Nesigurnost segmentacije slike s obzirom na uzorke izvan distribucije

Runtas, Jurica

UNIVERSITY OF ZAGREB
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

BACHELOR THESIS No. 1363

UNCERTAINTY OF IMAGE SEGMENTATION REGARDING OUT-OF-DISTRIBUTION SAMPLES

Jurica Runtas

Zagreb, June 2024

UNIVERSITY OF ZAGREB
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

BACHELOR THESIS No. 1363

UNCERTAINTY OF IMAGE SEGMENTATION REGARDING OUT-OF-DISTRIBUTION SAMPLES

Jurica Runtas

Zagreb, June 2024

UNIVERSITY OF ZAGREB
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

Zagreb, 04 March 2024

BACHELOR THESIS ASSIGNMENT No. 1363

Student:	**Jurica Runtas (0036538794)**
Study:	Electrical Engineering and Information Technology and Computing
Module:	Computing
Mentor:	assoc. prof. Tomislav Petković

Title: **Uncertainty of Image Segmentation Regarding Out-of-distribution Samples**

Description:

In image segmentation one of the predetermined classes is assigned to each pixel in the image. Deep neural networks (DNNs) are a contemporary solution to the segmentation problem and commonly use normalized exponential functions (softmax) in the output layer. A dataset used to train DNNs typically contains only examples of the classes that must be recognized, however, in practice input images may contain many unknown classes that are out-of-distribution samples. Recognizing such inputs is an open problem. In the thesis a brief overview of DNNs which are used for segmentation shall be provided with a special emphasis on DNN architectures that can assess the (un)certainty of the segmentation and that can recognize out-of-distribution samples. Then, one existing DNN which is used in traffic applications and which has a softmax output layer shall be selected, and a new or an improved solution for determining the uncertainty of segmentation shall be proposed, or, alternatively, a new or an improved solution for recognizing out-of-distribution inputs shall be proposed. The proposed solution shall be quantitatively evaluated using two separate datasets which contain expected and unexpected classes, e.g., for traffic applications the Cityscapes dataset contains expected in-distribution classes and COCO contains unexpected out-of-distribution classes.

Submission date: 14 June 2024

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

ZAVRŠNI ZADATAK br. 1363

Pristupnik:	**Jurica Runtas (0036538794)**
Studij:	Elektrotehnika i informacijska tehnologija i Računarstvo
Modul:	Računarstvo
Mentor:	izv. prof. dr. sc. Tomislav Petković

Zadatak: **Nesigurnost segmentacije slike s obzirom na uzorke izvan distribucije**

Opis zadatka:

Kod segmentacije slike svakom pikselu slike pridružujemo jednu od unaprijed zadanih klasa. Duboke neuronske mreže (DNM) su suvremeno rješenje segmentacijskog problema i uobičajeno koriste normalizirane eksponencijalne funkcije (softmax) u izlaznom sloju. Pri učenju DNM-a koristi se konačan skup podataka koji tipično sadrži samo primjere onih klasa koje je potrebno prepoznati, dok u primjeni ulazna slika može sadržavati nepoznate klase koje se nalaze izvan distribucije naučenih klasa. Prepoznavanje takvih ulaza je otvoren problem. U završnom radu je potrebno dati kratki pregled DNM-a koje se koriste za segmentaciju s posebnim naglaskom na arhitekture koje mogu procijeniti (ne)sigurnost segmentacije i koje mogu prepoznati uzorke izvan distribucije. Zatim je potrebno odabrati neku postojeću DNM koja se koristi u prometu i koja koristi softmax, te je za nju potrebno predložiti novo ili poboljšati postojeće rješenje za određivanje nesigurnosti segmentacije ili za prepoznavanje ulaznih uzoraka koji su izvan distribucije. Predloženo rješenje je potrebno kvantitativno ispitati korištenjem dva skupa podataka koji sadrže očekivane i neočekivane klase, npr. za primjene u prometu skup podataka Cityscapes sadrži očekivane klase koje su unutar distribucije, a COCO one neočekivane izvan distribucije.

Rok za predaju rada: 14. lipnja 2024.

This page intentionally left blank

Contents

1 Introduction

Semantic segmentation is a computer vision task in which each pixel of an image is assigned into one of the predefined classes. Examples of a real-world application are autonomous driving systems where semantic segmentation is an important component for visual perception of a driving environment [1, 2, 3].

Deep neural networks (DNNs) are a contemporary solution to the semantic segmentation task. They are usually trained to operate on a predefined closed set of classes. However, this is in a contradiction with the nature of an environment in which aforementioned autonomous driving systems are deployed. Such systems operate in so-called open-set environments where DNNs can encounter *anomalies*, i.e., objects that do not belong to any class from the predefined closed set of classes used during training [4]. Scenes, i.e., images that contain anomalies are commonly referred to as *out-of-distribution (OoD)* samples while anomalies are also known as OoD objects [5]. From a safety standpoint, it is of the utmost importance to ensure that when DNN encounters an anomaly, it is able to classify its pixels as anomalous. A presence of anomaly indicates that DNN is operating outside of its learned domain and further action should be taken, e.g., there is an unknown object on the road, emergency braking procedure is initiated.

In this thesis, we explore the method for detecting, i.e., segmenting anomalies called *entropy maximization*. Such method is combined with a post-processing step called *meta-classification* in order to further improve the reliability of detecting anomalies. We propose a modification to the meta classification which significantