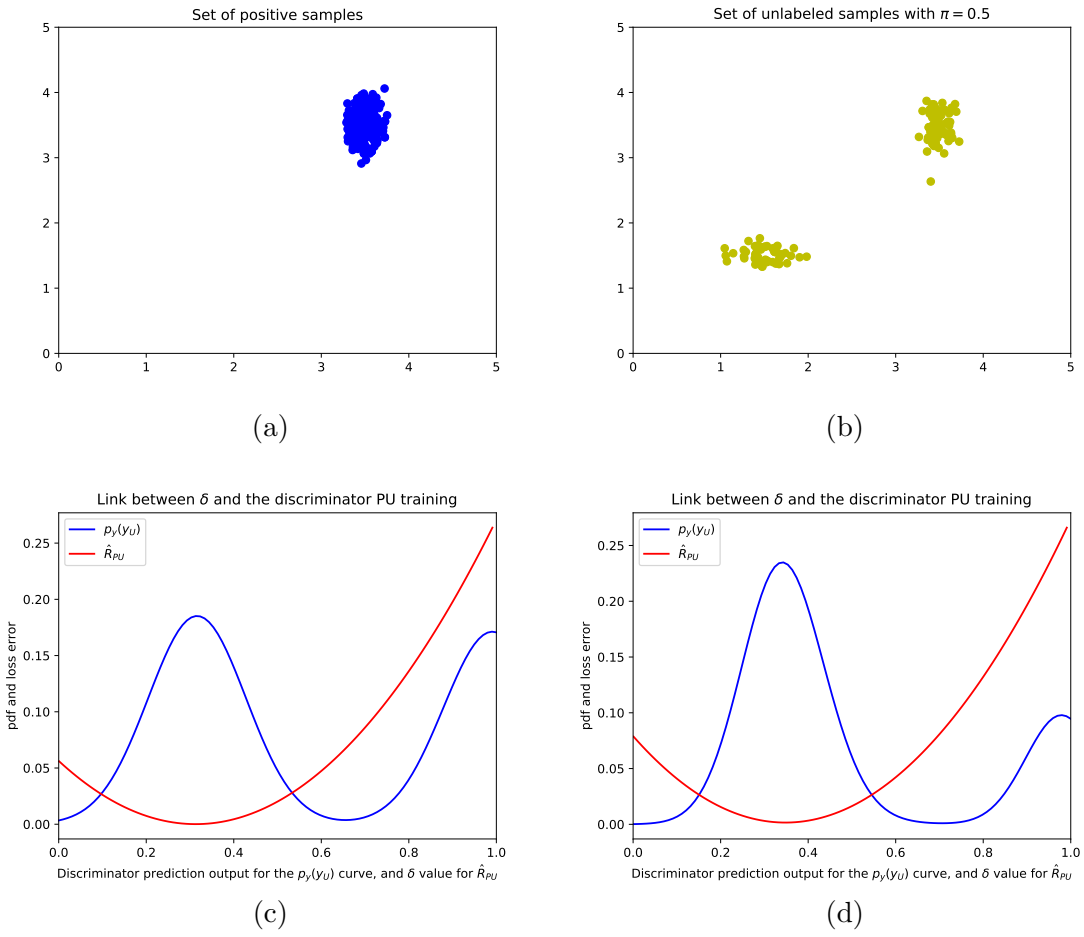


Figure 4.6: Link between the PN loss function suggested (Eq. 4.15) and the distribution of the discriminator output predictions for an input training minibatch. For this experiment, D is a multi-layer perceptron. D has been trained to distinguish a 2D gaussian distribution to another one by using the risk R_{PU} on a PU dataset. (a) Shows a set of 2D points considered as positive samples. (b) Shows a set of 2D points considered as unlabeled samples. Both curves in (c) and (d) have been normalized to get a better visualization. For (c), $p_Y(y_U)$ (in blue), with $y_U = D(x_U)$, represents the probability distribution of D predicted outputs for a minibatch of unlabeled samples, with $\pi_P = 0.5$. $\hat{R}_{PU}(D)$ (in red) represents the PN risk computed in function of δ with the R_{PN} proposed Equation 4.16 on a minibatch of positive and negative labeled samples, once D is trained with R_{PU} risk (Eq. 4.2). (d) shows the same curves as in (c) but by giving in input a concatenation of an unlabeled minibatch with a positive labeled minibatch. Unlabeled positive and labeled positive samples provide a unified prediction output distribution.

4.4.2.2 Impact of regularizations on the discriminator

We compare in Figure 4.7 the ability of D to distinguish positive from negative samples distributions included inside the unlabeled training dataset when D is trained on a PU image

dataset without normalization and with BN, LN, and SN normalizations. We also consider the cases when they are combined with the dropout regularization. In this experiment, D is trained alone such that it is not adversarially trained with G. This enables to better observe and anticipate the adversarial behaviour of D, and consequently the behaviour of G during the adversarial training.

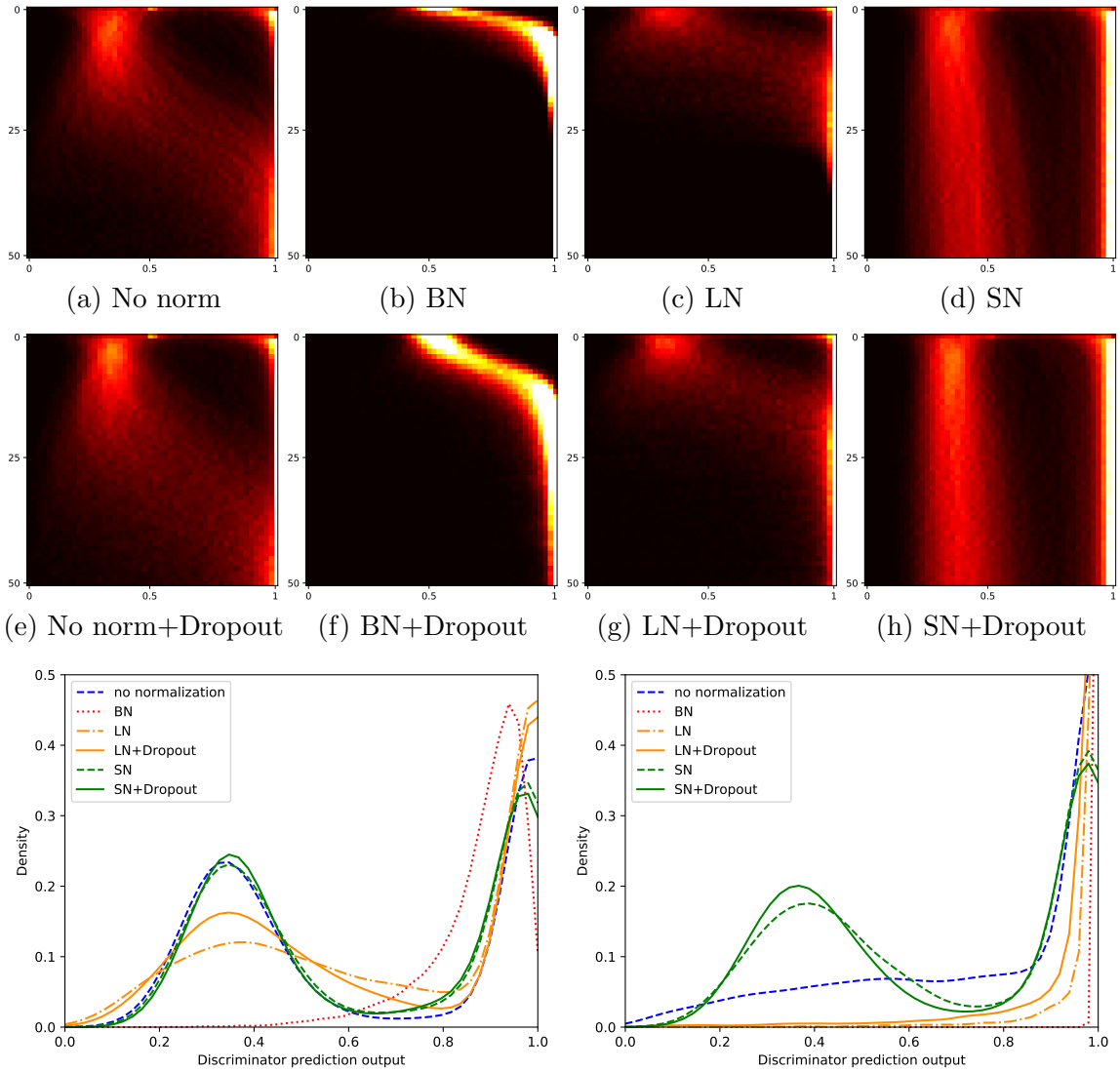
We show in Figure 4.7 the histograms of D predictions concerning the unabeled training examples. As previously explained in Sec. 4.3.2, if D associates exclusively the label 1 with the distribution p_N, then we can observe a mixture of two distributions in the corresponding histograms. The one on the right corresponds to D predictions for unlabeled negative examples. The second one on the left corresponds to D predictions for unlabeled positive examples. It is shifted away from the label 1 and centered around δ. Both distributions cannot be observed with BN. With LN, we can observe both distributions at the beginning of the training before the appearance of an overfitting problem for the unlabeled positive examples. Consequently, at the end of the training, both distributions have merged as with BN. In contrast, SN considerably decreases this overfitting problem. Moreover, the addition of the dropout further helps, such that the dispersion of D predictions is attenuated. This confirms that BN is not compatible with the proposed framework. LN can be used for relatively short trainings. And we conclude that the combination $SN + Dropout$ is the best solution to preserve the distinction between p_P and p_N for long trainings. This is consistent with the arguments discussed in Sec. 4.3.4.

Now that we have validated the discriminator ability to separate positive and negative distributions from a positive unlabeled dataset, we select the most appropriate regularization techniques SN and dropout to train adversarially the discriminator and the generator hereafter. The proposed GAN based PU model ability to generate relevant counter-examples is assessed in the next section.

4.4.2.3 Generating counter-examples

From a qualitative point of view, and contrary to the PGAN model, the proposed D-GAN paradigm generates items which only follow the counter-examples distribution for diverse data types. This is illustrated in Figure 4.8 for 2D point datasets and in Figure 4.9 for image datasets.

In Figure 4.8, we can observe on the top line that the generated sample exclusively follows the distribution of the counter-examples included in the unlabeled set (i.e. simultaneously not positive and unlabeled). On the bottom line, we can observe that the generator has learned the distribution of confident complements of the positive sample distribution over the uniform distribution of unlabeled sample. In addition, we can also observe that a small



(a) No norm (b) BN (c) LN (d) SN

(e) No norm+Dropout (f) BN+Dropout (g) LN+Dropout (h) SN+Dropout

(i) Slice after 5 epochs (j) Slice after 25 epochs

Figure 4.7: D predictions on unlabeled training examples. (a), (b), (c), (d), (e), (f), (g), (h) images show the evolutions of the histograms of predictions during the training of D. Each horizontal line of pixels represents the histogram of predictions, between 0 and 1 along the horizontal axis, of D on the entire unlabeled training dataset. Clear hot colors represent a high density of prediction. The vertical axis indicates the training iterations from 0 to 50 epochs. Figures (i) and (j) represent the corresponding histograms of predictions after 5 and 25 epochs. Settings are with positive class 8 and negative class 3 of MNIST dataset, with $\pi_P = 0.5$.

area around the positive sample distribution is not captured by the generator. This shows the ability of the proposed generative model to not overfit the positive sample distribution boundary.

In Figure 4.9, we can also observe that the generated examples systematically follow the

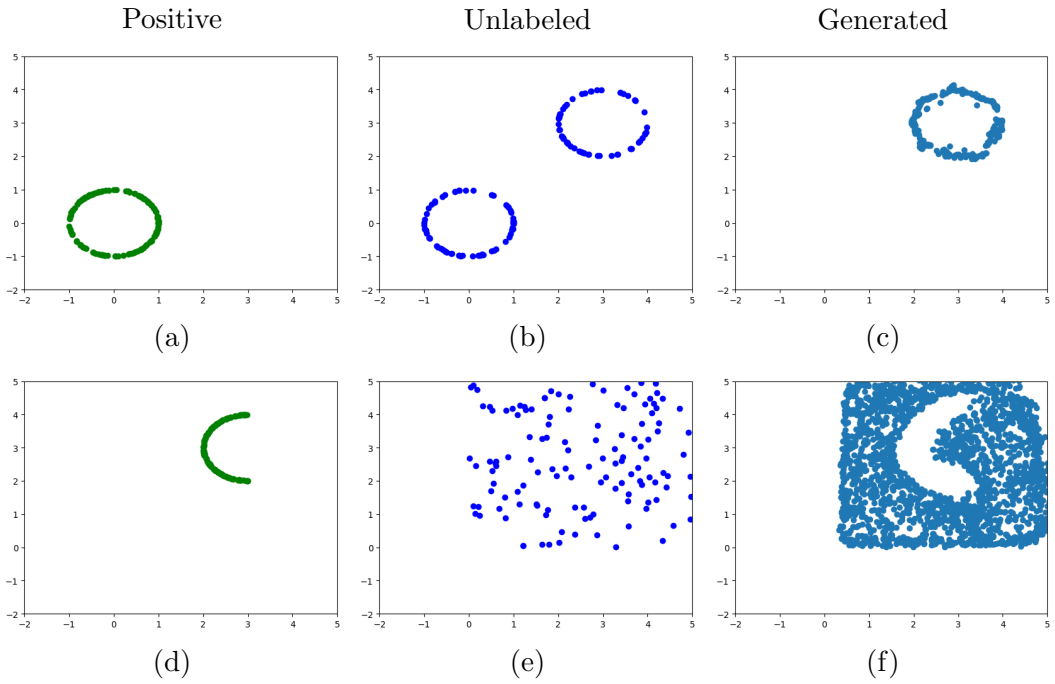


Figure 4.8: Proposed approach applied to two different clusters of 2D points. D and G have a multilayer-perceptron structure with respectively 128 hidden units. From left to right, figures are respectively labeled positive, unlabeled with $\pi_P = 0.5$, and generated samples. Figures (a), (b), (c) case corresponds to distributions following circle shapes. Figures (d), (e), (f) case corresponds to a half circle distribution of positive examples, and a uniform distribution over a defined interval for unlabeled examples.

counter-examples distribution on three image datasets: MNIST, CIFAR-10 and celebA.

In order to enable reproducibility, a D-GAN implementation corresponding to Figure 4.10 results is applied on the LS-GAN model [135]. Our code also includes the method proposed by [26] to establish a PU training dataset from a fully labeled dataset with parameters ρ and π_P.

Morevover, as mentioned previously, the regularization technique used in the discriminator has a direct impact on the samples generated by the generator. Figure 4.10 shows samples generated by G depending on the normalization technique used in D. We can observe that in the first row, with $\pi_P = 0.3$, we naturally obtain around thirty percent of men faces generated using any normalization techniques with the orginial GAN framework used in PGAN. The generated images quality seems visually equivalent between BN, LN or SN. As previously explained, in the second row, also with $\pi_P = 0.3$, the proposed D-GAN approach is not compatible with BN. On the contrary, with LN, it exclusively generates counter-examples: women faces with only few men patterns like facial hairs. Finally, it exclusively generates women faces with SN. Those results are consistent with Sec. 4.3.4 and 4.4.2.2.

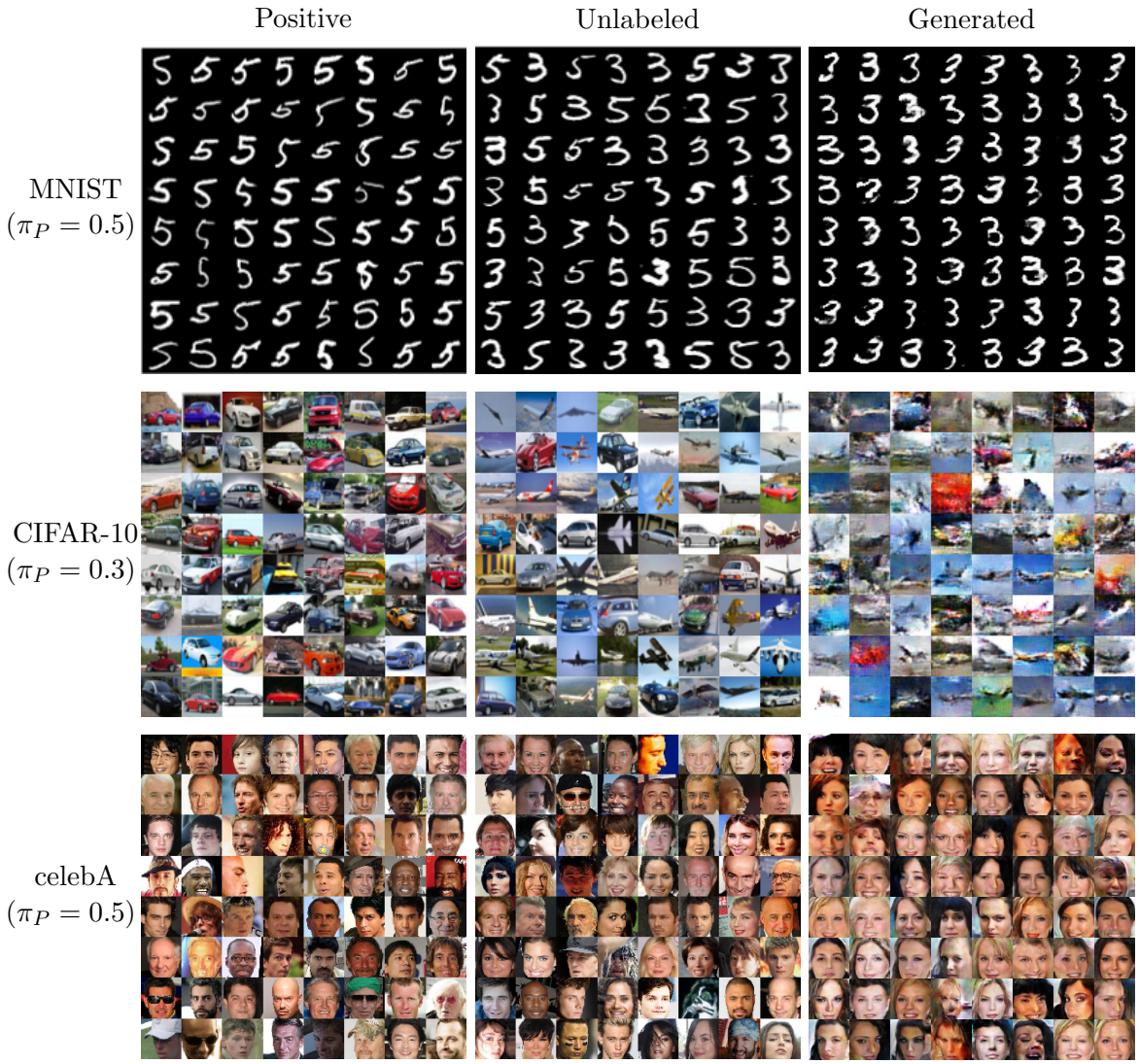


Figure 4.9: Counter-examples generation from Positive Unlabeled image datasets. The two left columns present input positive and unlabeled training samples x_P and x_U. The right column presents output generated minibatch samples x_G. The first row presents results for MNIST classification task 5-*vs*-3 when $\pi_P = 0.5$. The second row presents results for CIFAR-10 classification task *Car-vs-Airplane* when $\pi_P = 0.3$. The third row presents results for the arbitrary celebA classification task *Male-vs-Female* when $\pi_P = 0.5$. Visually, all generated examples observed follow the counter-examples distribution included in the unlabeled training set.

The D-GAN trained with $\pi_P = 0.5$ and BN naturally generates around fifty percents of men faces, as we recall that BN does not enable to capture the counter-examples distribution. The D-GAN also performs relatively well with SN+Dropout when $\pi_P = 0.5$. It exclusively generates women faces. This confirms that the generator behaviour is highly dependent on the discriminator generalization ability, which in turn depends on normalization techniques

used. This also confirms that the proposed D-GAN framework presents the interesting ability to exclusively hallucinate counter-examples on a real PU image dataset when it is combined with appropriate discriminator regularizations.

Figure 4.10: Discriminator regularizations impacts on the generated samples from a PU celebA image dataset after 100 training epochs iterations. The three columns correspond respectively to training experiments with BN, LN, and SN normalization techniques. The first row presents samples generated using the original LS-GAN discriminator loss function. The two bottom rows present the samples generated by integrating the proposed model discriminator loss function term $\mathbb{E}_{x_P \sim p_P}[MSE(D(x_P), 0)]$ in the original LS-GAN loss function, with MSE the mean squared error metric.

We have shown in this section, from a qualitative point of the view, the discriminator ability to separate positive and negative distributions from a positive unlabeled dataset, and the generator ability to learn the counter-examples distribution on various datasets during

the first stage. Next, we propose in Sec. 4.4.3 to quantitatively evaluate the proposed model through an empirical study by focusing on the second-stage classifier C output predictions.

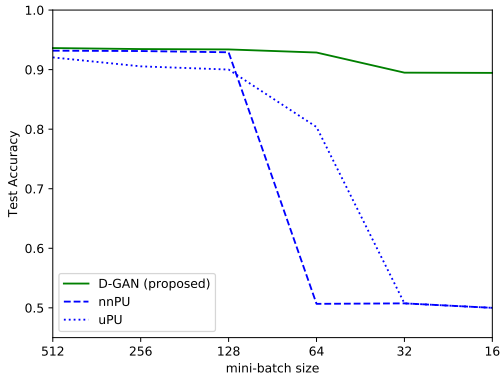
4.4.3 Divergent-GAN for Positive Unlabeled learning

In this section, we evaluate empirically our method on standard PU learning tasks such that we can test its ability to address respective issues of the state-of-the-art methods presented in Section 4.2.

Concerning these comparative experiments, we use the DCGAN [136] architecture.

4.4.3.1 Robustness to prior noise

Nowadays, the *stochastic gradient descent* (SGD) method remains a useful deep learning regularization technique for large-scale machine learning problems [14]. SGD provides a regularizing effect by using minibatches [165]. However, a smaller batch size implies a higher prior noise per batch. Thus, in this section, we empirically study the proposed model robustness to prior noise using small batch sizes.



(a) Curves

Even-vs-Odd (MNIST)		**D-GAN**	nnPU	uPU
Without prior		✓	×	×
minibatch size	$std(\pi_P) \cdot 10^2$	Test Accuracy		
512	2.22	**0.936**	0.932	0.921
256	3.2	**0.935**	0.931	0.906
128	4.31	**0.934**	0.929	0.9
64	6.51	**0.929**	0.507	0.804
32	8.62	**0.895**	0.508	0.508
16	13.06	**0.907**	0.5	0.5

(b) Detailed scores

Figure 4.11: Prediction test Accuracy on MNIST for the *Even-vs-Odd* classification task, as a function of the minibatch size. We choose the prior value $\pi_P = 0.5$, as the standard deviation of the real prior per minibatch is the highest in this way (see Fig. 4.1). This eases to observe the prior sensitivity. We reproduce the experiment *exp-mnist* proposed by nnPU. The PU dataset contains one thousand positive labeled examples, which are *even digits*. The unlabeled set is composed of the entire initial dataset, thus including also the positive labeled ones. $std(\pi_P)$ is the standard deviation of the prior per minibatch. uPU and nnPU results have been obtained with the code provided by the authors of the nnPU work. (b) details the prediction scores used to plot the curves in (a).

We reproduce the *Even-vs-Odd* experiment proposed by [91] as a function of the batch training size. It consists of learning to discriminate *even* digits *0, 2, 4, 6, 8* from *odd* digits *1,*

3, 5, 7, 9. Concerning the second-stage classifier, we use the *multilayer perceptron* architecture provided in [91] [3]. We only replace the bottom fully connected layer of the classifier by a convolutional layer, similarly to the generator top layer and discriminator bottom layer in the DCGAN [136] structure that we use. This avoids compatibility problems between the generator top convolutional layer output and the bottom classifier layer input. Unwanted artifacts in output of GANs MLP structure are slightly different from unwanted artifacts observed in output of GANs convolutional structures. It turns out that PU approaches using prior such as uPU, nnPU and GenPU make the assumption that the global training dataset prior π_P is fixed and known. But in the same PU context, when the minibatch size decreases, the dispersion of π_P per minibatch consequently increases. Figure 4.11 (a) shows that using small batch training sizes causes critical prediction performances collapse issues for unbiased techniques like nnPU and uPU. On the other hand, our proposed approach without using prior knowledge is drastically less sensitive to this problem: While nnPU and uPU methods become ineffective in terms of test Accuracy (i.e. Accuracy score around 0.5), the D-GAN still provides a prediction test Accuracy of 0.907 for training minibatches of size 16 in D, G and C to address the *Even-vs-Odd* MNIST superclass classification task, as detailed in Figure 4.11 (b). We can conclude that the D-GAN outperforms nnPU and uPU in terms of prediction performances such that it can use minibatches to take advantage of SGD. This capacity is also interesting for incremental learning requirements where only small sample sizes may be managed at each new training iteration. Moreover, recent studies show that it is possible to continually train GANs models [98].

The next section compares the proposed approach with GenPU in the context of few positive labeled examples.

4.4.3.2 PU learning with few positive labeled examples

Table. 4.2 compares the D-GAN Accuracy prediction to GenPU for the One-vs-One task with few positive labeled examples [4]. The D-GAN presents the best test Accuracy for 100 labeled positive examples after 40 first-stage training epochs iterations. In the meantime, this confirms that GenPU remains an interesting choice for fewer labeled positive examples, as it also generates fake labeled positive examples during the first stage. It is also interesting to observe that the D-GAN still globally outperforms nnPU and uPU methods without using prior knowledge.

[3] The code is available at: `https://github.com/kiryor/nnPUlearning`.

[4] It would have been interesting to evaluate GenPU performances as well within some other comparative experiments, but it turns out that all information concerning hyper-parameters of GenPU training loss functions are lacking in the authors article [74]. This does not enable us to fairly conduct additional comparative experiments.

Table 4.2: Comparative results of Accuracy prediction performances on MNIST in a One-vs-One mode. GenPU, uPU and nnPU results are reported from the GenPU article. n_P is the number of positive labeled examples and n_U is the number of unlabeled examples which mixes all the rest of positive and negative examples.

One-vs-One	'3'-vs-'5'				'8'-vs-'3'			
Dataset: MNIST	**D-GAN**	GenPU	nnPU	uPU	**D-GAN**	GenPU	nnPU	uPU
Without prior	✓	×	×	×	✓	×	×	×
n_P=100 : n_U=9900	**0.987**	0.983	0.969	0.914	**0.989**	0.982	0.974	0.932
n_P=50 : n_U=9950	0.964	**0.982**	0.966	0.854	0.974	**0.979**	0.965	0.873

Now that we have shown that the proposed model is robust to prior noise, we continue the comparative tests with the methods which do not use prior knowledge π_P in their training cost-functions to address the PU learning task.

4.4.3.3 One-versus-Rest challenge

We compare in this section the proposed approach with the PGAN and RP methods that we consider as baselines for the PU learning task without prior knowledge. More specifically, we evaluate them on the challenging One-vs-Rest task which consists of trying to distinguish a class from all the other ones. This task is interesting for binary image classification applications where the labeling effort may be exclusively done on the class of interest, the positive class. Another motivation is that One-vs-Rest binary classification brings the tools for multiclass classification, as presented in [156].

Tables 4.3 and 4.4 show average predictions for the One-vs-Rest task over MNIST and CIFAR-10 datasets. We use the F1-Score metric for its relevance in such information retrieval and binary classification tasks as highlighted by [103]: the F1-score measures the positive examples retrieval. The PU datasets are simulated as proposed in PGAN chapter 3 such that we can evaluate the results as a function of several π_P fractions. Concerning the second-stage classifier in these experiments, we have used the convolutional architecture presented in Figure 4.4 (c).

First, the Table 4.3 confirms the regularization effects previously observed in Figure. 4.7. Indeed, SN+dropout is an interesting combination for relatively long trainings, as this is the case on the CIFAR-10 dataset on which we have adversarially trained D during 1000 epochs.

One-vs-Rest	AVG_{MNIST}		$AVG_{\text{CIFAR-10}}$	
π_p	LN	SN+dropout	LN	SN+dropout
0.1	0.99	0.99	0.75	**0.82**
0.3	0.98	0.98	0.73	**0.79**
0.5	0.97	0.97	**0.75**	**0.75**
0.7	0.92	**0.94**	0.71	**0.72**

Table 4.3: Regularization methods comparison through the One-vs-Rest task without prior.

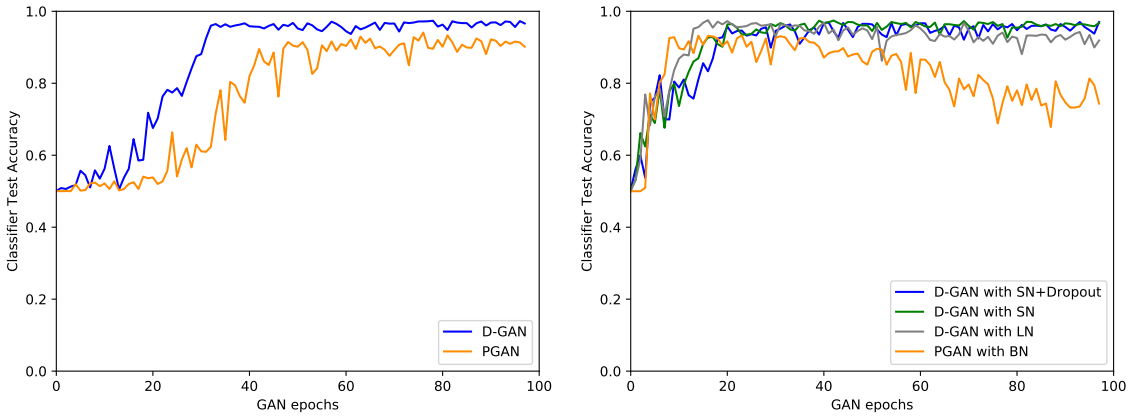
Second, we can observe on Table. 4.4 that the D-GAN, using SN+Dropout regularizations, globally outperforms PGAN and RP methods in terms of F1-Score on both MNIST and CIFAR-10 datasets. Moreover, PNGAN represents GAN-based methods reference for the ideal case where $\pi_P = 0$, such that we train during the first stage a GAN exclusively over all the initial cleanly labeled counter-examples set. PNGAN results highlight the GAN-based methods data augmentation advantage on complex datasets. This justifies the superior scores obtained by our method compared to RP over the CIFAR-10 dataset.

One-vs-Rest	AVG_{MNIST}					$AVG_{\text{CIFAR-10}}$				
π_P	PN	PNGAN	**D-GAN**	PGAN	RP	PN	PNGAN	**D-GAN**	PGAN	RP
0.1	0.993	0.988	**0.989** (0.01)	0.965 (0.01)	0.967 (0.02)	0.680	0.812	**0.815** (0.05)	0.745 (0.08)	0.622 (0.10)
0.3	0.993	0.988	**0.983** (0.01)	0.958 (0.01)	0.975 (0.02)	0.680	0.812	**0.792** (0.05)	0.760 (0.03)	0.730 (0.07)
0.5	0.993	0.988	**0.971** (0.01)	0.946 (0.02)	0.951 (0.04)	0.680	0.812	**0.751** (0.04)	0.748 (0.03)	0.716 (0.06)
0.7	0.993	0.988	**0.938** (0.02)	0.875 (0.05)	0.933 (0.07)	0.680	0.812	**0.721** (0.04)	0.702 (0.03)	0.684 (0.08)

Table 4.4: One-vs-Rest task with two-stage **PU methods without prior**, as proposed in PGAN [26]: From a fully labeled PN dataset, we firstly select a fraction ρ of positive labeled examples that we put in the simulated unlabeled set. Then, we add negative labeled examples in the latter to obtain up to a fraction π_P of positive examples in this unlabeled set. Compared to nnPU simulation method, this simulation method has the advantage to simultaneously and independently control the number of positive labeled examples to keep, and the fraction π_P for the unlabeled set to simulate. *PNGAN* expression represents GAN-based methods reference for the ideal case where $\pi_P = 0$, such that we train during the first stage a GAN exclusively over all the initial cleanly labeled counter-examples set. For each dataset and depending on the fraction π_P, we have tested respectively the ten One-vs-Rest task possibilities and display the corresponding average test F1-score predictions. The standard deviation is indicated in parenthesis.

Reducing the overfitting problem: In addition, we can observe that the proposed model also outperforms PGAN on MNIST with a significant margin. This is due to the fact that, compared to the PGAN which is trained to generate unlabeled examples, the proposed approach only generates counter-examples as previously shown in Figures 4.8 and 4.9. Consequently, the proposed first-stage generative model does not learn the positive samples distribution, and it avoids the PGAN first-stage overfitting issue on simple datasets like MNIST. Figure 4.12 illustrates this phenomenon. In Figure 4.12 (a), without normalization, the D-GAN method gets faster a better Accuracy than PGAN when both are trained under the same conditions. In Figure 4.12 (b), the D-GAN with LN, SN or SN+dropout follows the learning speed of the PGAN with BN, while demonstrating a steadier behaviour once the Accuracy progression is finished, as it overcomes the PGAN first-stage overfitting problem.

To sum up, in Sec. 4.4.2, we demonstrate that the proposed approach is effective at capturing and delivering the counter-examples distribution of our class of interest from only positive and unlabeled data, without using the prior information π_P. In addition, comparative experiments in Sec. 4.4.3 have subsequently highlighted the proposed model



(a) (b)

Figure 4.12: Second-stage Classifier (architecture presented in Figure 4.4 (c)) test Accuracy evolution as a function of the first-stage GAN epochs. 8-vs-Rest MNIST task, with $\rho = 0.5$ and $\pi_P = 0.5$. (a) D-GAN and PGAN are trained without normalization layers. (b) D-GAN and PGAN are respectively trained with LN, SN, SN + dropout, and BN inside the discriminator.

ability to address state-of-the-art PU learning issues such as prior sensitivity and first-stage overfitting. It turns out that addressing simultaneously thoses issues fosters the proposed approach to outperform PU state-of-the-art methods in terms of prediction scores without using prior on both simple and complex image datasets.

4.5 Conclusion

In this chapter, we proposed to incorporate a constrained PU risk into the GAN discriminator loss function in order to deal with PU learning. In this way, the proposed model generates relevant counter-examples from a PU dataset. It outperforms state-of-the-art PU learning methods by addressing their respective issues. Namely, it addresses the prior knowledge dependence of cost-sensitive PU methods and the lack of generalization of selective processes. Moreover, it reduces the PGAN first-stage overfitting, while keeping the minimalist standard GAN architecture, such that it is easily adaptable to recent GANs variants. A side contribution of this work is to have identified discriminator normalizations effects appearing when one manipulates multiple minibatches distributions to deal with a PU training dataset.

Although being competitive with the state-of-the-art, the proposed approach cannot deal with a noisy labeled positive set. A solution to overcome the issue is to extend the proposed framework for censoring PU learning by drawing on existing asymmetric noisy labeled learning techniques [23], as presented in the next chapter 5.

Chapter 5

Noisy labeled learning using GANs

Contents

The previous two chapters deal with positive and unlabeled data such that the positive set contains only positive labeled examples, and the unlabeled set contains an unknown fraction of negative examples supplemented with a fraction of positive examples. As discussed in [124], the unlabeled set can be considered as a corrupted negative labeled set which contains a fraction of correctly labeled negative examples supplemented with a fraction of mislabeled positive examples. Furthermore, keeping in mind this equivalence, if the set labeled as positive contains some mislabeled examples, then the resulting noisy PU dataset can be considered as a noisy labeled positive negative dataset that includes mislabled positive and mislabled negative instances. It turns out that some noisy labeled learning methods deal with such training datasets containing corrupted labels. However, prediction performances of existing methods on small-scale datasets still leave room for improvements. With this objective, and knowing that GANs are effective for data augmentation, we propose to address in this chapter the following question:

- *Can Generative Adversarial Networks be effectively applied to deal with small-scale noisy labeled datasets ?*

In order to answer this question, we present in this chapter a GAN-based method, referred to as Noisy Labeled GAN (NL-GAN), able to generate a clean augmented training dataset from a small and noisy labeled dataset. The proposed approach combines binary asymmetric noisy labeled learning principles with GAN state-of-the-art techniques. We demonstrate the usefulness of the proposed approach through an empirical study on simple and complex tiny image datasets.

5.1 Introduction

Nowadays, the lack of clean labels remains an issue in many image classification applications. As a consequence, we need to tackle the problem of handling noisy labeled data. In the context of binary classification, a noisy labeled dataset contains examples of the positive and negative classes, however, a fraction of the labels are flipped. Noisy labeled learning methods [124], [167], [78] target this issue.

5.1.1 Related work

5.1.1.1 Learning from noisy labels

Some approaches can deal with class-conditional noise as in [124] and [160]; the probability for a label to be corrupted depends on the initial ground truth value of this label. The class-conditional noise-tolerance is proven theoretically in [124], for biased SVM and weighted logistic regression when they use a weigthed loss function satisfying a symmetric condition. In this way, an unbiased estimator can be obtained. However, as empirically observed in the previous chapter, such unbiased approaches suffer from a high sensitivity to the class-conditional noise prior knowledge estimation π_P and π_N as they are used in the unbiased loss function defined as follows:

$$\tilde{l}(t,y) = \frac{(1-p_{-y})\cdot l(t,y) - p_y \cdot l(t,-y)}{1-p_{+1}-p_{-1}}, \tag{5.1}$$

with l the surrogate loss function, such that $E_{\tilde{y}}[\tilde{l}(t,\tilde{y})] = l(t,y)$ for any $t \in (-1,1)$, $y \in \{-1,1\}$.

Noisy labels with neural networks: The interest for using deep neural network (DNN) classifiers for dealing with noisy labels has been introduced and empirically demonstrated in [160]. More specifically, a DNN classifier is propoosed which incoporates a top noisy layer enabling to correctly train the given classifier from noisy labels. Note that this work in [160] deals with labels obtained automatically using tags from social web site or

keywords from image search engines. This motivation is relatively close to ours, as we aim at dealing with noisy labels automatically generated using self-supervised learning techniques in the context of autonomous driving perception, as previously discussed in chapter 2.

Stopping the training before the overfitting of outliers (i.e. noisy labeled instances): Transversely, it has been proposed in [112] to deal with multiclass noisy labels. The approach adapts the training loss function of a CNN classifier depending on the dimensionality of the latent feature map subspaces. Indeed, this dimensionality has been empirically demonstrated to be in correlation with the overfitting of the corrupted labels. First, the dimensionality decreases during the underfitting generalization. Then, it increases during the overfitting. The dimensionality measure used is referred to as Local Intrinsic Dimensionaly (LID) and has been previously introduced in [75]. An advantage of this approach is that it is complementary to other NL related approaches in the sense that avoiding overfitting is a particularly critical issue if one desires to train neural networks on noisy labels. However this approach has only been tested with symmetric noise (i.e. independent of the class) and with relatively small fractions of noise. This approach alone is thus not appropriate for high fractions of asymmetric noise.

It turns out that Rank Pruning (RP) is a state-of-the-art solution, able to deal with potentially high and asymmetric fractions of corrupted labels in binary classification [127]. It consists of first iteratively identifying confident positive and negative examples. During the second step, it trains a classifier with identified examples by considering them as correctly labeled. However, small and complex noisy labeled datasets remain challenging.

Weakly supervised learning using GANs: Recently, some GAN-based approaches [26], [74] have demonstrated state-of-the-art prediction performances to overcome similar issues on partially labeled datasets. In particular, GANs are compelling for learning the representation of sub-distributions and for data augmentation on small and complex datasets.

RP and GAN-based approaches consist of preparing a clean Positive-Negative (PN) dataset from the input noisy one. Consequently they are referred to as two-stage methods.

5.1.1.2 GANs

We recall that the original GAN [64] is an unsupervised generative model. It contains a classifier model, often called discriminator D, and a generator G. D is trained to distinguish real samples x_R from generated samples $G(z)$, with z an input random vector following a uniform or normal distribution p_z. Adversarially, G is trained to generate examples which are considered as real as possible by D. In this way, the generated examples distribution converges towards the real examples distribution p_R. This two-player game can be formalized

with the following minimax value function:

$$\begin{aligned}\min_G \max_D V(G,D) =& \mathbb{E}_{x_R \sim p_R}[-H(D(x_R),1)] \\ &+ \mathbb{E}_{z \sim p_z}[-H(D(G(z)),0)],\end{aligned} \tag{5.2}$$

with H the binary cross entropy metric. Moreover, the GAN litterature provides nowadays effective techniques to overcome the original mode collapse issue [152] and to improve the hallucinated examples quality [85]. The DCGAN [136] method stabilizes the original GAN for image datasets by using convolutional layers and batch normalization (BN) [77]. The spectral GAN [120] increases the examples quality by replacing BN with the spectral normalization (SN). Even more recently, the SAGAN [172] has incorporated attention layers to take into consideration spatial features correlations.

5.1.2 Contributions

To sum up, on the one hand the noisy labeled learning methods can manage noisy labeled datasets. On the other hand, GAN-based approaches have demonstrated their effectiveness for the partially labeled learning task on small and complex datasets. For these reasons, we propose a novel GAN-based approach to tackle the noisy labeled learning task on small and complex datasets. The main contributions of this work consist of:

- incorporating a noisy labeled risk inside the GAN discriminator loss function;
- carefully applying regularization techniques during the GAN adversarial training. This addresses GAN mode collapse and discriminator overfitting issues;
- exploiting prior knowledge of the corrupted labels fractions in order to estimate the most appropriate adversarial training labels.

The outline of the chapter is as follow. Section 5.2 presents the proposed approach. Section 5.3 presents the experimental results. Then, the chapter ends by a conclusion.

5.2 Proposed Method

The insight of the proposed approach is to train two generators to generate examples which are considered by the discriminator as the most positive, respectively most negative, with the highest confidence as possible. To correctly guide the generators, we first identify the discriminator prediction behaviour when it is trained on a noisy labeled dataset.

5.2.1 Problem statement

We start by describing the noisy labeled dataset. The positive and negative samples x_P and x_N follow distributions p_P and p_N respectively. The noisy labeled training dataset is composed of partially corrupted positive and negative samples $x_{\hat{P}}$ and $x_{\hat{N}}$ with the distributions $p_{\hat{P}}$ and $p_{\hat{N}}$ respectively. These latter are mixtures of distributions of p_P and p_N such that $p_{\hat{P}} = \pi_P p_P + (1 - \pi_P)p_N$ and $p_{\hat{N}} = \pi_N p_N + (1 - \pi_N)p_P$. π_P is the fraction of correctly labeled (not corrupted) positive examples, and π_N is the fraction of correctly labeled (not corrupted) negative examples. Finally, we make the assumption that $(\pi_P + \pi_N) \in (1, 2)$, such that the majority of labels are not corrupted. In other words, we always have $(1 - \pi_P) + (1 - \pi_N) \in (0, 1)$ such that less than fifty percents of the initial ground truth labels are mislabeled.

5.2.2 Noisy labeled training

We train the discriminator D to predict the label value 0 for corrupted positive samples $x_{\hat{P}}$ and the label value 1 for corrupted negative samples $x_{\hat{N}}$ such that the corresponding training loss function L_{Noisy} is defined as

$$\begin{aligned} L_{Noisy}(D) =& \mathbb{E}_{x_{\hat{P}} \sim \boldsymbol{p_{\hat{P}}}}[H(D(x_{\hat{P}}), \mathbf{0})] \\ &+ \mathbb{E}_{x_{\hat{N}} \sim \boldsymbol{p_{\hat{N}}}}[H(D(x_{\hat{N}}), \mathbf{1})]. \end{aligned} \tag{5.3}$$

As we use binary cross entropy H metric in the training loss function, we recall that (as demonstrated in appendix A)

$$\alpha \cdot \mathbb{E}_{x \sim p_x}[H(x, 0)] + \beta \cdot \mathbb{E}_{x \sim p_x}[H(x, 1)] = (\alpha + \beta) \cdot \mathbb{E}_{x \sim p_x}[H(D(x_N), \frac{\beta}{\alpha + \beta})]. \tag{5.4}$$

with $(\alpha + \beta) \in (1, 2)$. Thus considering the composition of corrupted distributions $p_{\hat{P}}$ and $p_{\hat{N}}$ depending on π_P and π_N, Eq. (5.3) can be developed as

$$\begin{aligned} L_{Noisy}(D) =& \mathbb{E}_{x_P \sim p_P}[\pi_P \cdot H(D(x_P), 0) + (1 - \pi_N) \cdot H(D(x_P), 1)] \\ &+ \mathbb{E}_{x_N \sim p_N}[(1 - \pi_P) \cdot H(D(x_N), 0) + \pi_N \cdot H(D(x_N), 1)], \end{aligned} \tag{5.5}$$

and, by taking into account Eq. (5.4), the following resulting biased and cleanly labeled loss function is obtained

$$\begin{aligned} L_{Noisy}(D) =& (\pi_P + (1 - \pi_N))\mathbb{E}_{x_P \sim \boldsymbol{p_P}}[H(D(x_P), \boldsymbol{\delta_P})] \\ &+ ((1 - \pi_P) + \pi_N)\mathbb{E}_{x_N \sim \boldsymbol{p_N}}[H(D(x_N), \boldsymbol{\delta_N})], \end{aligned} \tag{5.6}$$

with $\delta_P = \frac{(1-\pi_N)}{\pi_P + (1-\pi_N)}$ and $\delta_N = \frac{\pi_N}{(1-\pi_P)+\pi_N}$. In practice, if we do not know prior π_P and π_N, we can estimate δ_P and δ_N values with a clustering algorithm such as a Gaussian Mixture Model (GMM) [122]. It is sufficient to apply GMM on the discriminator prediction output for a training batch of noisy labeled examples (see Fig. 5.1).

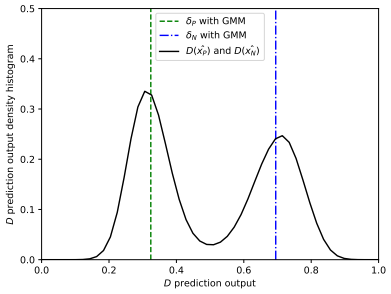


Figure 5.1: Histogram of discriminator output predictions for a training batch including the same proportion of $x_{\hat{P}}$ and $x_{\hat{N}}$ samples. We trained D during 15 epochs on the MNIST dataset with "5" as the positive class, "7" as the negative class, $\pi_P = 0.7$ and $\pi_N = 0.7$. GMM clustering algorithm identifies empirically δ_P and δ_N. This histogram is empirically consistent with the proposed equivalence between Eq. (5.3) and (5.6).

5.2.3 Noisy labeled image classification

Concerning the noisy labeled learning task, we firstly use the proposed GAN-based approach to generate a clean augmented dataset from the noisy labeled one. Fig. 5.2 presents the framework of this first-stage.

5.2.3.1 Generative models step: training loss functions

The proposed GAN-based model contains a discriminator D, a positive generator G_P, and a negative generator G_N. We train G_P to hallucinate fake positive samples x_{GP} and we train G_N to hallucinate fake negative samples x_{GN}. We train G_P and G_N to minimize loss functions L_{GP} and L_{GN}, using labels δ_P and δ_N, as follow

$$\begin{cases} L_{GP}(D, G_P) = \mathbb{E}_{x_{GP} \sim \boldsymbol{p_{GP}}}[H(D(x_{GP}), \boldsymbol{\delta_P})] \\ L_{GN}(D, G_N) = \mathbb{E}_{x_{GN} \sim \boldsymbol{p_{GN}}}[H(D(x_{GN}), \boldsymbol{\delta_N})]. \end{cases} \tag{5.7}$$

Moreover, we train adversarially D with G_P and G_N such that we define D training loss function L_D as

$$\begin{aligned} L_D(D, G_P, G_N) = &\alpha \cdot [\mathbb{E}_{x_{\hat{P}} \sim \boldsymbol{p_{\hat{P}}}}[H(D(x_{\hat{P}}), \mathbf{0})] + \mathbb{E}_{x_{\hat{N}} \sim \boldsymbol{p_{\hat{N}}}}[H(D(x_{\hat{N}}), \mathbf{1})]] \\ &+ \beta \cdot [\mathbb{E}_{x_{GP} \sim \boldsymbol{p_{GP}}}[H(D(x_{GP}), \mathbf{1})] + \mathbb{E}_{x_{GN} \sim \boldsymbol{p_{GN}}}[H(D(x_{GN}), \mathbf{0})]], \end{aligned} \tag{5.8}$$

with α and β the hyper-parameters such that $\alpha >> \beta$. This accentuates the guidelines to train G_P and G_N to converge towards the positive and negative samples distribution. As D, G_P and G_N are deep convolution models, we backpropagate the training errors in their weights with the stochastic gradient descent (SGD) method [14].

Note that in practice, we can replace the binary cross-entropy H by the mean squared error (MSE) metric, while preserving the same training labels.

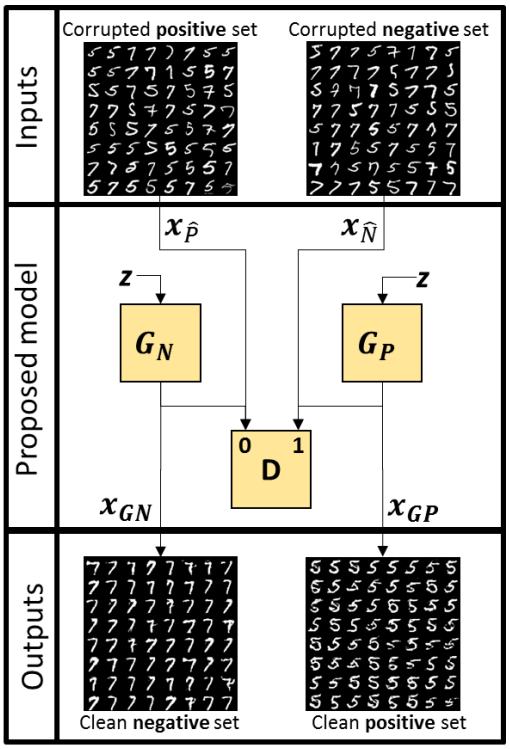


Figure 5.2: Proposed GAN-based label denoising model: This illustrates how to generate a cleanly labeled augmented dataset from a small input noisy labeled dataset. z represents an input random vector following a uniform or normal distribution p_z, such that $x_{GP} = G_P(z)$ and $x_{GN} = G_N(z)$.

5.2.3.2 Posterior step: A standard classification

Concerning the binary noisy labeled classification task, once we have generated a cleanly labeled augmented dataset with the proposed generative model during the first stage, we can train a classifier C with this relevant dataset by considering x_{GP} and x_{GN} samples as respectively real correctly labeled samples x_P and x_N.

From a practical aspect, Algorithm 3 presents the proposed GAN-based two-stage NL framework.

5.2.3.3 Regularizations

In practice, regularization techniques ensure the expected behaviour. We use BN to help the generators training stability and to accelerate their convergence. However, in the discriminator we rather use SN instead of BN. As SN is a weight normalization technique, it is not influenced by the use of four different minibatch samples distributions (see Fig. 5.2). Moreover, we avoid overfitting problems on small datasets by using dropout [158] in the discriminator. More specifically, we activate it during the discriminator training while it is

Algorithm 3 Minibatch SGD training of the NL-GAN

GAN training (1^{st} step)

for number of training iterations **do**

Sample minibatch of m noise samples $\{z^{(1)}, ..., z^{(m)}\}$ from noise prior p_z.

Sample minibatch of m noisy labeled positive examples $\{x_{\hat{P}}^{(1)}, ..., x_{\hat{P}}^{(m)}\}$ from data distribution $p_{\hat{P}}$.

Sample minibatch of m noisy labeled negative examples $\{x_{\hat{N}}^{(1)}, ..., x_{\hat{N}}^{(m)}\}$ from data distribution $p_{\hat{N}}$.

Update D by descending its stochastic gradient:

$$\nabla_{\theta_D} \frac{1}{m} \sum_{i=0}^{m} \Big[-\alpha \big[log[1 - D(x_{\hat{P}}^{(i)})] + logD(x_{\hat{N}}^{(i)}) \big] - \beta \big[logD(G_P(z^{(i)})) + log[1 - D(G_N(z^{(i)}))] \big] \Big]$$

Sample minibatch of m noise samples $\{z^{(1)}, ..., z^{(m)}\}$ from noise prior p_z.

Update G_P by descending its stochastic gradient:

$$\nabla_{\theta_{GP}} \frac{1}{m} \sum_{i=0}^{m} \Big[-\delta_P \cdot logD(G_P(z^{(i)})) - (1 - \delta_P) \cdot log[1 - D(G_P(z^{(i)}))] \Big]$$

Update G_N by descending its stochastic gradient:

$$\nabla_{\theta_{GN}} \frac{1}{m} \sum_{i=0}^{m} \Big[-\delta_N \cdot logD(G_N(z^{(i)})) - (1 - \delta_N) \cdot log[1 - D(G_N(z^{(i)}))] \Big]$$

end for

Classifier training (2^{nd} step):

for number of training iterations **do**

Sample minibatch of m noise samples $\{z^{(1)}, ..., z^{(m)}\}$ from data distribution p_z.

Update C by descending its stochastic gradient:

$$\nabla_{\theta_C} \frac{1}{2 \cdot m} \sum_{i=1}^{m} \Big[l(C(G_P(z^{(i)})), 1) + l(C(G_N(z^{(i)})), 0) \Big]$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We use Adam in our experiments. l is an arbitrary binary classification cost-function which can be for instance binary cross-entropy H or mean squared error MSE.

disabled during the generators trainings.

The next section demonstrates the effectiveness of the proposed approach through an empirical study.

5.3 Experiments

The proposed approach has been tested on small and complex image datasets MNIST [96] and CIFAR-10 [92]. First, we present experimental settings defining the way to simulate a binary classification dataset with noisy labels and the proposed learning model architecture with training hyper-parameters. Then, we present the cleanly labeled samples generated from noisy labeled datasets. Finally, we show that the accuracy prediction performances,

obtained on small and complex highly corrupted datasets, confirm the proposed approach competitiveness.

5.3.1 Settings

In order to test our approach, we first prepared appropriate challenging NL datasets.

5.3.1.1 Noisy labeled dataset simulation

The main goal of this chapter is to test a NL stategy using GANs on small-scale training sets which are highly corrupted. Thus, we simulated the corrupted labels from fully cleanly labeled datasets MNIST and CIFAR-10 depending on prior knowledge parameters π_P and π_N as illustrated in Fig. 5.3. More specifically, we aimed at simulating the following corrupted training sets:

$$\begin{aligned} \hat{P}_{NL-train} &= \{\pi_P \cdot n_{\hat{P}}\ P\ ;\ (1-\pi_P) \cdot n_{\hat{P}}\ N\}, \\ \hat{N}_{NL-train} &= \{\pi_N \cdot n_{\hat{N}}\ N\ ;\ (1-\pi_N) \cdot n_{\hat{N}}\ P\}, \end{aligned} \tag{5.9}$$

where notations $a\ P$ and $b\ N$ respectively refer to a positive and b negative items (i.e. images or examples), and $n_{\hat{P}}$ and $n_{\hat{N}}$ are correspondingly the total number of instances present in the simulated corrupted positive and negative training sets $\hat{P}_{NL-train}$ and $\hat{N}_{NL-train}$. To comply with this requirement for binary symmetric (i.e. $\pi_P = \pi_N$) and asymmetric NL learning (i.e. $\pi_P \neq \pi_N$), we propose to up-sample or down-sample the number of items included in the initial cleanly labeled sets depending on π_P, $(1-\pi_P)$, π_N, $(1-\pi_N)$, $n_{\hat{P}}$, $n_{\hat{N}}$ in such a way that we get:

$$\begin{aligned} P_{init-train\ up-down\ sampled} &= \pi_P \cdot n_{\hat{P}} + (1-\pi_N) \cdot n_{\hat{N}}\ P, \\ N_{init-train\ up-down\ sampled} &= \pi_N \cdot n_{\hat{N}} + (1-\pi_P) \cdot n_{\hat{P}}\ N, \end{aligned} \tag{5.10}$$

with $P_{init-train\ up-down\ sampled}$ and $N_{init-train\ up-down\ sampled}$ the up-sampled or down-sampled initial training datasets. By following the previously presented requirements, empirical comparative experiments are performed with rates of corruption settled up to fourty percents (i.e. $\pi_P = 0.6$) and on small sizes of sets as detailed in Table 5.1. A size of 100 for the MNIST task $\{5; 7-vs-2, 4\}$ means that we only use in average 25 partially corrupted examples for each subclass. Thus, after such drastic dataset reductions, we systematically perform an upsampling such that the training datasets used always have a size of 10000 examples. It introduces redundancy in the training dataset, but it empirically enables to keep the same number of epochs iterations for any dataset reduction. Hyper-parameters and the proposed framework architecture are presented in next section.

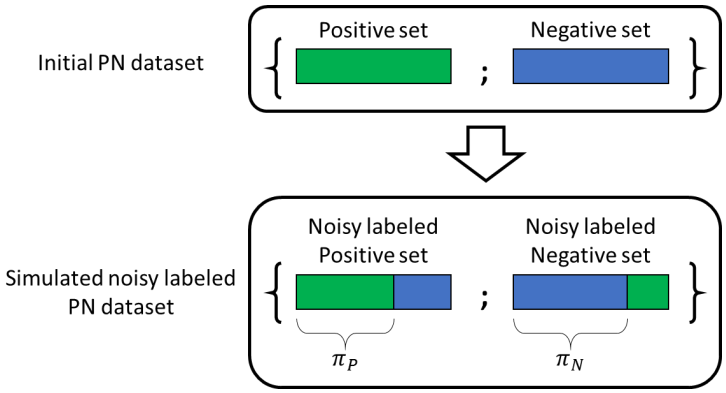


Figure 5.3: Visualization of our simulated NL training data compared to the initial cleanly labeled data. π_P and π_N represent the fraction of not corrupted positive and negative labels present in the NL training dataset.

Table 5.1: Composition of the simulated training sets with noisy labels depending on π_P and π_N, and resulting G_P and G_N training loss function parameters δ_P and δ_N.

NL training sets contents	$\pi_P = 0.85$, $\pi_N = 0.85$	$\pi_P = 0.7$, $\pi_N = 0.85$	$\pi_P = 0.6$, $\pi_N = 0.65$
Size: 1000	$\hat{P}_{NL-train} = \{425\ P\ ;\ 75\ N\}$ $\hat{N}_{NL-train} = \{75\ P\ ;\ 425\ N\}$	$\hat{P}_{NL-train} = \{350\ P\ ;\ 150\ N\}$ $\hat{N}_{NL-train} = \{75\ P\ ;\ 425\ N\}$	$\hat{P}_{NL-train} = \{300\ P\ ;\ 200\ N\}$ $\hat{N}_{NL-train} = \{175\ P\ ;\ 325\ N\}$
Size: 100	$\hat{P}_{NL-train} = \{43\ P\ ;\ 7\ N\}$ $\hat{N}_{NL-train} = \{7\ P\ ;\ 43\ N\}$	$\hat{P}_{NL-train} = \{35\ P\ ;\ 15\ N\}$ $\hat{N}_{NL-train} = \{7\ P\ ;\ 43\ N\}$	$\hat{P}_{NL-train} = \{30\ P\ ;\ 20\ N\}$ $\hat{N}_{NL-train} = \{17\ P\ ;\ 33\ N\}$
G_P and G_N loss function parameters	$\delta_P = 0.15$, $\delta_N = 0.85$	$\delta_P = 0.18$, $\delta_N = 0.74$	$\delta_P = 0.37$, $\delta_N = 0.62$

5.3.1.2 Settings of the proposed GAN-based NL framework

Concerning the loss function L_D, we established empirically $\alpha = 5$ and $\beta = 0.5$, while parameters δ_P and δ_N of generator loss functions L_{GP} and L_{GN} are theoretically inferred depending on prior knowledge π_P and π_N and indicated in Table 5.1. For the corresponding first-stage learning models D, G_P and G_N, we adapted the previous DCGAN [136] architecture to this novel framework as specified in Fig. 5.4. D contains two bottom convolutional layers, followed by two top fully-connected layers. The input convolutional layer contains 64 3*3 filters, the next one has 128 3*3 filters, and the hidden fully connected layer has 1024 filters. G_P and G_N contain symetrically two bottom fully connected layers followed by two deconvolutional layers with the same number of filters. The generators input is a vector z of 100 random values following a uniform distribution. As discussed in the regularization subsection, we use BN on the generators deconvolutional layers. In D, we apply SN on convolutional layers, and dropout of 0.5 in the fully connected hidden layer. To deal with the relatively complex CIFAR-10 image dataset containing 32*32*3 RGB images, we included in D an additional hidden convolutional layer with 256 filters. G_P and G_N consequently include a hidden deconvolutional layer. Concerning the second-stage classifier C, we use

the convolutional structure previously mentioned in [26] [1]. We use the Adam SGD method [88] for all previously enumerated learning models, and a learning rate initialized to $2 \cdot 10^{-4}$ during the first-stage and to $1 \cdot 10^{-4}$ during the classification step. We train D, G_P and G_N adversarially during 40 epochs on MNIST and during 500 epochs on CIFAR-10. Then, we train during 25 epochs the classifier C, as we train RP [2] during 25 epochs.

Input: $z \in \mathbb{R}^{100} \sim \mathcal{N}(0,1)$
FullyConnected (1024) BN ReLU
FullyConnected ($256 \times \frac{h}{8} \times \frac{w}{8}$) BN ReLU
DeConv2D* (128 filters 4×4) BN* ReLU*
DeConv2D (64 filters 4×4) BN ReLU
DeConv2D (ch filters 4×4) Sigmoid
Output: Image $x \in \mathbb{R}^{h \times w \times ch}$

(a) Generators G_P and G_N

Input: Image $x \in \mathbb{R}^{h \times w \times ch}$
Conv2D (64 filters 4×4) SN LeakyReLU
Conv2D (128 filters 4×4) SN LeakyReLU
Conv2D* (256 filters 4×4) SN* LeakyReLU*
FullyConnected (1024) SN LeakyReLU
Dropout (0.5) FullyConnected (1) Sigmoid
Output: Scalar $y \in (0,1)$

(b) Discriminator D

Input: Image $x \in \mathbb{R}^{h \times w \times ch}$
Conv2D (32 filters 5×5) ReLU Maxpooling (2×2)
Conv2D (64 filters 5×5) ReLU Maxpooling (2×2)
FullyConnected (1024) ReLU
Dropout (0.5) FullyConnected (2) Softmax
Output: One hot vector $y \in (0,1)^2$

(b) Classifier C

Figure 5.4: Convolutional GAN model architecture used for 28*28 grayscale MNIST and 32*32 RGB CIFAR-10 image datasets. For MNIST we set h=28, w=28, ch=1. For CIFAR-10 we set h=32, w=32, ch=3. Minibatch size: 64, optimizer: Adam, strides of 2×2 for generators Deconv2D and discriminator Conv2D layers, strides of 1×1 for the classifier Conv2D layers. We trained the model during 40 epochs and 1000 epochs respectively on MNIST and CIFAR-10 datasets. Functions assigned with * are added when dealing with the latter.

5.3.2 Qualitative results

Fig. 5.5 illustrates the images that the proposed approach is able to generate on MNIST and the natural image dataset CIFAR-10. We corrupt up to 40 percents of the training labels. However, despite the fact that the generated examples are cleanly labeled, the hallucinated images quality probably still has the potential to be improved with hyper-parameters fine-tuning study in the context of this novel framework.

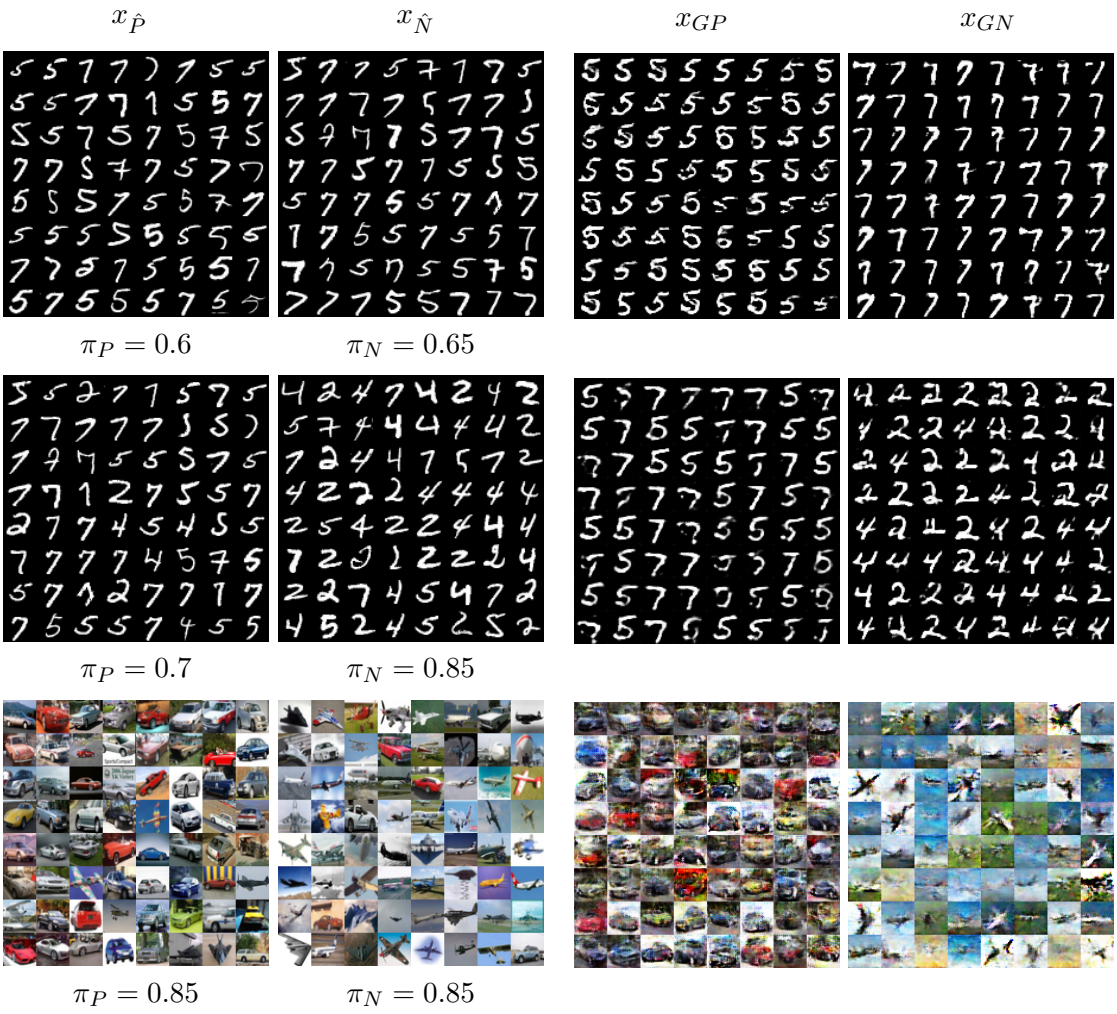


Figure 5.5: Cleanly labeled dataset generation from noisy labeled datasets. The two left columns present noisy labeled minibatch input positive samples $x_{\hat{P}}$ and negative samples $x_{\hat{N}}$. The two right columns present output generated minibatch samples x_{GP} and x_{GN}. The first row presents results for MNIST classification task 5-*vs*-7 when $\pi_P = 0.6$ and $\pi_N = 0.65$. The second row presents results for MNIST classification task $\{5; 7\}$-*vs*-$\{2; 4\}$ when $\pi_P = 0.7$ and $\pi_N = 0.85$. The third row presents results for CIFAR-10 classification task *Car-vs-Airplane* when $\pi_P = 0.85$ and $\pi_N = 0.85$. Visually, every generated samples observed hallucinate cleanly labeled examples.

5.3.3 Comparative results

Table 5.2 presents comparative accuracy prediction performances on small corrupted training datasets. PN baseline reference in Table 5.2 represents a training of the classifier, used during the second stage, on the initial dataset reduced and without corrupted labels, such that

[1] *https : //github.com/tensorflow/tensorflow/blob/master/tensorflow/examples/tutorials/mnist/mnist_softmax.py*

[2] RP code is available at: *https : //github.com/cgnorthcutt/rankpruning*

Table 5.2: Two-stage noisy labeled learning comparative results in terms of test accuracy prediction performances on small noisy labeled image datasets. As we use a SGD optimization method, each result is respectively the average of five identical independent trainings.

Test Accuracy	ref	$\pi_P = 0.85$, $\pi_N = 0.85$		$\pi_P = 0.7$, $\pi_N = 0.85$		$\pi_P = 0.6$, $\pi_N = 0.65$	
$\{5;7\}$-vs-$\{2;4\}_{\text{MNIST}}$	PN	NL-GAN	RP	NL-GAN	RP	NL-GAN	RP
Size: 1000	0.97	0.961	**0.965**	**0.956**	0.948	**0.926**	0.809
Size: 100	0.9	**0.909**	0.892	**0.881**	0.853	**0.814**	0.703
Car-vs-Airplane$_{\text{CIFAR-10}}$	PN	NL-GAN	RP	NL-GAN	RP	NL-GAN	RP
Size: 1000	0.878	**0.843**	0.833	**0.824**	0.794	**0.704**	0.659
Size: 100	0.789	**0.789**	0.761	**0.782**	0.739	**0.687**	0.64

$\pi_P = 1$ and $\pi_N = 1$. The other columns show results for three experiments: $\pi_P = \pi_N = 0.85$, $\pi_P = 0.70$ and $\pi_N = 0.85$, and $\pi_P = 0.60$ and $\pi_N = 0.65$, respectively.

The proposed approach, referred to as Noisy Labeled GAN (NL-GAN), globally outperforms RP method on both MNIST and CIFAR-10 datasets. In particular, the proposed approach becomes especially interesting with high fractions of corrupted training labels. Nonetheless, because of the adversarial training, we recall that 500 first-stage epochs iterations are necessary on CIFAR-10 to get these results while only 40 epochs are necessary on MNIST. Therefore, if we can afford the computational complexity, the proposed approach remains competitive on complex image datasets like CIFAR-10.

5.4 Conclusion

In this chapter, we proposed a novel GAN-based framework to deal with small-scale noisy labeled image datasets. The proposed approach compares favorably with the state-of-the-art in terms of prediction performances. Furthermore, experimental results show that it is possible to generate an augmented cleanly labeled dataset from asymmetric noisy labeled datasets with GANs. Thus, *combining GANs and NL learning is an answer for the lack of clean hand-labeled training data.*

However, we expect that the proposed approach can further be improved for complex images by including recent GAN-based advances [17] as discussed in chapter 7.

The next chapter 6 presents real application adaptations of PU and NL techniques in the context of aerial and urban road image analysis for potentially moving obstacles detection and semantic segmentation from, weakly hand-labeled datasets, or partial automatic labels obtained using hand-crafted techniques playing the role of supervision through a complete self-supervised learning framework.

Chapter 6

Applications

Contents

In this chapter, we propose to apply weakly supervised image classification strategies studied in previous chapters for real image understanding applications. More specifically, this chapter addresses the following questions:

- 1) Is Positive Unlabeled data useful to address an object detection task on aerial images?
- 2) Is Positive Unlabeled data useful to address an urban semantic segmentation task?
- 3) Is it possible to detect, segment, and classify urban moving obstacles without hand-labeled data?

Corresponding answers are respectively presented in next sections in the same order.

6.1 PU learning for vehicle detection on aerial images

This section presents a solution for vehicle detection on aerial images using PU learning. This objective is to drastically reduce the need of hand labeled training data required with Positive Negative (PN) techniques, while providing equivalent or better prediction performances. Although the selected object detection strategy which incorporates the proposed PU technique is not novel, to the best of our knowledge, we are the first to apply PU techniques for an object detection task.

Sec. 6.1.1 introduces the target application and its specificities. Sec. 6.1.2 presents the object detector integrating the PGAN framework (see chapter 3) in order to deal with a PU object detection dataset. Sec. 6.1.3 presents experiments demonstrating the potential of PU learning techniques for vehicle detection from aerial images. Then a conclusion is drawn in Sec. 6.1.4.

6.1.1 Context and Motivation

Image object detection is a common task in the state-of-the-art computer vision literature. It consists of automatically delineating with bounding boxes the target objects in a given image. During the last decade, object detection techniques, such as Faster R-CNN [144] and Yolov3 [143], have appeared, using deep convolutional neural networks for visual feature extraction. They achieve state-of-the-art performances, both in terms of predictions [144] and real-time computational cost [143]. However, such competitive techniques require large-scale labeled datasets. Thus, if one aims at applying an object detector on a novel target domain, then an adapted labeling effort is required which can entail a significant cost.

This section focuses on object detection on aerial images for civil [28] and military applications. More specifically, we focus on vehicle detection [21], [8]. Providing automatic suggestions of detection to an operator has the potential to ease and accelerate his task. In our context, it is required to consider the following restricting conditions:

- Avoid False Negative (FN) predictions in order to not miss potential vehicles of interest.
- The object detector is applied on low-resolution images.
- Altitude of the aerial camera view point is constant.
- Reduce as much as possible hand-labeling task of the operator for the training stage. For instance, the operator may only have to partially label the training data.

The first point can be addressed by enabling the operator to adjust the detection threshold. We address the second point by using deep learning architectures dedicated to dealing

with tiny images. With the third point, we do not need to consider multiscale analysis. Concerning the last point, it turns out that unlabeled data of the target domain are easily available during the offline training step. Thus, we propose to drastically reduce the need of labeled data by only focusing the labeling effort for positive instances and by exploiting unlabeled data instead of counter-examples. In other words, it is equivalent to performing PU learning. Thus, we propose to address this visual object detection task using a classifier trained through the GAN-based PU framework introduced in chapter 3 and referred to as PGAN.

6.1.2 Proposed approach

We propose to adopt the sliding window strategy, commonly used for object detection task, as the object detection task can then be defined as a classification problem. Consequently, we can directly adapt the previously proposed PGAN image classification PU techniques. The sliding window strategy consists of applying a classifier as a patch filter on the given image to analyze. In this way, a prediction score is obtained for several overlapping location in the target image. Then, in order to keep only the most relevant location, we apply Non-Maximum Suppression (NMS) algorithm using bounding box size and coordinates in order to estimate the intersection over union (i.e. interestion area between two given boundox boxes) of existing bounding boxes. Finally, we only keep the non-overlapping bounding boxes with the highest prediction probabilities.

In order to perform a sliding window strategy during the prediction step, we first need to train the classifier to apply on this sliding window.

6.1.2.1 PU classifier training

Concerning the PU training of the classifier, we need a learning technique able to deal with a PU dataset without prior knowledge concerning the unlabeled data. Moreover, the images under consideration have low-resolution, represent natural content, and contain color information. The PGAN technique does not suffer from overfitting on tiny complex RGB images, empirically demonstrated on CIFAR-10 and STL-10 datasets in chapter 3. Therefore, we propose to use the PGAN technique to address this vehicle detection task on aerial images.

In this context, we provide sets of vehicle patch images (i.e. positive examples) and unlabeled patches, which potentially include a high fraction of background counter-examples, as input to the PGAN. This patch dataset preparation is detailed in Sec. 6.1.3.

Then, as PGAN is a two-stage technique, we only keep the second-stage trained classifier during the detection task.

6.1.2.2 Object detector

Concerning the object detector, the framework is relatively simple. As we know that the distance with the ground is constant, all vehicles approximately have the same scale. Hence, we only apply the sliding window classifier on a single scale. The size of the sliding window has been defined statistically using the prior information of the median size of vehicles in terms of pixels from the given altitude of the camera view point. In addition, it is not necessary to have the orientation and width/height of the detected vehicles. Thus, the presented detector does not integrate additional regression predictions to complement the classifier role.

First, we apply the trained classifier on patches extracted with a sliding window. In order to reduce the computational time, we decide to first extract all patches to be classified of the given aerial image to process, and then we apply simultaneously and independantly the classifier on every patch through parallelization on a GPU. Then, we keep the patch coordinates for which the classifier has predicted a positive probability higher than a predefined threshold. This threshold is selected by the operator depending on the desired False Positive rate.

Then, in order to reduce the number of false detections while preserving isolated detections with low confidence, we apply the Non-Maximum-Suppression (NMS) algorithm. In this way, we only keep the best centered patches coordinates. IoU threshold parameter of NMS is also a parameter predefined by the operator.

The next section shows the experimental results of the developed approach.

6.1.3 Experiments

This experiment section is structured as follow. First, we describe the parameter settings concerning the dataset, and learning model architectures. Then, we present PGAN generated images that we consider as counter-examples during the PGAN second-stage. We also illustrate the object detector inputs and outputs. Next, we show an example of the proposed PU detector predictions when it is trained with few positive labeled examples and unlabeled examples. Finally, we present experiments comparing a PN training with a Positive Unlabeled training in terms of classification prediction performances.

6.1.3.1 Settings

Aerial image dataset preparation: We use VEhicle Detection in Aerial Imagery (VEDAI) dataset [140] as it is composed of 1269 labeled aerial images which contain 1378 labeled car vehicles. Images are available with 1024*1024 and 512*512 RGB pixel resolutions. In order to train the classifier, we extract patches of size 48*48*3 for 1024*1024*3 images, and of

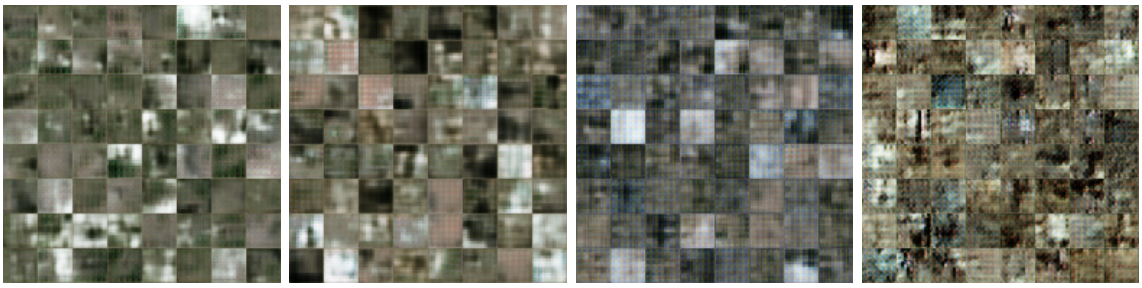
size 24*24*3 for 512*512*3 images. We have defined these patch dimensions by computing the median width and height of ground truth bounding box annotations of car vehicles. In this way, we have extracted patches centered on car vehicles concerning the positive sets. Concerning negative examples, we have randomly extracted up to 9000 patches corresponding to the background. However, for the unlabeled set, we have randomly extracted 18000 patches without considering potential overlaps with annotated positive vehicle patches. The last 400 car vehicles are used for the test evaluation of classifier prediction performances. During the training step, we perform a data augmentation processing including horizontal and vertical patch flipping, translation and rotation transformations. Moreover, we perform upsampling on small-scale datasets in order to have 18000 instances for each training set.

Learning models: Concerning the first-stage generative adversarial network, we have selected the Wasserstein GAN (WGAN) [1] optimization process as it converges relatively faster than the original GAN [64] towards a given target distribution. In order to deal with real natural tiny visual patches, we have used the DCGAN architecture. Concerning the second-stage classifier, we have adapted the ResNet model with 32 layers (ResNet32) to 48*48*3 and 24*24*3 resolutions. This model is an interesting compromise between prediction accuracy and computational cost, which motivated our choice. In these experiments, ResNet32 is trained during 20 epoch iterations. As we deal with sets of small size, this training duration avoids potential overfitting issues. The sliding window has an arbitrary stride of 6 on 1024*1024 images and a stride of 3 on 512*512 images.

Sec. 6.1.3.2 presents visual predictions of the developed GAN-based PU detector.

6.1.3.2 Qualitative results

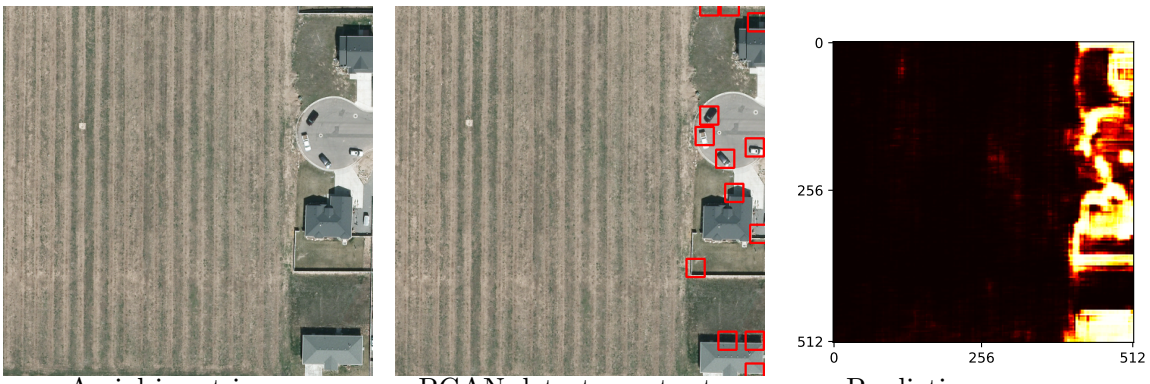
Fig. 6.1 shows the evolution of generated samples appearence during the PGAN first-stage training. We can observe that the quality progressively increases while still exhibiting some visual artifacts. This weak visual quality is maybe due to the fact that the background lacks specific visual patterns in opposition to vehicles. Moreover, the mode collapse issue prevents the GAN to learn to generate vehicles distribution as their proportion in unlabeled data is drastically smaller than background examples. As an advantageous consequence, we can observe that exclusively counter-examples distribution of our class of interest is learned. On another note, as previously discussed in chapters 3 and 5, it is not necessary to generate samples of good visual quality to have interesting prediction performances during the second-stage classifier training. Thus, for reasons of computational time, and despite the fact that the generated samples visual quality still has the potential to be improved, we arbitrarily decided to stop the PGAN first-stage training at 2500 epoch iterations.



50 epochs 100 epochs 1000 epochs 2500 epochs

Figure 6.1: Visualization of generated images during the first-stage training of the PGAN. The model is trained on 18000 unlabeled patches of size 24*24*3 randomly selected in the 512*512*3 aerial images of VEDAI dataset.

Fig. 6.2 presents predictions of the proposed PGAN detector approach when it is trained with few positive examples. Moreover, NMS is used to avoid several detections for a given single instance. Car vehicles are correctly detected, but we can also observe some false positive detections. However, we recall that the number of labels is drastically reduced and that the final application is to propose an assisting tool. Moreover, the prediction map shows that the trained PU classifier is very sensistive to most of the salient regions.



Aerial input image PGAN detector output Prediction map

Figure 6.2: Visualization of PGAN detector predictions using only 100 vehicle patches and unlabeled patches randomly selected. The prediction map is obtained by concatenating predictions outputs of the positive class neuron of the sliding window classifier.

Now that we have shown the predicted outputs of the proposed framework, we propose to evaluate the interest of the proposed model in terms of prediction performances with few labeled examples.

6.1.3.3 Quantitative comparison

We used SGD optimization method which brings prediction variations on small-scale datasets. Each test accuracy result presented in this section is the average of five identical independent trainings for which we have respectively preserved the best test Accuracy.

Before evaluating the PGAN detector, we first compare the prediction performances of the state-of-the-art classifier ResNet32, depending on the resolution and on the size of the labeled PN training dataset. We observe in Table 6.1 equivalent or better prediction performances of ResNet32 classifier with 24*24*3 patches than with higher resolution 48*48*3 patches. Surprisingly, results are globally better with the lowest resolution. This may be due to the fact that the exploited ResNet32 architecture was initially designed for tiny images of CIFAR-10 dataset.

Table 6.1: Comparative results in terms of test accuracy prediction performances on small-scale PN image datasets.

Test Accuracy	Car-vs-Rest$_{\text{VEDAI}}$				
training sets	$\{900\ P; 9000\ N\}$	$\{100\ P; 900\ N\}$	$\{100\ P; 100\ N\}$	$\{50\ P; 50\ N\}$	$\{10\ P; 10\ N\}$
Window size: $48 * 48 * 3$	0.992	0.957	0.883	**0.872**	0.621
Window size: $24 * 24 * 3$	**0.993**	**0.961**	**0.898**	0.869	**0.667**

In order to observe the potential interest of PU techniques for real aerial images analysis, we propose to compare PN and PGAN trainings. In these experiments, both techniques use the same ResNet32 [72] classifier architecture. Concerning the first-stage generative model, it is trained during 2500 epoch iterations. Comparative results are presented in Table 6.2. We can observe that the GAN-based PU technique enables to drastically reduce the number of labeled instances while preserving results competitive with PN trainings. For instance, the PGAN technique using 100 positive examples achieves a similar prediction score as the PN training with ten times fewer labeled examples. Moreover, with very few labeled examples, the PGAN approach significantly outperforms the predictions of the PN training. However, from the operator point of view, a classifier with a prediction accuracy of 0.828 still may not be enough to help during the analysis task.

Table 6.2: Comparative results in terms of test accuracy prediction performances on small-scale PN (i.e. the baseline) and PU image datasets. ResNet32 classifier is directly applied on PN sets while PGAN method is applied on PU training sets. ∞ symbol corresponds in practice to 18000 unlabeled patches.

Test Accuracy	Car-vs-Rest$_{\text{VEDAI}}$							
methods training sets	PN $\{100\ P; 900\ N\}$	PGAN $\{100\ P; \infty\ U\}$	PN $\{100\ P; 100\ N\}$	PGAN $\{100\ P; \infty\ U\}$	PN $\{50\ P; 50\ N\}$	PGAN $\{50\ P; \infty\ U\}$	PN $\{10\ P; 10\ N\}$	PGAN $\{10\ P; \infty\ U\}$
Window size: $24 * 24 * 3$	0.961	**0.962**	0.898	**0.962**	0.869	**0.952**	0.667	**0.828**

To sum up on these experiments, it is possible to adopt a PU detection strategy in order to drastically reduce the need of labeled training data. Nonetheless, a minimum labeling effort is still required with the presented technique in order to provide an exploitable detector.

6.1.4 Conclusion

To conclude, we can answer to the first question of this chapter as follows:

- *Yes, it is possible to apply a PU training dataset for car vehicle detection on aerial images. However, the presented approach needs further improvements to provide more accurate predictions, for instance by applying a regressor on feature maps predicted at the classifier top layers, for predicting the detected vehicles width and height.*

6.2 Positive Unlabeled analysis for semantic segmentation of urban potentially moving obstacles

This section studies the ability of a deep learning segmentation model to deal with a Positive Unlabeled segmentation dataset. In the context of autonomous vehicle perception, we focus on the task of segmenting potentially moving obstacles.

The outline of this section is as follow. Sec. 6.2.1 presents the motivation of this work. Sec. 6.2.2 presents a novel generic and simple strategy for Positive Unlabeled Segmentation task, referred to as *PUseg*. Sec. 6.2.3 presents the corresponding empirical results obtained on a PU segmentation dataset simulated using Cityscapes dataset [32]. Sec. 6.2.4 draws a conclusion.

6.2.1 Context and Motivation

This section first presents the targeted autonomous vehicle perception application and then presents two distinct types of existing solutions: Analytical and learning methods. Then, we introduce a self-supervised learning framework combining advantages of previously mentioned techniques, such that it is able to deal with the lack of hand-labeled data. This framework is based on a PUseg model, which is the focus of this section. Corresponding contributions are then stated before describing the PUseg model in Sec. 6.2.2.

6.2.1.1 Targeted application: Potentially Moving Obstacles Segmentation without hand-labeled data

A target application in autonomous driving is the ability to robustly identify surrounding potentially moving obstacles (i.e. moving obstacles, but also static obstacles which can

potentially move in the near future.) of the ego-vehicle. In the urban scene context, these obstacles are generally other road users, such as vehicles, cyclists, and pedestrians. Correctly identifying them is a critical task for an autonomous vehicle for safety reasons. However, their shapes may be complex and varied.

Analytical techniques (i.e. hand-crafted) enable to correctly highlight the moving ones during the sensor data analysis. Several techniques are mentioned in the overview presented in [94]. These strategies are often based on computing temporal variations of the target visual scene, for example using temporal camera frame sequences, as proposed in [9], using optical flow or in [53] and [182] by performing background subtraction [15] using an EM-based [125] Gaussian Mixture Model (GMM) algorithm [138].

Conversely, some recent deep learning methods, as discussed in surveys [42] and [105], present state-of-the-art prediction performances to detect obstacles [133], even in real-time from a single monocular camera frame [168]. Furthermore, all potentially moving obstacles can be delineated by combining such bounding box detection modules with a semantic segmentation mask [71]. However, this type of approach often needs to be trained with a large number of manually annotated data. As previously discussed in chapter 1, this limitation combined with a potential overfitting on training data, reduces prediction performances of pre-trained state-of-the-art techniques on a specific real world application unlabeled dataset. As a consequence and to the best of our knowledge, state-of-the-art learning techniques still leave the door open to identify all potentially moving obstacles instantaneously at a given frame, without using any training hand-labeled data. It turns out that, from a practical point of view, it can be easy to acquire unlabeled data for urban autonomous driving applications.

6.2.1.2 Motivation

Based on the above analytical and learning strategies, we envision to exploit labeling information automatically extracted from unlabeled camera sensor data with analytical techniques to train a learning model through a partially labeled dataset: Identified moving obstacles represent examples of our class of interest (i.e. the positive class *potentially moving obstacle*), the tip of the iceberg of all potentially moving obstacles. The hidden parts of this iceberg, in other words the remaining static visual patterns, are considered as unlabeled. Indeed, a static region can also be a potentially moving obstacle, which can start to move in the near future. Examples include a car waiting in front of a stop sign, a pedestrian waiting to cross the street, or any of them constrained to remain static on the traversable area due to an accident.

In practice, we propose to capture some moving positive instances using GMM background subtraction from a static point of view to reduce as much as possible the false positive

per-pixel labels. Complementarily, we propose to record unlabeled data from a moving ego-vehicle in order to capture as many unlabeled counter-examples (i.e. everything which is not a potentially moving obstacle) as possible. These two Positive and Unlabeled sets define a Positive Unlabeled segmentation (PUseg) learning challenge. The proposed SSL PU framework combining background subtraction and PU learning for potentially moving obstacle segmentation is summarized through the illustration in Fig. 6.3.

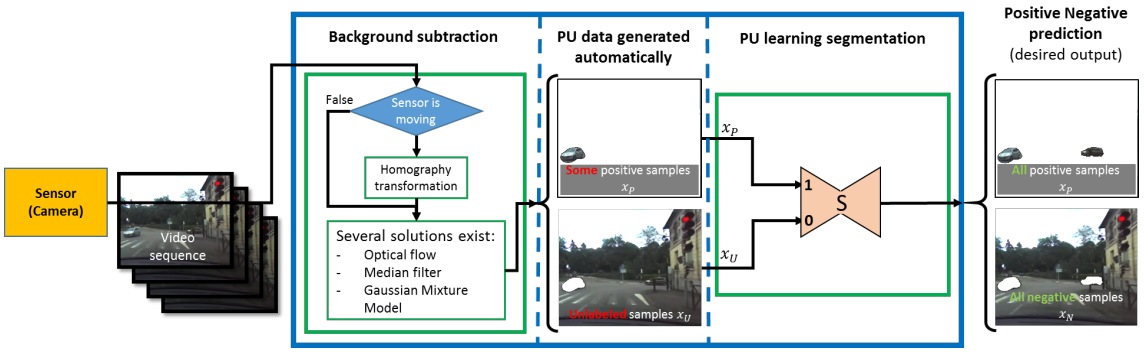


Figure 6.3: Potentially moving obstacles segmentation through a monocular visual learning framework partially self-supervised using temporal information. This potential Self-supervised framework provides us the guidelines to follow to design our PUseg model S to integrate inside.

6.2.1.3 Contribution

In order to deal with the PUseg challenge included in this SSL application framework, the following provides a preliminary study showing a proof of concept for performing a PUseg task. Moreover, this empirical study shows that it is possible to coarsly estimate the fraction of positive examples present in the unlabeled training set. This PUseg study includes the following contributions:

- Study the behaviour of an AE designed for image segmentation when it is trained on a PU dataset.
- Highlight the possibility to coarsly estimate prior knowledge of a sample set of a given PUseg training dataset.

Sec. 6.2.2 presents the proposed PUseg framework.

6.2.2 Proposed PUseg approach

The main idea of the proposed approach consists of adapting PU classification baseline principles to PUseg. This enables to directly consider PUseg as the PU classification problem [7] previously addressed in chapters 3 and 4.

More specifically, we propose to consider every pixel-level predicted label as an independant output of a given classifier. This PUseg problem formalization leads us to the folllowing assumption: If one trains a learning segmentation model with PU labels, then, for a given pixel, we can observe at the corresponding predicted output a histogram of predictions similar to the histogram of a biased PU classifier. Consequently, this pixel-level histogram of predictions should be a mixture of the two following distributions:

- The distribution of negative pixels, centered around the label associated to unlabeled items.
- The distribution of positive pixels, centered around an intermediate label value δ depending on the fraction of unlabeled positive pixels.

Positive pixels and negative pixels, respectively, refer to pixels of the positive and the negative classes. Now that we know the expected behavior of the PUseg model, we need to define the requirements of such a learning model. In particular, we need to choose the appropriate training loss function and the segmentation model architecture to train with.

6.2.2.1 Biased PU training loss function with symmetric properties

The PU training loss function must be able to separate positive from negative pixel examples using PU labels. As discussed in chapter 4, a biased solution without using prior knowledge is to enforce the trained model to associate a unified intermediate label value $\delta \in (0, 1)$, depending on the unknown prior knowledge, with all positive examples (i.e. labeled and unlabeled positive examples), and a distinct label value for unlabeled negative examples. In this PUseg context, the prior knowledge refers to the fraction α of unlabeled positive examples over the total number of positive examples (i.e. the sum of labeled and unlabeled positive examples).

The correct convergence of the learning model must ensure that the empirical intermediate label value predicted for positive examples corresponds to the theoretical argument of the global minimum of the selected biased PU training loss function. The interest is that given an empirical estimation of the global minimum argument equal to δ, one can deduce the prior knowledge α.

Problem statement: We want to associate negative samples x_N following the distribution p_N to the negative class label 0, and the positive samples x_P following a distribution p_P to a different label value. Thus, we propose to associate available unlabeled samples x_U to the label 0 and the labeled positive samples to the label value 1. In terms of proportion in the training PU dataset, let β be the fraction of positive samples and $(1 - \beta)$ be the fraction of negative samples. And let α be the fraction of positive example which are unlabeled as

previously mentioned. We define the global distribution p_{PU} of the PU dataset samples x_{PU} as

$$\begin{aligned} p_{PU} &= \beta \cdot p_P + (1-\beta) \cdot p_N \\ &= \beta\alpha \cdot p_P + \beta(1-\alpha) \cdot p_P + (1-\beta) \cdot p_N \\ &= \beta(1-\alpha) \cdot p_P + p_U, \end{aligned} \tag{6.1}$$

with $p_U = \beta\alpha \cdot p_P + (1-\beta) \cdot p_N$. Now let S be our segmentation model which takes as input pixel samples following the distribution p_{PU}, and which predicts for a given input pixel a value over an interval between 0 and 1. Let L_{PU} be the biased PU training loss function of the proposed PUseg model S that we train to associate the label 1 to labeled positive pixels and the label 0 to unlabeled pixels. Let l be the cost function used in L_{PU} to associate a label to a given sample. Considering the above considerations, we define L_{PU} as

$$L_{PU}(S) = \beta(1-\alpha) \cdot \mathbb{E}_{x_P \sim p_P}[l(S(x_P), 1)] + \mathbb{E}_{x_U \sim \boldsymbol{p_U}}[l(S(x_U), 0)]. \tag{6.2}$$

According to the composition of the distribution p_U and the expectation linearity, we can develop L_{PU} as

$$\begin{aligned} L_{PU}(S) =& \beta \cdot \mathbb{E}_{x_P \sim p_P}\Big[(1-\alpha) \cdot l(S(x_P), 1) + \alpha \cdot l(S(x_P), 0)\Big] \\ &+ (1-\beta) \cdot \mathbb{E}_{x_N \sim \boldsymbol{p_N}}[l(S(x_N), 0)]. \end{aligned} \tag{6.3}$$

Desired loss function properties: The selected loss function l must associate all positive examples following the distribution p_P to a label value different to the label associated to p_N. Thus, we propose the two following desired properties for the cost function l:

- A unified intermediate label value δ can be associated to given samples x_A following the distribution p_A if one asssociates the latter, using l, to two different label value y_B and y_C, depending on the fraction α such that:

$$\begin{aligned} \arg\min_{x_A}(l(x_A, \delta)) &= \arg\min_{x_A}\Big(\alpha \cdot l(x_A, y_B) + (1-\alpha) \cdot l(x_A, y_C)\Big) \\ &= \delta, \end{aligned} \tag{6.4}$$

  with $\delta = \frac{\alpha \cdot y_B + (1-\alpha) \cdot y_C}{\alpha + (1-\alpha)}$.

- l must be symmetric such that its input arguments can be permuted as follows:

$$l(y_1, y_2) = l(y_2, y_1), \tag{6.5}$$

  with y_1 and y_2 the input arguments.

The second condition is due to the empirical observation that with symmetry, the labeled positive examples distribution can be learned at the same rythm that unlabeled positive

examples. This ensures in practice that the learning model predicts on average the δ intermediate label value depending on α for all input pixel positive examples.

Loss function selection: The binary cross-entropy loss function H has been used in the previous chapters for weakly supervised learning tasks. But we empirically observed that it is not the best candidate to estimate prior knowledge during the training due to its asymmetry: Labeled positive examples distribution is not learned at the same rythm that unlabeled positive examples distribution. Consequently, we do not use H hereafter. As developed in Appendix A, it turns out that Mean Squared Error (MSE) is a symmetric loss function with interesting properties for dealing with corrupted training labels. MSE respects both conditions mentioned with eq. (6.4) and eq. (6.5). MSE presents the wished symmetric and global minimum argument caracteristics, while enabling a deep learning model to converge sufficiently fast. Therefore, we propose to use MSE as the cost function l, such that we can formulate L_{PU} as

$$L_{PU}(S) = \beta(1-\alpha) \cdot \mathbb{E}_{x_P \sim p_P}[MSE(S(x_P), 1)] + \mathbb{E}_{x_U \sim \boldsymbol{p_U}}[MSE(S(x_U), 0)], \tag{6.6}$$

with $MSE(y_1, y_2) = (y_1 - y_2)^2$, with y_1 and y_2 the input arguments.

Prior knowledge estimation: Concerning the prior knowledge estimation, as MSE respects the previously mentioned conditions, then considering the labels associated using L_{PU} to labeled and unlabeled examples, one can deduce that

$$\delta = \frac{\alpha \cdot 0 + (1-\alpha) \cdot 1}{1} \Leftrightarrow \alpha = 1 - \delta. \tag{6.7}$$

Thus, prior knowledge α can be estimated by computing the empirical average prediction of the trained PUseg model S for available labeled positive pixels.

Next section presents the selected segmentation model architecture.

6.2.2.2 Segmentation model architecture

AutoEncoder architecture for image segmentation: Several learning model architectures designed for image semantic segmentation have been previously proposed in the literature as detailed in [57]. State-of-the-art techniques generally present an architecture composed of an encoding step. Some convolutional layers of classifier architectures can be used during this step. Then, during the second step, a decoding model takes as input the encoded latent space of a given input image. Inspired by deep learning image classification techniques, this decoder can use convolutional layer combined with upsampling filters, or dilated, transposed convolutional layers. To deal with the semantic segmentation task, a softmax top layer is finally added as output of the decoder to predict a per-pixel semantic classification.

AutoEncoder for weakly supervised learning: It has been previously proposed in [118] to deal with PU image classification using an AutoEncoder. Although this was not a pixelwise semantic segmentation task, this shows the potential of AEs to deal with PU image datasets. On another note, a self-supervised feature learning technique proposed in [129] shows autoencoder segmentation models ability to smooth output predictions despite noisy input labels. Recently, it has been proposed in [73] to deal with noisy labels for biomedical images using state-of-the-art AutoEncoder segmentation architectures as FCN [108], SegNet [3] [86], U-Net [148]. U-Net has been highlighted through this study as the more robust model. This confirms the ability of decoding layers of a segmentation autoencoder to behave similarly to successive convolutional layers of classification models in terms of prediction robustness against label noise.

These knowledge guided our reasoning concerning the capacity to deal with a PUseg training dataset using a segmentation AutoEncoder architecture. Several techniques exist and the current state-of-the-art techniques present high definition and accurate predictions. However, this exploratory study mainly aims at focusing on the ability to deal with PUseg labels. Thus, we want to avoid unexpected prediction phenomenon which are potentially not interpretable using the theoretically defined biased PU training loss L_{PU}. For this reason, we made the choice to use a relatively simple AE segmentation architecture. FCN is a good candidate as it contains only convolutional layers. However, it suffers from a lack of accuracy concerning the boundary details of instances to segment. SegNet contains convolutional and deconvolutional layers combined with skip connections in order to deal with unpooling information loss issue. This improves low frequency prediction details. SegNet uses in addition a Conditional Random Fiels (CRF) post-processing step [159] to obtain its final prediction performances. In turn, U-Net is composed of a convolutional encoder sharing its predicted feature maps with the input feature maps of the deconvolutional decoder, by concatenating them. These shared feature maps play the same role as the skip connections in SegNet.

A more recent segmentation technique appeared called PSPNet [176]. It proposes to deal with multiscale limitation of the previously mentioned techniques. This is done by concatenating feature maps predicted with convolutional layers applied respectively at different scale of a given output encoder feature map. We argue that this novel process is transversal to the biased PU technique loss function effect, as this is the case for CRF technique. However, PSPNet model has a more important training computational cost which is not interesting for the purpose of this study. In addition, we recall that the targeted final application consists of using positive pixel labels automatically generated using a background subtraction technique. It turns out that considering large-scale context information of the

given image to analyze can bring the following issue: The learning model can statistically learn that potentially moving obstacles are always standing on the traversable area dedicated to them. For instance, potential undetected outliers may include parked car drivers, and pedestrians out of sidewalks and pedestrian crossings. This is a potentially critical issue that we want to avoid.

Based on these remarks, we propose to use U-Net architecture during experiments presented in Sec. 6.2.3.

Concerning regularization techniques to prevent the model to overfit mislabeled positive examples, the combination SN+dropout can be used as in previous chapters. However, these two regularization techniques increase the training computational cost of the learning model. Therefore, although we have the conviction that these techniques are beneficial for improving prediction performances, we do not use them on U-Net during experiments in the next section.

6.2.3 Experiments

In this section, we empirically study the predictiong behaviour of U-Net autoencoder segmentation model when it is trained on a PUseg dataset. We first describe the experimental settings concerning the PU dataset preparation and the selected PUseg architecture. Then, we present both qualitative and quantitative results for the proposed PUseg framework, investigating the relationship between the fraction α of unlabeled positive examples, the semantic pixel-level segmentation output predictions, and the learning convergence speed.

6.2.3.1 Experimental settings

Semantic segmentation dataset selection: As the target application is to segment potentially moving obstacles at the pixel-level on monocular images, we decide to study the PUseg model behaviour on Cityscapes dataset [32]. This dataset is interesting as it contains up to 213 241 labeled instances of urban potentially moving obstacles.

PUseg dataset simulation: We first identify the fine ground truth labels of potentially moving obstacle instances. Second, we make a list of these instances. Then we randomly unlabel some of them depending on the probability fraction α that we have defined. In these experiments, we have simulated three distinct PUseg datasets with respectively $\alpha = \{0.3, 0.5, 0.7\}$. This fraction of unlabeled positive examples aims at simulating all static potentially moving obstacles present in camera recorded unlabeled data of a real-world application. For computation cost reasons, the initial Cityscapes dataset images resolution is resized to 128*256 RGB pixels.

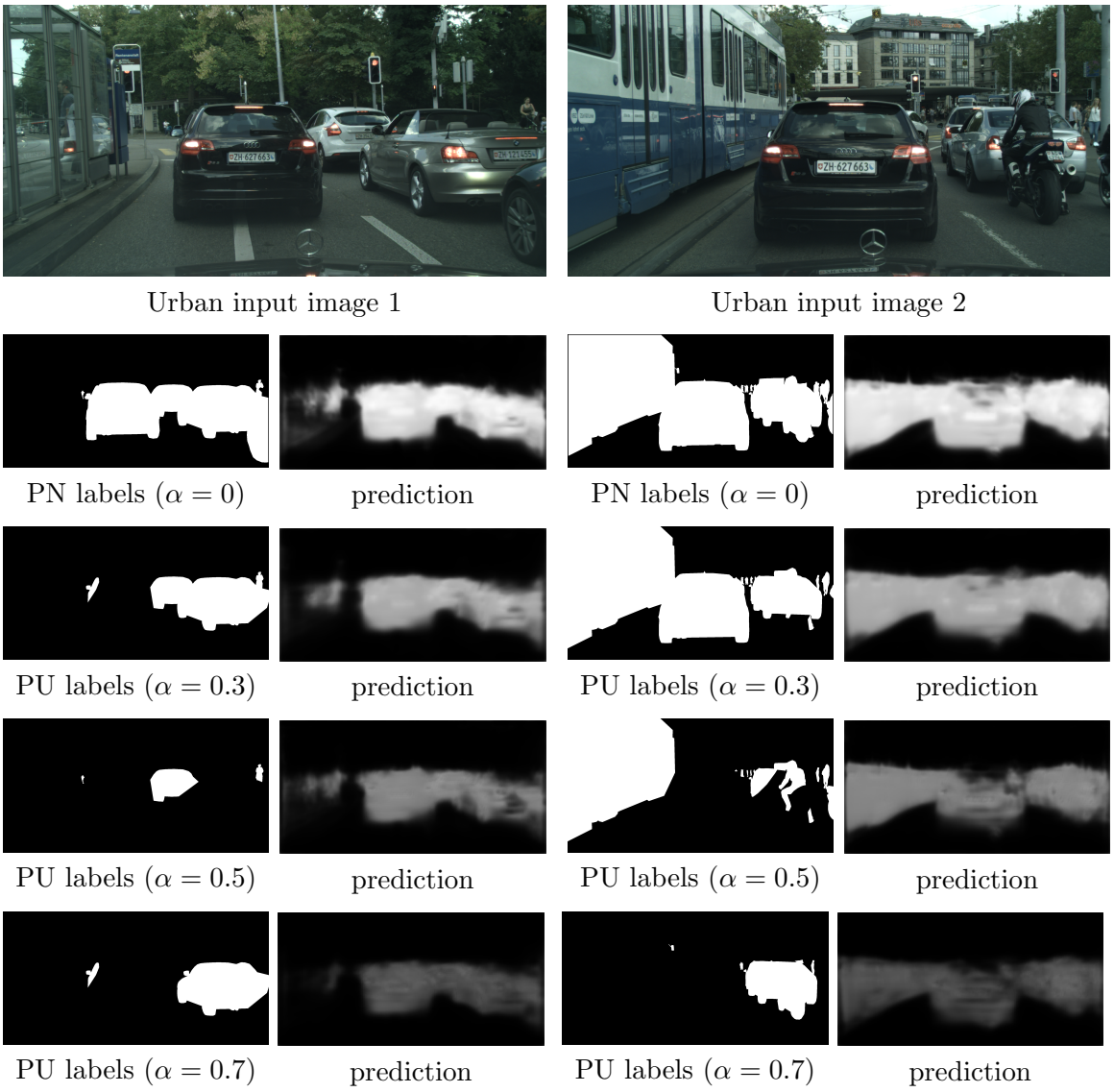


Figure 6.4: Visualization of output binary semantic segmentation predictions of the PUseg model on two training images depending on the corresponding training PU labels fraction α.

PUseg model architecture: As previously motivated in Sec. 6.2.2, we use U-Net AE semantic segmentation model in these experiments. More specifically, we use the original U-Net architecture presented in [148]. The encoder part includes 8 convolutional layers with 3*3 filters combined with intermediate Maxpooling layers with filters of size 2*2. The bottlneck part, working on the encoded feature map latent space, contains two layers with 3*3 convolutional filters. The decoder includes, symmetrically to the encoder, 8 convolutional layers with respectively 3*3 filters combined with 4 intermediate 2*2 upsampling layers. Then, an output convolutional layer is composed of one chanel only as we consider binary segmentation. A second channel is not relevant in this PU context, as we want to interpret

the predicted output rather than combining softmax and argmax output functions. We use the Stochastic Gradient Descent optimization algorithm to train the model. Concerning the training loss function, we use MSE as previously discussed in Sec. 6.2.2.

Evaluation metric: Regarding the evaluation metric, we use Intersection over Union (IoU) metric to evaluate the proposed model ability to classify positive pixels. We perform a One-versus-Rest task, such that the negative class includes everything that is not a potentially moving obstacle.

6.2.3.2 Empirical results

Yes, PUseg learning is possible! Fig. 6.4 presents the predictions of the PUseg model on training images when it is trained with different fractions α. We can observe that when $\alpha = 0.7$, this drastically reduces for a given training image the proportion of training labeled positive pixels. However, despite this severe disadvantage, the PUseg model is still able to segment the central pixel regions of corresponding observable potentially moving obstacle instances. Even more importanly, we can observe that the PUseg model is able to segment similarly labeled positive and unlabeled positive instances as they are predicted with the same pixel color intensity. We can conclude that the PUseg model does not suffer from overfitting concerning the training positive examples initially considered as unlabeled. Otherwise, the output prediction intensity, represented in gray colors, would have been visually different between labeled and unlabeled positive examples. Namely, we would have observed whiter predictions concerning the labeled positive examples. In addition, we can observe that the intensity of predictions directly depends on α prior knowledge.

Empirical prior knowledge estimation: In order to confirm this dependance between output prediction values and the fraction α of training positive examples which are unlabeled, Fig. 6.5 shows histograms of pixel-level predictions for a given minibatch sample depending on α. We can observe on every histogram a mixture of two distributions. The one centered at the value 0 corresponds to the PUseg predictions for negative pixels. For the other one, its distance to the label value 1 depends on α. We can note that the center of this second distribution is close to the previously discussed theoretical value δ. As we recall that $\alpha = 1 - \delta$, we can approximately deduce from these histograms the prior knowledge α by looking at the abscisse coordinate of the distribution located on the right side. This is the distribution of predictions for our class of interest, the potentially moving obstacles. Moreover, it is interesting to see that the center of the right distribution is closer to the theoretical expected value δ when there is a high fraction α of unlabeled positive examples in the PUseg training dataset. It may be interesting to further investigate this empirical phenomenon.

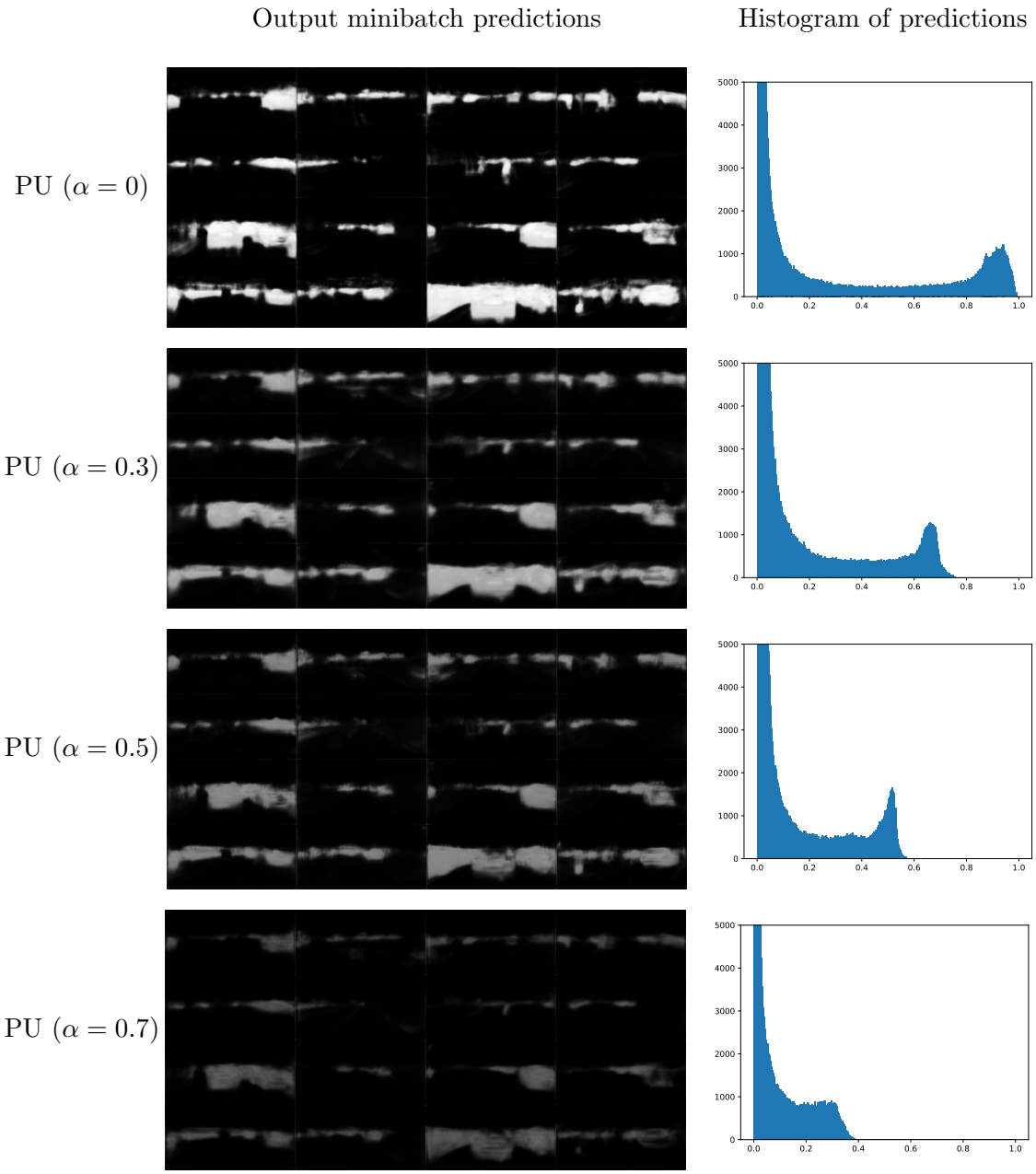


Figure 6.5: Visualization of the histograms of the pixel-level predictions of the PUseg model depending on α, for a given minibatch sample. The first column presents predictions of the PUseg model for a given sample minibatch depending on the fraction α of unlabeled positive instances included in the simulated training PUseg cityscapes dataset. The second column presents the corresponding histograms of pixelwise predictions.

A trade-off between convergence speed and hand-labeling effort? Fig. 6.6 presents the training evolution of prediction results on the test dataset of the presented PUseg model in terms of IoU scores. The threshold selected to separate positive from negative binary predictions is respectively equal to $(1 - \alpha)/2$. If we do not know the prior

knowledge α, and considering the shapes of histograms of predictions in Fig. 6.5 which are similar to gaussian distributions, we propose to define this threshold by applying GMM clustering algorithm on output predictions. In this way, predictions can be clustered into positive and negative sets with an associated probability. However, as tails of the visualized histogram distributions are relatively thick in comparaison to gaussian distribution tails, some other EM-based algorithms recently proposed [146] may be more relevant than GMM to define the threshold. We can observe in Fig. 6.6 that the PUseg model learning speed convergence highly depends on the fraction α of positive examples considered as unlabeled during the training. For instance, an IoU score of 0.35 is obtained with $\alpha = 0.3$ before 300 epochs, while with $\alpha = 0.7$ the same IoU prediction score is obtained after more than 1400 epoch iterations.

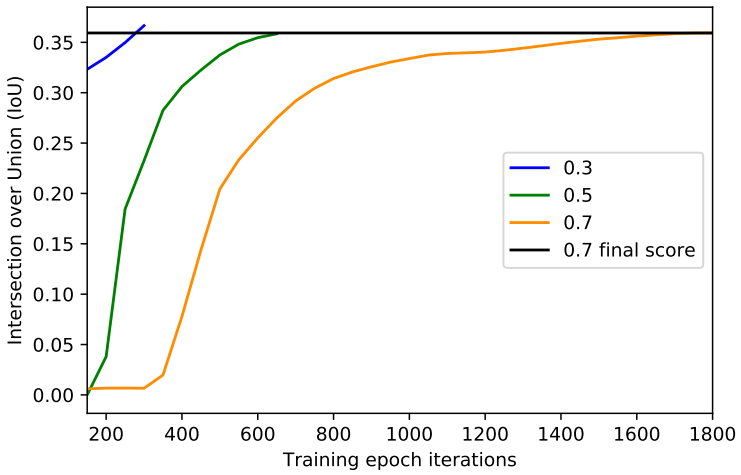


Figure 6.6: Evolution of IoU test score during training epoch iterations on simulated PU Cityscapes segmentation datasets depending on α.

As a complementary information to Fig. 6.6, final target IoU scores depending on epoch iteration and α are indicated in Table 6.3. This confirms the previous analysis of Fig. 6.6: The higher α is, the lower the model convergence speed is along training epoch iterations.

Moreover, from a practical point of view, the following idea could be considered based on Table 6.3: *For a given prediction accuracy score expected in the target application, and depending on the available computational ressources, one can decide the fraction of instances to label in its offline training dataset.*

6.2.4 Conclusion

To sum up, the proposed study has been focused on the output predictions of an AE trained to semantic segmentation using pixel-level PU training labels. Empirical results obtained confirm both following theoretical expectations:

Table 6.3: Comparative results in terms of IoU score on simulated PU Cityscapes segmentation datasets depending on α. The positive class represents potentially moving obstacles. The negative class represents all the counter-examples of the positive class. This is a One-vs-Rest binary image segmentation task.

IoU test scores	One-vs-Rest$_{\text{Cityscapes}}$			
training labels	PN (baseline)	PU		
α	0	0.3	0.5	0.7
Epoch iteration	200	350	650	1800
Positive class IoU	0.36	0.38	0.36	0.36

- AutoEncoder pixel-level output predictions can be independantly interpreted as classifier output predictions such that an AE can be trained using Positive Unlabeled pixel-level labels.
- Prior knowledge concerning the unlabeled fraction of positive instances can be coarsly approximated using the distribution of the segmenter AE output predictions.

To the best of our knowledge, it is the first time that a PU learning process is studied for the semantic image segmentation task. More importantly, we think that the presented PUseg model can be considered as a baseline reference because of its simplicity while demonstrating empirical results consistent with biased PU training loss function theoretical functionning. Therefore we expect that existing state-of-the-art PU classification techniques can be directly applied for image segmentation by using an AutoEncoder framework. This opens several application perspectives to address perception tasks of major importance like urban scene understanding in autonomous driving.

This study enables to answer the second main question of this chapter as follows: *Yes, it is possible to deal with a Positive Unlabeled segmentation dataset, and to coarsly estimate the prior knowledge.* However, the proposed framework still needs further improvements in order to be competitive with fully supervised segmentation techniques of the current state-of-the-art.

6.3 Unsupervised classification of urban moving obstacles using temporal information

Previous sections of this chapter have proposed to use PU learning strategies to reduce the labeling effort for potentially moving obstacles detection and segmentation. In this section, as a complementary and exploratory study[1], we propose to classify the identified moving

[1]This section study has been realized through an internship supervised in the context of this thesis.

obstacles, without hand labeled data, into subcategories by using their respective temporal information.

More specifically, we propose to provide motion pattern temporal information of obstacles to classify in input of a state-of-the-art image clustering approach [79]. The proposed approach can be decomposed into the two following stages:

- (1) The former stage consists of
  - segmenting moving obstacles using GMM background subtraction [182],
  - detecting moving obstacles inspired by the self-supervised detection technique proposed in [12],
  - tracking detected moving obstacles using the Simple Online and Realtime Tracking method (SORT) proposed in [11],

  from a static point of view. Using the prior knowledge that the ego-vehicle is static ensures more robust motion analysis of visual patterns of interest. In this way, sequences of image patches, as illustrated in Fig. 6.7, corresponding to motion patterns of detected moving obstacles can be automatically extracted.

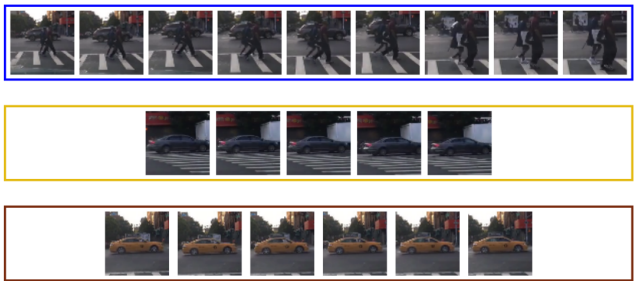


Figure 6.7: Patch sequences of moving obstacles automatically extracted from a static monocular video sequence.

- (2) Then, the second stage is to use these sequences of patches corresponding to potentially moving obstacles in input of an image clustering approach.

The usefulness of the proposed system is demonstrated through empirical experiments performed on BDD100K dataset [170]. This self-supervised training process improves the prediction performances of the exploited image clustering approach. Some illustrations of the proposed temporally self-supervised system can be found in Fig. 6.8.

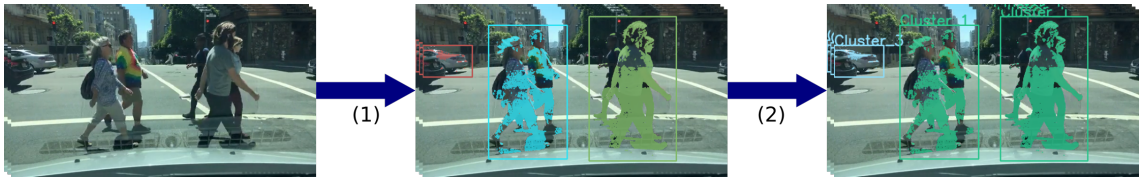


Figure 6.8: Inputs and outputs of the proposed visual monocular temporally self-supervised system for moving obstacles segmentation, detection and classification without hand-labeled data.

Chapter 7

Conclusion and perspectives

Contents

To conclude on this thesis, this chapter draws conclusions and perspectives of the previously presented research work and contributions.

7.1 Conclusion

This thesis research work aimed at reducing or completely avoiding the need of hand labeled training data for image analysis tasks, such as image classication, object detection and pixel-level semantic segmentation, in the context of the autonomous vehicle perception.

We first have presented existing Self-Supervised Learning (SSL) solutions in the context of autonomous vehicle perception tasks, as they propose to completely avoid the need of hand labeled data. For instance, SSL approaches can deal with low level perception taks such as depth map estimation from a monocular camera or high level perception tasks such as traversable area segmentation, and moving obstacles detection and tracking. Nevertheless, the SSL state-of-the-art still leaves the door open for some tasks of broad interest, such as the semantic classification, detection and segmentation of static potentially moving obstacles. Consequently, with the goal to address this challenge, we have proposed weakly supervised strategies presenting promising prediction performances. More specifically, we have proposed:

- Two solutions for dealing with Positive Unlabeled (PU) learning without the need of prior knowledge. They have been tested and compared on small-scale datasets of

natural tiny images. Moreover, we have shown that using Generative Adversarial Networks (GANs) can help to better generalize the distribution of the training dataset. In practice, this semantic data augmentation improves the prediction performances of the classifier used in the proposed frameworks.

- One solution for dealing with asymmetric noisy labeled and small-scale datasets of natural tiny images. The proposed approach consists of adapting GANs training loss functions in order to generate cleanly labeled augmented datasets using few noisy labels. Empirical results demonstrate the potential and usefulness of the proposed approach for dealing with asymmetric binary noisy labels.

Next, as the motivation of this study is to detect, segment, and classify potentially moving obstacles, we propose to adapt the previously studied PU learning concepts to deal with object detection and pixel-level semantic segmentation tasks. For this purpose, we reformulate these tasks as image classification tasks. More specifically, object detection can be considered as sliding window classification, and image semantic segmentation as independant per-pixel classifications. Empirical results demonstrate the possibility to detect and segment potentially moving obstacles using PU learning. Moreover, the prediction results, as a function of the ratio of labeled data, provide some insights concerning the amount of labeled data required to accomplish the target tasks.

Finally, the main advantage of proposed approaches is their ability to generalize the target information over weakly labeled datasets. Nonetheless, the prediction performances obtained still have the potential to be improved for real-world applications, and the scope of their autonomous driving applications to be extended. For instance, presented approaches only focus on binary discriminative tasks. Moreover, detection, segmentation and classification weakly supervised tasks have been studied separately. It would also be interesting to unify them into a complete SSL framework for potentially moving obstacles detection, segmentation and classification.

The following Sec. 7.2 presents corresponding perspectives.

7.2 Future research directions

Concerning the perspectives, we respectively propose future directions for potential improvements or extensions of the presented contributions.

7.2.1 Positive unlabeled learning using unlabeled data generation

PGAN system optimization presented in chapter 3 can be carried on by testing other generative models instead of the WGAN [1] or WGAN-GP variant [66], like BEGAN [10],

or the progressive growing GAN [85]. The latter can deal with very high dimensional images as shown empirically in Fig. 7.1[1]. We have tested it on urban images captured in Versailles. Some other types of generative models as variational autoencoders (VAEs) may be compatible with the proposed framework and may enable to generalize the proposed approach to other generative models. Another orientation is to exploit the z latent space of GANs to perform linear arithmetic operations, as in [13], to generate more relevant fake samples.

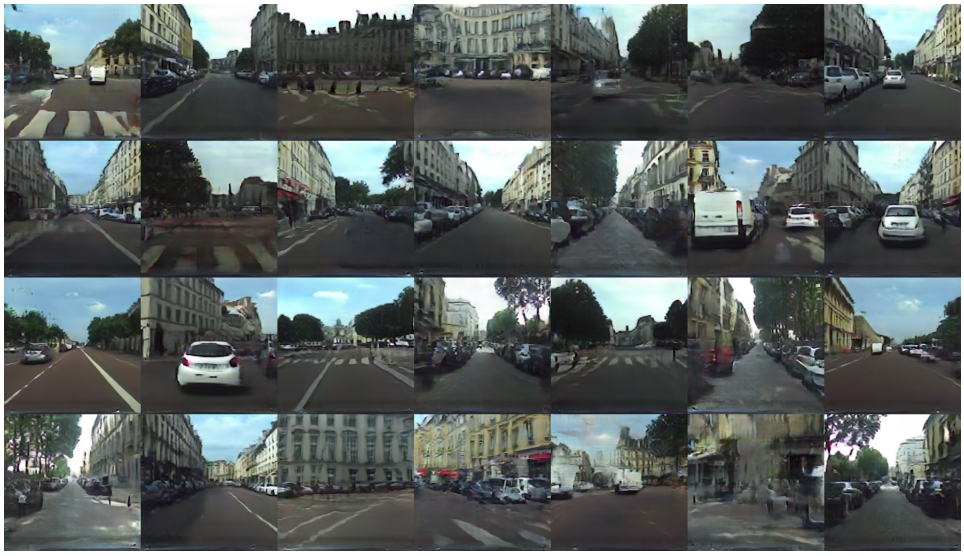


Figure 7.1: A sample of images generated (512*512*3) with the progressive growing GAN that we trained on a set of 5 000 unlabeled images captured in the city of Versailles, in France.

Overall, considering the promising performances obtained for image classifiation, an interesting direction is to extend this PU method to more complex tasks such as semantic segmentation [3].

7.2.2 Counter-examples generation from a Positive Unlabeled dataset

Concerning the second GAN-based PU method proposed in chapter 4, as adversarial and weakly supervised learning techniques are continuously evolving, we believe that the proposed approach stability, prediction performances, and computational cost still have the potential to be improved. For instance, recent promising GAN training approaches [17], not necessarily using BN, may be suitable to extend the proposed approach for higher dimensional image datasets.

Some other contribution perspectives can be considered as discussed in the following sections.

[1]Images have been captured using the camera ZED (https://www.stereolabs.com/zed/).

7.2.2.1 Discriminator predictions using weighted loss functions

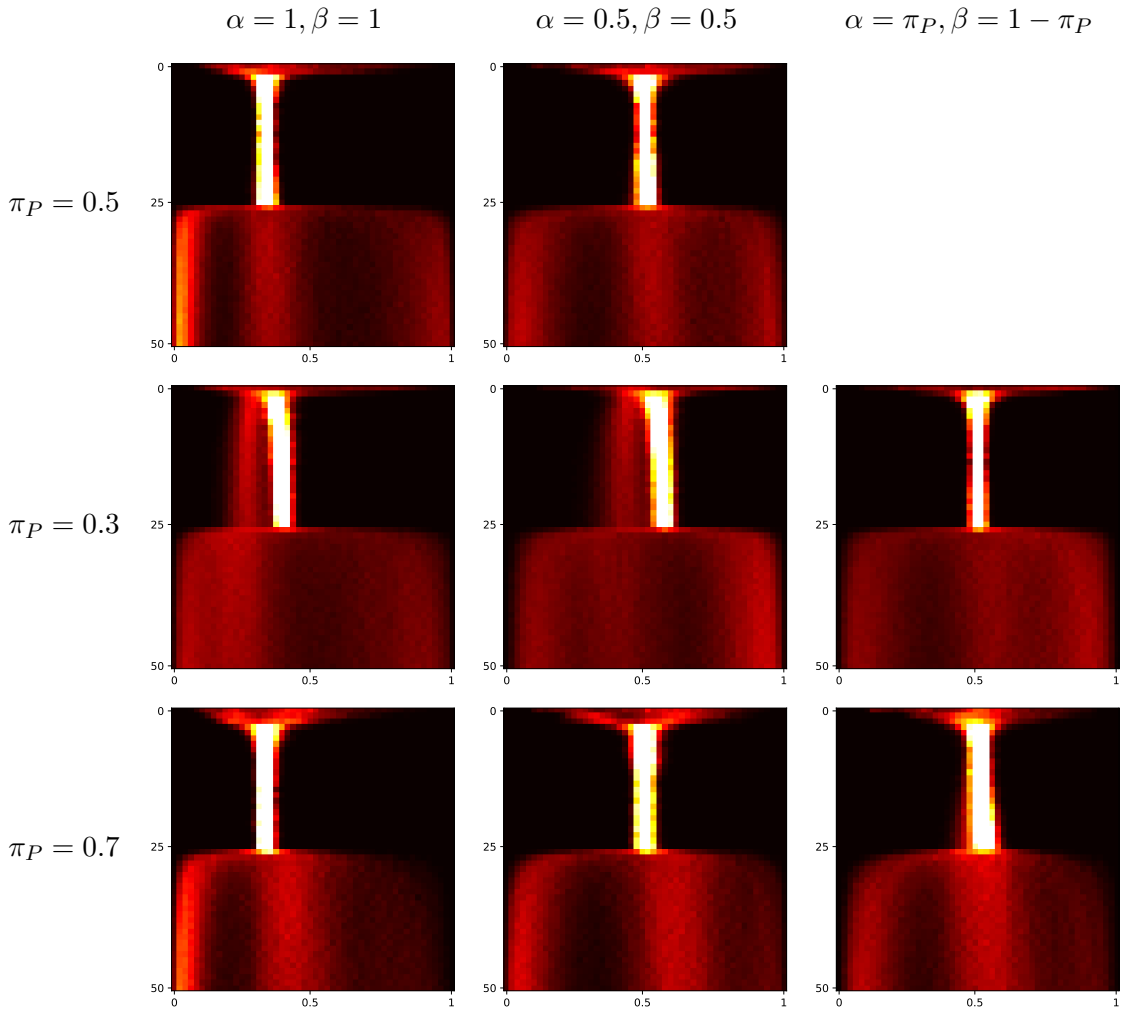


Figure 7.2: Discriminator behaviour study in function of α and β during the interruption of the adversarial training from 25 until 50 epoch iterations. Tests are realized on the MNIST dataset for the 8-vs-3 task for several values of π_P. The vertical axis represents the training epochs iterations. The horizontal axis represents output values predicted by the discriminator. Each horizontal line of pixels represents a histogram for a given epoch iteration. Pixel regions with clear hot colors express high densities of predictions. A sample of 10 000 examples composed by two third of unlabeled examples and one third of generated examples is predicted at each epoch iteration.

The Figure. 7.2 shows some impacts of applying weight hyper-parameters α and β to the terms of the discriminator loss function L_D, such that we obtain

$$\begin{aligned} L_D(G, D) = & \mathbb{E}_{x_U \sim p_U}[-logD(x_U)] \\ & + \alpha \cdot \mathbb{E}_{x_P \sim p_P}[-log(1 - D(x_P))] \\ & + \beta \cdot \mathbb{E}_{z \sim p_z}[-log(1 - D(G(z)))]. \end{aligned} \tag{7.1}$$

From 0 to 25 epochs, both D and G are alternatively trained. Then, from 25 until 50 epochs the discriminator D training continues, while that of generator G is stopped (i.e. G training parameters are fixed). The former step enables to observe the discriminator prediction behaviour during the adversarial training. The latter step enables to observe, independently to G, the discriminator behaviour such that unlabeled positive, unlabeled negative, and generated examples distributions become relatively easier to separate with D.

When $\alpha + \beta = 1$, with $\alpha = \pi_P$ and hence $\beta = 1 - \alpha = 1 - \pi_P$, the prediction dispersion during the first step seems to be reduced and to remain centered around the middle value 0.5.

During the second step, each prediction histogram becomes a mixture of distributions. We can observe three distinct distributions corresponding respectively from left to right to generated, unlabeled positive, and unlabeled negative distributions. In addition, we can observe that the density repartion of D predictions directly depends on π_P fraction:

- When $\pi_P = \alpha = 0.3$, the unlabeled positive distribution (i.e. the distribution centered at 0.5) includes more predictions than the unlabeled negative distribution (i.e. the distribution at 1).
- When $\pi_P = \alpha = 0.5$, the unlabeled positive distribution (i.e. the distribution centered at 0.5) is equivalent in terms of density to the unlabeled negative distribution (i.e. the distribution at 1).
- When $\pi_P = \alpha = 0.7$, the unlabeled positive distribution (i.e. the distribution centered at 0.5) includes less predictions than the unlabeled negative distribution (i.e. the distribution at 1).

Those histograms evolutions during the training reveal qualitatively a correlation between the fraction π_P and hyper-parameters α and β. This opens new perspectives to identify the prior knowledge and possibly to adapt automatically the hyper-parameters values during the adversarial training of the proposed framework.

7.2.2.2 GAN-based PU framework adaptation for image segmentation

As it has been previously empirically demonstrated in the literature that we can adversarially train with a discriminator a segmenter model for image semantic segmentation task [110], one can envision the extension of the proposed GAN-based PU framework for this task by using PU pixel-level training labels. We present in Fig. 7.3 an adaptation of the presented D-GAN framework for potentially moving obstacles segmentation using only positive and unlabeled pixel labels during the training. The generator G remains identical in the sense

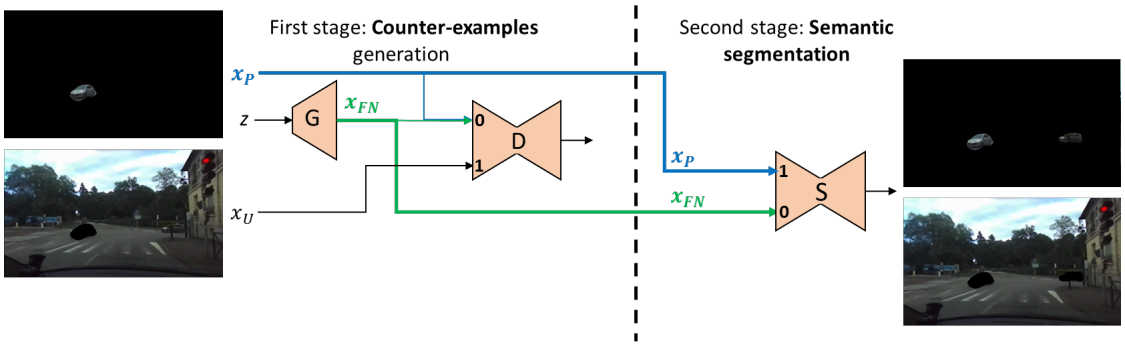


Figure 7.3: Potential future adaptation of the D-GAN framework for PU semantic segmentation.

that it generates counter-example images from a random latent vector z using a decoder architecture. However, the discriminator D and the classifier C replace their encoder architecture with an autoencoder architecture. This enables to discriminate and classify each pixel of the input image such that a pixelwise semantic segmentation is obtained as output. In addition, training loss functions previously used for G, D and C for image classification may prospectively be applied independently on each per-pixel output prediction in this novel context. Per-pixel classification using an autoencoder based architecture in order to deal with semantic segmentation has been previously proposed by methods such as FCN [108], U-Net [148], Segnet [3] or PSPNet [176].

7.2.3 Noisy labeled learning using GANs

This section presents perspectives for the GAN-based noisy labeled learning approach NL-GAN proposed in chapter 5.

Multi-class NL learning using a conditional GAN: In order to improve the proposed NL-GAN framework for complex multiclass image datasets, it may be relevant to take into consideration recent GANs using several generators [173]. More closely related to our topic, it has also been recently[2] proposed in [163] a conditional GAN model robust to label noise called RCGAN. It uses a confusion matrix representing the label noise and a projection discriminator [121] in order to deal with the corrupted training set. It may be interesting to adapt the conditional GAN ResNet architecture exploited by the RCGAN to our NL-GAN framework in order to improve the quality of generated samples and adapt it for multiclass classification. Moreover, as RCGAN can deal with multiclass datasets containing few[3] noisy

[2]Published at the end of 2018 (i.e. concurrently to the submission of the article presented in chapter 5).

[3]If we compare the same fraction of noise α between binary and multiclass classification, we can notice that in binary classification this consists of having α fraction of noise in the corrupted counter-examples set. However, the same fraction α of corrupted labels in a multiclass dataset of n classes means that there is a fraction of $(n-1)\cdot\alpha$ corrupted labels for a given class which are present in the corrupted counter-examples

labels, additional experiments comparing its performances with the proposed approach for both multiclass noisy labels and high binary asymmetric label noise could be interesting for a deeper and updated state-of-the-art comparative study.

Improved empirical estimation of intermediate labels δ_P and δ_N: On another note, in the absence of prior knowledge information we have naively proposed to use GMM clustering algorithm to identify the intermediate labels δ_P and δ_N. However, concerning the shape of the probability density function of the discriminator predictions, this implies the assumptions that such distributions are similar to gaussian distributions. Consequently, it may be interesting to reconsider different hypotheses about the shape of the probability density function of the CNN classifier predictions. For instance, we could test more noise-permissive clustering techniques such as the EM-like algorithm, recently proposed in [146], which can deal with various tails distributions.

Design training loss functions specialized for the proposed NL-GAN framework: Concerning the binary cross-entropy loss function used in the proposed generator and discriminator training loss functions, its asymmetry between left and right slopes can potentially change the learning convergence between corrupted and correct labels. Thus, the theoretical $\arg\min_{x_P \sim p_P} L_D = \delta_P$ and $\arg\min_{x_N \sim p_N} L_D = \delta_N$ values depending on π_P and π_N run the risk to not be empirically reached. This potential issue has a direct impact on generators G_P and G_N ability to respectively learn distributions p_P and p_N. Consequently it may be interesting to further investigate more relevant classification symmetric loss functions compatible with such a GAN-based framework.

Some other interesting perspectives could be to use:

- continuous conditional variables in the latent space of NL-GAN generators in order to set the probability estimated with GMM on D predictions, for a given generated sample, to follow a target distribution p_P or p_N.
- NL-GAN for assessing, without ground-truth, hand-crafted methods within a self-supervised framework.
- NL-GAN downstream to clustering methods by considering every cluster label as a noisy label. A similar state-of-the-art strategy [82] is to perform Unlabeled Unlabeled classification. This consists of learning to better clusterize instances from two unlabeled sets which contain asymmetric conditional fractions of each class.

class sets. Consequently, multiclass NL approaches are systematically tested with relatively small fractions of noise compared to binary NL approaches.

7.2.4 Applications

7.2.4.1 PU learning for vehicle detection on aerial images

Some perspectives concerning the PU detector presented in chapter 6 are envisioned:

- The operator may wish to identify vehicles of smaller sizes. Thus it can be required to design a learning model able to deal with patterns of smaller resolutions. In order to classify corresponding small patches, another solution can be to integrate in the first-stage generative step some super-image resolution techniques, as in [34] where it is proposed a deep multi-scale strategy, or in [97] where GANs potential interest for this task is highlighted.

- Encourage the learning of potentially more relevant counter-examples by extracting a higher fraction of salient unlabeled patches during the training PU dataset preparation as previously proposed in [140].

- Combine standard transformation data augmentation techniques with generative models for positive labeled examples as proposed in [28] to help the second-stage classifier to better generalize the boundary between positive and negative examples.

- Adapt PU techniques to more advanced models, such as Faster R-CNN [144], SSD [106] and Yolov3 [143]. For example, the region proposal process can be based on image classification principles such that PU classification techniques can be used.

7.2.4.2 Positive Unlabeled analysis for semantic segmentation of urban potentially moving obstacles

It may be interesting to integrate the PU segmentation (PUseg) model, presented in chapter 6, in the Self-supervised framework presented in the same chapter, in order to demonstrate its usefulness for learning to segment potentially moving obstacles without hand labeled training data.

We have proposed throught this study to consider PUseg challenge as per-pixel PU classification problem. Nonethelss, we think that it would be interesting to investigate novel reasearch for designing segmentation techniques specifically designed for this PUseg task. The recent competitive state-of-the-art Deeplab model [20] architecture could be investigated in this PUseg context.

It may be interesting to also study the effect of regularization techniques exploited in previous chapters 4 and 5 in the context of this novel PUseg framework. However, the training process is relatively longer in comparison to a PU training for image classification. Thus,

the previously selected combinations of regularization techniques need to be reconsidered in this context as they can considerably increase the training computational cost.

Concerning the potential applications for aerial and satellite images analysis as proposed in [132] by using deep neural networks, the proposed PUseg learning strategy may provide complementary information to the previously presented PU detector.

7.2.4.3 Unified self-supervised learning application perspectives

By combining the three different approaches presented in chapter 6, respectively for PU detection, PU segmentation, and unsupervised classification, we foresee future research perspectives to develop a unified Positive Unlabeled learning framework, self-supervised by temporal information. Inspiration for such a unified deep learning framework can come from supervised techniques such as Mask R-CNN [71], or panoptic segmentation concepts introduced in [89]. Such an envisioned SSL framework may then be able, by only learning on the spatio-temporal variations of the surrounding environment without any human supervision, to provide rich information concerning potentially moving obstacles as their shape, motion patterns, and their category.

Moreover, by automatically estimating the fraction of mislabeled predicted information, as proposed in chapter 6 for PUseg, then such a SSL PU learning system may have the capacity to autonomously evaluate itself without human supervision. Moreover, such a self-evaluation could enable to update online the learning model by using fine-tuning techniques as proposed in [81]. This may provide relatively fast online adaptation of a given deep learning SSL system through an incremental learning scenario.

Moreover, concerning the classification of detected moving obstacles into subcategories, it may be interesting to identify automatically the uncommon obstacles which can have unpredictable behaviours. For instance, an unusual wild animal must be handled differently to classifical road users. We argue that this kind of situation could be addressed by using anomaly detection techniques, as proposed in [174]. This approach deals with PU datasets containing a high fraction of unlabeled positive examples. It consists of training several classifiers trough a boosting process depending on the estimated positive class prior (i.e. fraction of positive examples in the unlabeled dataset). In our context, usual road users can be considered as positive examples, and wild animals as the anomalies to detect.

On another note, beyond the scope of this thesis study which focuses exclusively on visual monocular camera data analysis using deep learning, it may be interesting to exploit additional sensor information as proposed in [36] and [4] for obstacle analysis.

7.2.4.4 Create an autonomous driving dataset specifically designed for temporally self-supervised learning

In closing of this thesis research, we believe that it is of broad interest, for both academic and industrial autonomous driving research communities, to create and make publicly available a dataset designed for temporally self-supervised learning.

Nowadays, to the best of our knowledge, the largest dataset with video sequences intended for autonomous vehicle perception is BDD100K [170]. We observed on this dataset that for a given recording time, one can observe a wider number of moving obstacles on static scenes rather than during ego-vehicle movements. It turns out that BDD100K contains in total less than 1 hour of recording for static scenes under daytime clear weather conditions. Thus, increasing such a quantity of static video scene may improve unsupervised analysis of potentially moving obstacles.

As a matter of fact, one can aspire to record ego-vehicle sensor data at strategic urban road intersection places, with a wide variety of potentially moving obstacles. In this way, one could easily and effectively acquire a considerably large amount of rich information for improving current urban potentially moving obstacles analysis.

Furthermore, in order to effectively learn on such large-scale and high resolution vision data, it may be interesting to study strategies allowing to directly interpret compressed information, as previously proposed in [161] for camera motion estimation.

Appendix A

Corrupted training loss functions formalization

"Give me a place to stand, and I'll move the earth."

- Archimede

Contents

A.1 Training loss functions

One can find below the definitions of binary cross-entropy H and Mean Squared Error MSE loss functions:

- Binary Cross-Entropy H:

$$H(\hat{y}, y_t) = \mathbb{E}_{\hat{y}\sim p}[-y_t log(\hat{y}) - (1 - y_t) log(1 - \hat{y})], \tag{A.1}$$

- Mean Squared Error MSE:

$$MSE(\hat{y}, y_t) = \mathbb{E}_{\hat{y}\sim p}[(\hat{y} - y_t)^2], \tag{A.2}$$

with $\hat{y} \in (0, 1)$ the sample following a distribution p representing the predictions of a given learning model, and $y_t \in [0, 1]$ the corresponding training label.

A.2 Corrupted training loss functions, a constrained formalization

When some labels in a binary dataset are corrupted, then the training consists of associating two different labels for a given distribution depending on the fraction of corrupted labels. Let $\hat{y}$ be a sample prediction following a distribution p. By performing an empirical risk minimization of a loss function L, one can train a flexible learning model to associate $\hat{y}$ with two labels y_A and y_B depending respectively on weights α and β as follows

$$L(\hat{y}) = \mathbb{E}_{\hat{y}\sim p}[\alpha \cdot l(\hat{y}, y_A)] + \mathbb{E}_{\hat{y}\sim p}[\beta \cdot l(\hat{y}, y_B)], \tag{A.3}$$

with l an arbitrary cost function which can be H or MSE.

Then, expectation linearity enables to obtain

$$L(\hat{y}) = \mathbb{E}_{\hat{y}\sim p}[\alpha \cdot l(\hat{y}, y_A) + \beta \cdot l(\hat{y}, y_B)]. \tag{A.4}$$

It turns out that the $\arg\min_{\hat{y}}$ of L can be obtained as well by using a single intermediate label value δ which jointly replaces y_A and y_B depending on α and β with the loss function L_δ that we propose to define as follows

$$L_\delta(\hat{y}) = \mathbb{E}_{\hat{y}\sim p}[\gamma \cdot l(\hat{y}, \delta)], \tag{A.5}$$

with γ depending as well on α and β. More specifically, with $l = \{H, MSE\}$, we have the following relation between L and L_δ:

$$\begin{aligned} \arg\min_{\hat{y}\in(0,1)}[\alpha \cdot l(\hat{y}, y_A) + \beta \cdot l(\hat{y}, y_B)] &= \arg\min_{\hat{y}\in(0,1)}[l(\hat{y}, \delta)] \\ &= \delta, \end{aligned} \tag{A.6}$$

with

$$\delta = \frac{\alpha y_A + \beta y_B}{\alpha + \beta}. \tag{A.7}$$

Proof of Equation. (A.6) can be demonstrated by finding the value of $\hat{y}$ for which the partial derivative of L depending on $\hat{y}$ is equal to zero, such that we have

$$\frac{\partial(L(x))}{\partial(x)} = 0. \tag{A.8}$$

This can be generalized for multiclass labels as follows

$$\begin{aligned} \arg\min_{\hat{y}\in(0,1)}[\sum_{i=1}^{m}(w_i \cdot l(\hat{y}, y_i))] &= \arg\min_{\hat{y}\in(0,1)}[l(\hat{y}, \frac{\sum_{i=1}^{m} w_i y_i}{\sum_{i=1}^{m} w_i})] \\ &= \frac{\sum_{i=1}^{m} w_i y_i}{\sum_{i=1}^{m} w_i}, \end{aligned} \tag{A.9}$$

with w_i the probability to associate the label y_i with the predicted sample $\hat{y}$. From physical and geometrical perspectives point of view, this recalls the barycenter of masses formula enabling to estimate the center of gravity, introduced by Archimede.

Corrupted fractions estimation: Furthermore, let one suppose that a given learning model trained using L has correctly converged towards the minimum of L. Hence, we may have $\hat{y} = \delta$. So one can deduce the potentially unknown prior α using the equivalence

$$\hat{\delta} = \frac{\alpha y_A + \beta y_B}{\alpha + \beta} \Leftrightarrow \alpha = \beta \cdot \frac{y_B - \hat{\delta}}{\hat{\delta} - y_A}, \tag{A.10}$$

with y_A, y_B and β the prior parameters known.

Next sections show the demonstration of eq. (A.6) with the Binary Cross-Entropy H, and the Mean Squared Error MSE cost functions for dealing with two different labels associated with the same sample distribution.

A.2.1 Mean Squared Error (MSE)

With L_{MSE} the loss function version of L using the Mean Squared Error cost function $l = MSE$ defined in eq. (A.2), the derivative of L_{MSE} is as follows

$$\frac{\partial(L_{MSE}(\hat{y}))}{\partial(\hat{y})} = \frac{\partial(\alpha \cdot MSE(\hat{y}, y_A))}{\partial(\hat{y})} + \frac{\partial(\beta \cdot MSE(\hat{y}, y_B))}{\partial(\hat{y})}, \tag{A.11}$$

with

$$\begin{aligned} \frac{\partial(\alpha \cdot MSE(\hat{y}, y_A))}{\partial(\hat{y})} =& \alpha \cdot \frac{\partial(\mathbb{E}_{\hat{y}\sim p}[(\hat{y} - y_A)^2])}{\partial(\hat{y})} \\ =& \alpha \cdot \mathbb{E}_{\hat{y}\sim p}[2\hat{y} - 2y_A], \end{aligned} \tag{A.12}$$

and respectively

$$\begin{aligned} \frac{\partial(\beta \cdot MSE(\hat{y}, y_B))}{\partial(\hat{y})} =& \beta \cdot \frac{\partial(\mathbb{E}_{\hat{y}\sim p}[(\hat{y} - y_B)^2])}{\partial(\hat{y})} \\ =& \beta \cdot \mathbb{E}_{\hat{y}\sim p}[2\hat{y} - 2y_B], \end{aligned} \tag{A.13}$$

such that we obtain

$$\frac{\partial(L_{MSE}(\hat{y}))}{\partial(\hat{y})} = \mathbb{E}_{\hat{y}\sim p}[2(\alpha + \beta)\hat{y} - 2(\alpha y_A + \beta y_B)]. \tag{A.14}$$

Then, we can deduce the value of $\hat{y}$ for which the partial derivative of L_{MSE} depending on $\hat{y}$ is equal to zero as follows

$$\begin{aligned} \frac{\partial(L_{MSE}(\hat{y}))}{\partial(\hat{y})} = 0 &\Leftrightarrow \mathbb{E}_{\hat{y}\sim p}[2(\alpha + \beta)\hat{y} - 2(\alpha y_A + \beta y_B)] = 0 \\ &\Leftrightarrow \hat{y} = \frac{\alpha y_A + \beta y_B}{\alpha + \beta}, \end{aligned} \tag{A.15}$$

such that

$$\begin{aligned} \arg\min_{\hat{y}\in(0,1)}(L_{MSE}(\hat{y})) =& \frac{\alpha y_A + \beta y_B}{\alpha + \beta} \\ =& \arg\min_{\hat{y}\in(0,1)}(MSE(\hat{y}, \frac{\alpha y_A + \beta y_B}{\alpha + \beta})). \end{aligned} \tag{A.16}$$

A.2.2 Binary Cross-Entropy

Concerning the binary cross-entropy cost function H, we first demonstrate the identity between loss functions L and L_δ when the cost function $l = H$, such that we obtain L_H defined as:

$$L_H(\hat{y}) = \mathbb{E}_{\hat{y}\sim p}[\alpha \cdot H(\hat{y}, y_A) + \beta \cdot H(\hat{y}, y_B)]. \tag{A.17}$$

The identity can be demonstrated with labels $y_A, y_B \in [0, 1]$ as follow:

$$\begin{aligned}
\alpha \cdot H(\hat{y}, y_A) + \beta \cdot H(\hat{y}, y_B) =& \alpha[-y_A log(\hat{y}) - (1 - y_A)log(1 - \hat{y})] \\
&+ \beta[-y_B log(\hat{y}) - (1 - y_B)log(1 - \hat{y})] \\
=& -(\alpha y_A + \beta y_B)log(\hat{y}) - (\alpha + \beta - \alpha y_A - \beta y_B)log(1 - \hat{y}) \\
=& (\alpha + \beta)[-(\frac{\alpha y_A + \beta y_B}{\alpha + \beta})log(\hat{y}) - (1 - \frac{\alpha y_A + \beta y_B}{\alpha + \beta})log(1 - \hat{y})] \\
=& \gamma[-\delta log(\hat{y}) - (1 - \delta)log(1 - \hat{y})] \\
=& \gamma \cdot H(\hat{y}, \delta),
\end{aligned} \tag{A.18}$$

with $\gamma = \alpha + \beta$ and $\delta = \frac{\alpha y_A + \beta y_B}{\alpha + \beta}$. To find the $\underset{\hat{y}\in(0,1)}{\arg\min}$ of the loss function term $\gamma H(\hat{y}, \delta)$, we first estimate its partial derivative depending on $\hat{y}$ as follows

$$\frac{\partial[\gamma \cdot H(\hat{y}, \delta)]}{\partial \hat{y}} = -\gamma \cdot (\frac{\delta}{\hat{y}} + \frac{\delta - 1}{1 - \hat{y}}). \tag{A.19}$$

Second, the argument of the minimum output value can be found when this derivative is equal to zero, such that we have to solve the following equation

$$-\gamma \cdot (\frac{\delta}{\hat{y}} + \frac{\delta - 1}{1 - \hat{y}}) = 0, \tag{A.20}$$

bringing us to the following result

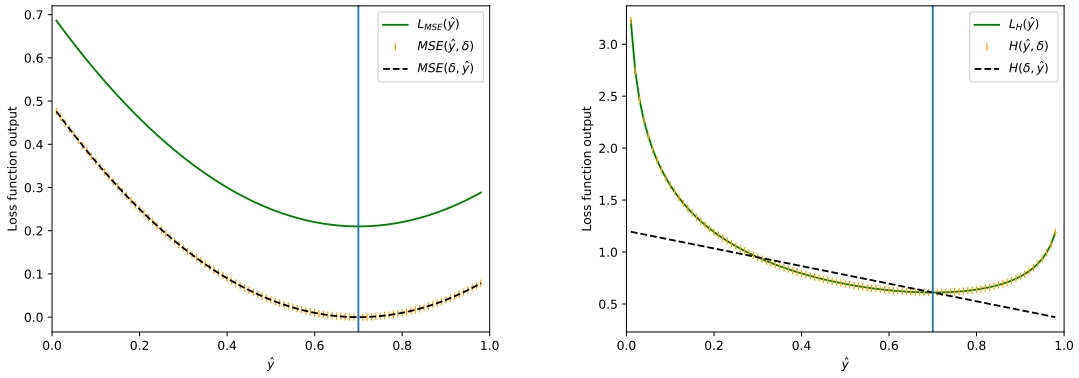
$$\begin{aligned}
-\gamma \cdot (\frac{\delta}{\hat{y}} + \frac{\delta - 1}{1 - \hat{y}}) =& 0 \\
\Leftrightarrow \frac{\delta}{\hat{y}} =& \frac{1 - \delta}{1 - \hat{y}} \\
\Leftrightarrow \frac{1 - \hat{y}}{\hat{y}} =& \frac{1 - \delta}{\delta} \\
\Leftrightarrow \hat{y} =& \delta.
\end{aligned} \tag{A.21}$$

Thus we have as well by using H the following statement:

$$\begin{aligned}
\underset{\hat{y}\in(0,1)}{\arg\min}(L_H(\hat{y})) =& \underset{\hat{y}\in(0,1)}{\arg\min}(H(\hat{y}, \frac{\alpha y_A + \beta y_B}{\alpha + \beta})) \\
=& \frac{\alpha y_A + \beta y_B}{\alpha + \beta}.
\end{aligned} \tag{A.22}$$

A.3 Graphical visualization

This section shows graphically in Fig. A.1 the loss functions L, $l(\hat{y}, \delta)$ and its permutation $l(\delta, \hat{y})$ depending on $\hat{y}$ and $l = \{MSE, H\}$, with $\alpha = 0.3$, $\beta = 0.7$, $y_A = 0$, $y_B = 1$. As MSE is symmetric, and H is not, we can observe graphically that $MSE(\hat{y}, \delta) = MSE(\delta, \hat{y})$, while $H(\hat{y}, \delta) \neq H(\delta, \hat{y})$.



(a) (b)

Figure A.1: Corrupted loss functions visualization. (a) Shows $MSE(\hat{y}, \delta)$, $MSE(\delta, \hat{y})$, L_{MSE} and $\arg\min(L_{MSE})$ depending on $\hat{y}$. Similarly, (b) shows $H(\hat{y}, \delta)$, $H(\delta, \hat{y})$, L_H and $\arg\min(L_H)$ depending on $\hat{y}$.

Appendix B

Résumé

Dans le contexte de la perception du véhicule à conduite déléguée, l'intérêt de la communauté pour les méthodes d'apprentissage profond a continuellement augmenté pendant ces deux dernières décennies. Cela peut être expliqué par le fait que ces techniques fournissent les meilleures performances de prédiction de l'état de l'art actuel, pour plusieurs tâches de vision par ordinateur. En particulier, ces méthodes peuvent fournir des informations sémantiques riches concernant les obstacles de formes complexes rencontrés dans des scénarios de conduite déléguée, à partir de divers types de données et dans des conditions climatiques variées. Cependant, obtenir les meilleures performances en prédiction de l'état l'art demande souvent un grand nombre de données manuellement labélisées, provenant du cas d'application ciblé. Le problème est que la labélisation manuelle a un coût non négligeable. Néanmoins, dans le contexte d'un véhicule équipé de capteurs, les données non labélisées quant à elles, peuvent être obtenues relativement plus facilement. Il se trouve qu'une catégorie de méthodes d'apprentissage, dites faiblement supervisées, permettent d'exploiter directement des données partiellement labélisées. Ainsi, notre objectif dans cette thèse est de réduire au possible le besoin en données manuellement labélisées en proposant des modèles d'apprentissage faiblement supervisés.

Nous commençons par présenter un type de méthodes d'apprentissage dites auto-supervisées. Elles consistent à substituer les données manuellement labélisées par des méthodes capable de générer automatiquement en amont des labels d'entraînement exploitables. Les techniques d'apprentissage auto-supervisées ont prouvé leur utilité dans le passé pour l'évitement d'obstacles et la planification de trajectoires à travers des environnements changeants, en apprenant lors de la phase d'application. Plus récemment, elles ont aussi été appliquées pour l'estimation de cartes de profondeurs, la segmentation de routes goudronnées, et pour le suivi et la segmentation d'obstacles en mouvement. Cependant, les méthodes auto-supervisées de l'état de l'art laissent encore la porte ouverte pour la détection, la segmentation, et la classification des obstacles statiques potentiellement mobiles. Ces

derniers que l'on rencontre également souvent dans le contexte de la conduite déléguée, peuvent être par exemple des voitures arrêtées à une intersection, ou des piétons patientant pour traverser la voie. En conséquence, nous proposons dans cette thèse trois nouvelles approches faiblement supervisées, avec l'objectif final de percevoir de tels usagers de la route en utilisant un système auto-supervisé.

Les deux premières contributions de ce travail ont pour objectif de répondre au problème de classification d'images partiellement labélisées, tel que l'effort de labélisation peut être focalisé exclusivement sur notre classe d'intérêt, la classe positive. La première approche consiste essentiellement à régulariser le biais provoqué par un apprentissage directement effectué sur des données partiellement labélisées. La deuxième approche proposée consiste à générer des contre exemples pertinents de la classe positive afin de résoudre le problème de sur-apprentissage de la première approche. Ensuite, nous proposons une approche pouvant traiter des données d'entraînement possédant une grande fraction de faux labels, et disponibles en faible quantité. La deuxième contrainte a été palliée efficacement en effectuant de l'augmentation de donnees, au travers de modèles génératifs antagonistes capables d'apprendre a généraliser la distribution des classes qui nous intérèssent. Ensuite, nous proposons de démontrer le potentiel de telles méthodes de classification d'image faiblement supervisées pour les deux applications réelles suivantes : détection et segmentation des obstacles potentiellement mobiles.

Enfin, nous présentons une conclusion sur ce travail de recherche, suivie par des perspectives de recherches et applications futures.

Bibliography

[1] Martin Arjovsky, Soumith Chintala, and Lon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017.

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[3] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[4] Adrien Bak, Samia Bouchafa, and Didier Aubert. Detection of independently moving objects through stereo vision and ego-motion extraction. In *2010 IEEE Intelligent Vehicles Symposium*, pages 863–870. IEEE, 2010.

[5] Dana H Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2):111–122, 1981.

[6] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.

[7] Jessa Bekker and Jesse Davis. Learning from positive and unlabeled data: A survey. *arXiv preprint arXiv:1811.04820*, 2018.

[8] Bilel Benjdira, Taha Khursheed, Anis Koubaa, Adel Ammar, and Kais Ouni. Car detection using unmanned aerial vehicles: Comparison between faster r-cnn and yolov3. In *2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS)*, pages 1–6. IEEE, 2019.

[9] Nicola Bernini, Massimo Bertozzi, Luca Castangia, Marco Patander, and Mario Sabbatelli. Real-time obstacle detection using stereo vision for autonomous ground vehicles: A survey. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 873–878. IEEE, 2014.

[10] David Berthelot, Tom Schumm, and Luke Metz. Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017.

[11] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468. IEEE, 2016.

[12] Alex Bewley, Vitor Guizilini, Fabio Ramos, and Ben Upcroft. Online self-supervised multi-instance segmentation of dynamic objects. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 1296–1303. IEEE, 2014.

[13] Piotr Bojanowski, Armand Joulin, David Lopez-Paz, and Arthur Szlam. Optimizing the latent space of generative networks. *arXiv preprint arXiv:1707.05776*, 2017.

[14] Lon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[15] Thierry Bouwmans, Fida El Baf, and Bertrand Vachon. Statistical background modeling for foreground detection: A survey. In *Handbook of pattern recognition and computer vision*, pages 181–199. World Scientific, 2010.

[16] Guillaume Bresson, Zayed Alsayed, Li Yu, and Sbastien Glaser. Simultaneous localization and mapping: A survey of current trends in autonomous driving. *IEEE Transactions on Intelligent Vehicles*, 2(3):194–220, 2017.

[17] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.

[18] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018.

[19] Florian Chabot, Mohamed Chaouch, Jaonary Rabarisoa, Celine Teuliere, and Thierry Chateau. Deep manta: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[20] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[21] Xueyun Chen, Shiming Xiang, Cheng-Lin Liu, and Chun-Hong Pan. Vehicle detection in satellite images by hybrid deep convolutional neural networks. *IEEE Geoscience and remote sensing letters*, 11(10):1797–1801, 2014.

[22] F. Chiaroni, M. C. Rahal, N. Hueber, and F. Dufaux. Hallucinating a Cleanly Labeled Augmented Dataset from a Noisy Labeled Dataset Using GANs. In *IEEE International Conference on Image Processing*, 2019.

[23] F Chiaroni, M-C Rahal, N. Hueber, and Frédéric Dufaux. Hallucinating a Cleanly Labeled Augmented Dataset from a Noisy Labeled Dataset Using GANs. In IEEE, editor, *26th IEEE International Conference on Image Processing (ICIP)*, September 2019.

[24] F Chiaroni, MC Rahal, F Dufaux, and N Hueber. Classification dimages en apprenant sur des échantillons positifs et non labélisés avec un réseau antagoniste génératif. *CNIA & RJCIA 2018*, page 35.

[25] Florent Chiaroni, Ghazaleh Khodabandelou, Mohamed-Cherif Rahal, Nicolas Hueber, and Frédéric Dufaux. Generating relevant counter-examples from a positive unlabeled dataset for image classification. *Submitted to Pattern Recognition. arXiv preprint arXiv:1910.01968*, 2019.

[26] Florent Chiaroni, Mohamed-Cherif Rahal, Nicolas Hueber, and Frederic Dufaux. Learning with a generative adversarial network from a positive unlabeled dataset for image classification. In *IEEE International Conference on Image Processing*, 2018.

[27] Florent Chiaroni, Mohamed-Cherif Rahal, Nicolas Hueber, and Frédéric Dufaux. Self-supervised learning for autonomous vehicles perception: A conciliation between analytical and learning methods. *Submitted to IEEE Signal Processing Magazine (under revision). arXiv preprint arXiv:1910.01636*, 2019.

[28] Yi-Min Chou, Chien-Hung Chen, Keng-Hao Liu, and Chu-Song Chen. Changing background to foreground: An augmentation method based on conditional generative network for stingray detection. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 2740–2744. IEEE, 2018.

[29] Marthinus Christoffel, Gang Niu, and Masashi Sugiyama. Class-prior Estimation for Learning from Positive and Unlabeled Data. In *Asian Conference on Machine Learning*, pages 221–236, 2016.

[30] Laurène Claussmann, Marc Revilloud, Dominique Gruyer, and Sébastien Glaser. A review of motion planning for highway autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[31] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudk, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 215–223, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR.

[32] 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 *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[33] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[34] Ryan Dahl, Mohammad Norouzi, and Jonathon Shlens. Pixel recursive super resolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5439–5448, 2017.

[35] Hendrik Dahlkamp, Adrian Kaehler, David Stavens, Sebastian Thrun, and Gary R. Bradski. Self-supervised monocular road detection in desert terrain. In *Robotics: science and systems*, volume 38. Philadelphia, 2006.

[36] Abdelkader Dairi, Fouzi Harrou, Mohamed Senouci, and Ying Sun. Unsupervised obstacle detection in driving environments using deep-learning-based stereovision. *Robotics and Autonomous Systems*, 100:287–301, 2018.

[37] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):1052–1067, 2007.

[38] Lucas de Carvalho Pagliosa and Rodrigo Fernandes de Mello. Semi-supervised time series classification on positive and unlabeled problems using cross-recurrence quantification analysis. *Pattern Recognition*, 80:53–63, 2018.

[39] Franois Denis. PAC learning from positive statistical queries. In *International Conference on Algorithmic Learning Theory*, pages 112–126. Springer, 1998.

[40] Julie Dequaire, Peter Ondruska, Dushyant Rao, Dominic Wang, and Ingmar Posner. Deep tracking in the wild: End-to-end tracking using recurrent neural networks. *The International Journal of Robotics Research*, page 0278364917710543, 2017.

[41] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[42] PN Druzhkov and VD Kustikova. A survey of deep learning methods and software tools for image classification and object detection. *Pattern Recognition and Image Analysis*, 26(1):9–15, 2016.

[43] Marthinus Du Plessis, Gang Niu, and Masashi Sugiyama. Convex formulation for learning from positive and unlabeled data. In *International Conference on Machine Learning*, pages 1386–1394, 2015.

[44] Marthinus C. du Plessis, Gang Niu, and Masashi Sugiyama. Analysis of learning from positive and unlabeled data. In *Advances in neural information processing systems*, pages 703–711, 2014.

[45] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014.

[46] Rajmadhan Ekambaram, Sergiy Fefilatyev, Matthew Shreve, Kurt Kramer, Lawrence O. Hall, Dmitry B. Goldgof, and Rangachar Kasturi. Active cleaning of label noise. *Pattern Recognition*, 51:463–480, 2016.

[47] Charles Elkan and Keith Noto. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 213–220. ACM, 2008.

[48] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2012.

[49] Gunnar Farnebck. Two-frame motion estimation based on polynomial expansion. *Image analysis*, pages 363–370, 2003.

[50] Farzan Farnia, Jesse Zhang, and David Tse. Generalizable adversarial training via spectral normalization. In *International Conference on Learning Representations*, 2019.

[51] Mona Fathollahi and Rangachar Kasturi. Autonomous driving challenge: To Infer the property of a dynamic object based on its motion pattern using recurrent neural network. *arXiv preprint arXiv:1609.00361*, 2016.

[52] 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.

[53] Nir Friedman and Stuart Russell. Image segmentation in video sequences: A probabilistic approach. In *Proceedings of the Thirteenth conference on Uncertainty in artificial intelligence*, pages 175–181. Morgan Kaufmann Publishers Inc., 1997.

[54] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2002–2011, 2018.

[55] Kunihiko Fukushima, Sei Miyake, and Takayuki Ito. Neocognitron: A neural network model for a mechanism of visual pattern recognition. *IEEE transactions on systems, man, and cybernetics*, (5):826–834, 1983.

[56] Yarin Gal. *Uncertainty in Deep Learning*. PhD thesis, PhD thesis, University of Cambridge, 2016.

[57] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, Pablo Martinez-Gonzalez, and Jose Garcia-Rodriguez. A survey on deep learning techniques for image and video semantic segmentation. *Applied Soft Computing*, 70:41–65, 2018.

[58] Ravi Garg, Vijay Kumar BG, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European Conference on Computer Vision*, pages 740–756. Springer, 2016.

[59] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012.

[60] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[61] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[62] Clment Godard, Oisin Mac Aodha, and Gabriel Brostow. Digging into self-supervised monocular depth estimation. *arXiv preprint arXiv:1806.01260*, 2018.

[63] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

[64] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[65] Vitor Guizilini and Fabio Ramos. Online self-supervised segmentation of dynamic objects. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 4720–4727. IEEE, 2013.

[66] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5769–5779, 2017.

[67] Raia Hadsell, Pierre Sermanet, Jan Ben, Ayse Erkan, Marco Scoffier, Koray Kavukcuoglu, Urs Muller, and Yann LeCun. Learning long-range vision for autonomous off-road driving. *Journal of Field Robotics*, 26(2):120–144, 2009.

[68] Sid Ali Hamideche, Florent Chiaroni, and Mohamed-Cherif Rahal. Self-supervised classification of dynamic obstacles using the temporal information provided by videos. *arXiv preprint arXiv:1910.09094*, 2019.

[69] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143, 2015.

[70] Christopher G Harris, Mike Stephens, et al. A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Citeseer, 1988.

[71] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[72] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[73] Nicholas Heller, Joshua Dean, and Nikolaos Papanikolopoulos. Imperfect segmentation labels: How much do they matter? In *Intravascular Imaging and Computer Assisted Stenting and Large-Scale Annotation of Biomedical Data and Expert Label Synthesis*, pages 112–120. Springer, 2018.

[74] Ming Hou, Brahim Chaib-Draa, Chao Li, and Qibin Zhao. Generative adversarial positive-unlabeled learning. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, pages 2255–2261. AAAI Press, 2018.

[75] Michael E. Houle. Local intrinsic dimensionality I: an extreme-value-theoretic foundation for similarity applications. In *International Conference on Similarity Search and Applications*, pages 64–79. Springer, 2017.

[76] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106–154, 1962.

[77] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[78] Shantanu Jain, Martha White, and Predrag Radivojac. Estimating the class prior and posterior from noisy positives and unlabeled data. In *Advances in Neural Information Processing Systems 29*, pages 2693–2701. Curran Associates, Inc., 2016.

[79] Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9865–9874, 2019.

[80] Longlong Jing and Yingli Tian. Self-supervised Visual Feature Learning with Deep Neural Networks: A Survey. *arXiv preprint arXiv:1902.06162*, 2019.

[81] Christoph Käding, Erik Rodner, Alexander Freytag, and Joachim Denzler. Fine-tuning deep neural networks in continuous learning scenarios. In *Asian Conference on Computer Vision*, pages 588–605. Springer, 2016.

[82] Hirotaka Kaji and Masashi Sugiyama. Binary classification only from unlabeled data by iterative unlabeled-unlabeled classification. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3527–3531. IEEE, 2019.

[83] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1409–1422, July 2012.

[84] Maxim Karpushin, Giuseppe Valenzise, and Frdric Dufaux. Keypoint detection in rgbd images based on an anisotropic scale space. *IEEE Transactions on Multimedia*, 18(9):1762–1771, 2016.

[85] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018.

[86] Alex Kendall, Vijay Badrinarayanan, and Roberto Cipolla. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *arXiv preprint arXiv:1511.02680*, 2015.

[87] Shehroz S. Khan and Michael G. Madden. One-class classification: taxonomy of study and review of techniques. *The Knowledge Engineering Review*, 29(3):345–374, 2014.

[88] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[89] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollr. Panoptic segmentation. *arXiv preprint arXiv:1801.00868*, 2018.

[90] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, and Agnieszka Grabska-Barwinska. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, page 201611835, 2017.

[91] Ryuichi Kiryo, Gang Niu, Marthinus C du Plessis, and Masashi Sugiyama. Positive-Unlabeled Learning with Non-Negative Risk Estimator. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 1675–1685. Curran Associates, Inc., 2017.

[92] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

[93] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[94] Jaya S Kulchandani and Kruti J Dangarwala. Moving object detection: Review of recent research trends. In *2015 International Conference on Pervasive Computing (ICPC)*, pages 1–5. IEEE, 2015.

[95] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[96] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[97] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[98] Timothée Lesort, Hugo Caselles-Dupré, Michael Garcia-Ortiz, Jean-François Goudou, and David Filliat. Generative Models from the perspective of Continual Learning. In *Workshop on Continual Learning, NeurIPS 2018 - Thirty-second Conference on Neural Information Processing Systems*, Montréal, Canada, December 2018.

[99] Mei Li, Shirui Pan, Yang Zhang, and Xiaoyan Cai. Classifying networked text data with positive and unlabeled examples. *Pattern Recognition Letters*, 77:1–7, 2016.

[100] David Lieb, Andrew Lookingbill, and Sebastian Thrun. Adaptive Road Following using Self-Supervised Learning and Reverse Optical Flow. In *Robotics: science and systems*, pages 273–280, 2005.

[101] Patrick Lin. Why ethics matters for autonomous cars. In *Autonomous driving*, pages 69–85. Springer, Berlin, Heidelberg, 2016.

[102] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[103] Bing Liu, Wee Sun Lee, Philip S. Yu, and Xiaoli Li. Partially supervised classification of text documents. In *International Conference on Machine Learning*, volume 2, pages 387–394. Citeseer, 2002.

[104] F. Liu, C. Shen, G. Lin, and I. Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(10):2024–2039, Oct 2016.

[105] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *arXiv preprint arXiv:1809.02165*, 2018.

[106] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot multibox detector. In *European Conference on Computer Vision*, pages 21–37. Springer, 2016.

[107] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[108] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[109] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[110] Pauline Luc, Camille Couprie, Soumith Chintala, and Jakob Verbeek. Semantic segmentation using adversarial networks. *arXiv preprint arXiv:1611.08408*, 2016.

[111] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. 1981.

[112] Xingjun Ma, Yisen Wang, Michael E. Houle, Shuo Zhou, Sarah M. Erfani, Shu-Tao Xia, Sudanthi Wijewickrema, and James Bailey. Dimensionality-driven learning with noisy labels. *arXiv preprint arXiv:1806.02612*, 2018.

[113] Will Maddern, Geoff Pascoe, Chris Linegar, and Paul Newman. 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research (IJRR)*, 36(1):3–15, 2017.

[114] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2017.

[115] Mohamed Amine Marnissi, Hajer Fradi, and Jean-Luc Dugelay. On the discriminative power of learned vs. hand-crafted features for crowd density analysis. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[116] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error, 2015.

[117] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3061–3070, 2015.

[118] Erinc Merdivan, Mohammad Reza Loghmani, and Matthieu Geist. Reconstruct & crush network. In *Advances in Neural Information Processing Systems*, pages 4548–4556, 2017.

[119] Anton Milan, S. Hamid Rezatofighi, Anthony Dick, Ian Reid, and Konrad Schindler. Online multi-target tracking using recurrent neural networks. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[120] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

[121] Takeru Miyato and Masanori Koyama. cGANs with projection discriminator. In *International Conference on Learning Representations*, 2018.

[122] Todd K. Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60, 1996.

[123] Fantine Mordelet and J.-P. Vert. A bagging SVM to learn from positive and unlabeled examples. *Pattern Recognition Letters*, 37:201–209, 2014.

[124] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with Noisy Labels. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 1196–1204. Curran Associates, Inc., 2013.

[125] Radford M Neal and Geoffrey E Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. Springer, 1998.

[126] Gang Niu, Marthinus Christoffel du Plessis, Tomoya Sakai, Yao Ma, and Masashi Sugiyama. Theoretical comparisons of positive-unlabeled learning against positive-negative learning. In *Advances in Neural Information Processing Systems*, pages 1199–1207, 2016.

[127] Curtis G. Northcutt, Tailin Wu, and Isaac L. Chuang. Learning with Confident Examples: Rank Pruning for Robust Classification with Noisy Labels. In *Uncertainty in Artificial Intelligence (UAI)*, 2017.

[128] Peter Ondruska and Ingmar Posner. Deep tracking: Seeing beyond seeing using recurrent neural networks. *arXiv preprint arXiv:1602.00991*, 2016.

[129] Deepak Pathak, Ross Girshick, Piotr Dollr, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2701–2710, 2017.

[130] Anh Pham. *Weak-supervision: Probabilistic Models and Inference*. PhD thesis, 2018.

[131] Marco AF Pimentel, David A. Clifton, Lei Clifton, and Lionel Tarassenko. A review of novelty detection. *Signal Processing*, 99:215–249, 2014.

[132] Tristan Postadjian, Arnaud Le Bris, Hichem Sahbi, and Clément Mallet. Investigating the potential of deep neural networks for large-scale classification of very high resolution satellite images. *ISPRS Annals*, 4:183–190, 2017.

[133] Gowdham Prabhakar, Binsu Kailath, Sudha Natarajan, and Rajesh Kumar. Obstacle detection and classification using deep learning for tracking in high-speed autonomous driving. In *2017 IEEE Region 10 Symposium (TENSYMP)*, pages 1–6. IEEE, 2017.

[134] K Prazdny. Egomotion and relative depth map from optical flow. *Biological cybernetics*, 36(2):87–102, 1980.

[135] Guo-Jun Qi. Loss-sensitive generative adversarial networks on lipschitz densities. *arXiv preprint arXiv:1701.06264*, 2017.

[136] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[137] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *Advances in Neural Information Processing Systems*, pages 3546–3554, 2015.

[138] Carl Edward Rasmussen. The infinite gaussian mixture model. In *Advances in neural information processing systems*, pages 554–560, 2000.

[139] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, pages 525–542. Springer, 2016.

[140] Sebastien Razakarivony and Frederic Jurie. Vehicle detection in aerial imagery: A small target detection benchmark. *Journal of Visual Communication and Image Representation*, 34:187–203, 2016.

[141] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[142] Joseph Redmon and Ali Farhadi. YOLO9000: Better, Faster, Stronger. *arXiv preprint arXiv:1612.08242*, 2016.

[143] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[144] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[145] Angie K Reyes, Juan C Caicedo, and Jorge E Camargo. Fine-tuning deep convolutional networks for plant recognition. *CLEF (Working Notes)*, 1391, 2015.

[146] Violeta Roizman, Matthieu Jonckheere, and Frédéric Pascal. A flexible em-like clustering algorithm for noisy data. *arXiv preprint arXiv:1907.01660*, 2019.

[147] Henry Roncancio, Marcelo Becker, Alberto Broggi, and Stefano Cattani. Traversability analysis using terrain mapping and online-trained terrain type classifier. In *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pages 1239–1244. IEEE, 2014.

[148] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[149] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary R Bradski. Orb: An efficient alternative to sift or surf. In *ICCV*, volume 11, page 2. Citeseer, 2011.

[150] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[151] Renato F. Salas-Moreno, Richard A. Newcombe, Hauke Strasdat, Paul H.J. Kelly, and Andrew J. Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[152] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

[153] Emanuele Sansone, Francesco GB De Natale, and Zhi-Hua Zhou. Efficient training for positive unlabeled learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[154] Ashutosh Saxena, Min Sun, and Andrew Y Ng. Make3d: Learning 3d scene structure from a single still image. *IEEE transactions on pattern analysis and machine intelligence*, 31(5):824–840, 2008.

[155] Davide Scaramuzza and Friedrich Fraundorfer. Visual odometry [tutorial]. *IEEE robotics & automation magazine*, 18(4):80–92, 2011.

[156] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[157] Hoo-Chang Shin, Holger R Roth, Mingchen Gao, Le Lu, Ziyue Xu, Isabella Nogues, Jianhua Yao, Daniel Mollura, and Ronald M Summers. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging*, 35(5):1285–1298, 2016.

[158] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[159] Paul Sturgess, Karteek Alahari, Lubor Ladicky, and Philip HS Torr. Combining appearance and structure from motion features for road scene understanding. 2009.

[160] Sainbayar Sukhbaatar and Rob Fergus. Learning from noisy labels with deep neural networks. *arXiv preprint arXiv:1406.2080*, 2(3):4, 2014.

[161] Yap-Peng Tan, Drew D Saur, Sanjeev R Kulkami, and Peter J Ramadge. Rapid estimation of camera motion from compressed video with application to video annotation. *IEEE Transactions on Circuits and Systems for Video Technology*, 10(1):133–146, 2000.

[162] Marvin Teichmann, Michael Weber, Marius Zoellner, Roberto Cipolla, and Raquel Urtasun. MultiNet: Real-time Joint Semantic Reasoning for Autonomous Driving. *arXiv preprint arXiv:1612.07695*, 2016.

[163] Kiran K Thekumparampil, Ashish Khetan, Zinan Lin, and Sewoong Oh. Robustness of conditional gans to noisy labels. In *Advances in Neural Information Processing Systems*, pages 10271–10282, 2018.

[164] Gill Ward, Trevor Hastie, Simon Barry, Jane Elith, and John R. Leathwick. Presence-only data and the EM algorithm. *Biometrics*, 65(2):554–563, 2009.

[165] D. Randall Wilson and Tony R. Martinez. The general inefficiency of batch training for gradient descent learning. *Neural Networks*, 16(10):1429–1451, 2003.

[166] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[167] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2691–2699, 2015.

[168] Yuchuan Xu, Chunhai Gao, Lei Yuan, Simon Tang, and Guodong Wei. Real-time obstacle detection over rails using deep convolutional neural network. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 1007–1012. IEEE, 2019.

[169] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[170] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2018.

[171] Cong Zhang, Hongsheng Li, Xiaogang Wang, and Xiaokang Yang. Cross-scene crowd counting via deep convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[172] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7354–7363, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[173] Hongyang Zhang, Susu Xu, Jiantao Jiao, Pengtao Xie, Ruslan Salakhutdinov, and Eric P Xing. Stackelberg gan: Towards provable minimax equilibrium via multi-generator architectures. *arXiv preprint arXiv:1811.08010*, 2018.

[174] Jiaqi Zhang, Zhenzhen Wang, Jingjing Meng, Yap-Peng Tan, and Junsong Yuan. Boosting positive and unlabeled learning for anomaly detection with multi-features. *IEEE Transactions on Multimedia*, 21(5):1332–1344, 2018.

[175] Jiaqi Zhang, Zhenzhen Wang, Junsong Yuan, and Yap-Peng Tan. Positive and unlabeled learning for anomaly detection with multi-features. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 854–862. ACM, 2017.

[176] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.

[177] Yiran Zhong, Yuchao Dai, and Hongdong Li. Self-supervised learning for stereo matching with self-improving ability. *arXiv preprint arXiv:1709.00930*, 2017.

[178] Yu Zhong. Intrinsic shape signatures: A shape descriptor for 3d object recognition. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pages 689–696. IEEE, 2009.

[179] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127(3):302–321, 2019.

[180] Shengyan Zhou, Jianwei Gong, Guangming Xiong, Huiyan Chen, and Karl Iagnemma. Road detection using support vector machine based on online learning and evaluation. In *2010 IEEE Intelligent Vehicles Symposium*, pages 256–261. IEEE, 2010.

[181] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1851–1858, 2017.

[182] Zoran Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 2, pages 28–31. IEEE, 2004.

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Apprentissage faiblement supervisé pour la classification d'images et l'analyse des obstacles potentiellement mobiles

Mots clés: Véhicule autonome, Vision par ordinateur, Apprentissage automatique, Réseaux de neurones artificiels

Résumé: Dans le contexte des applications de perception pour le véhicule à conduite déléguée, l'intérêt pour les approches d'apprentissage automatique a continuellement augmenté pendant cette dernière décennie. Cependant, lorsque ces approches doivent être discriminatives, elle nécessitent généralement d'apprendre sur des données manuellement annotées. L'annotation manuelle a un coût non négligeable, tandis que les données non annotées peuvent être facilement obtenues dans le contexte d'un véhicule autonome équipé de capteurs. Il se trouve qu'une catégorie de stratégies d'apprentissage, dite d'apprentissage faiblement supervisé, permet d'exploiter des données partiellement labélisées. Ainsi, nous avons pour objectif dans cette thèse de réduire autant que possible le besoin de labélisation manuelle en proposant des techniques d'apprentissage faiblement supervisées.

Title: Weakly supervised learning for image classification and potentially moving obstacles analysis

Keywords: Self-driving car, Computer vision, Machine learning, Artificial Neural Networks

Abstract: In the context of autonomous vehicle perception, the interest of the research community for deep learning approaches has continuously grown since the last decade. This can be explained by the fact that deep learning techniques provide nowadays state-of-the-art prediction performances for several computer vision challenges. More specifically, deep learning techniques can provide rich semantic information concerning the complex visual patterns encountered in autonomous driving scenarios. However, such approaches require, as their name implies, to learn on data. In particular, state-of-the-art prediction performances on discriminative tasks often demand hand labeled data of the target application domain. Hand labeling has a significant cost, while, conversely, unlabeled data can be easily obtained in the autonomous driving context. It turns out that a category of learning strategies, referred to as weakly supervised learning, enables to exploit partially labeled data. Therefore, we aim in this thesis at reducing as much as possible the hand labeling requirement by proposing weakly supervised learning techniques.

We start by presenting a type of learning methods which are self-supervised. They consist of substituting hand-labels by upstream techniques able to automatically generate exploitable training labels. Self-supervised learning (SSL) techniques have proven their usefulness in the past for offroad obstacles avoidance and path planning through changing environments. However, SSL techniques still leave the door open for detection, segmentation, and classification of static potentially moving obstacles.

Consequently, we propose in this thesis three novel weakly supervised learning methods with the final goal to deal with such road users through an SSL framework. The first two proposed contributions of this work aim at dealing with partially labeled image classification datasets, such that the labeling effort can be only focused on our class of interest, the positive class. Then, we propose an approach which deals with training data containing a high fraction of wrong labels, referred to as noisy labels. Next, we demonstrate the potential of such weakly supervised strategies for detection and segmentation of potentially moving obstacles.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France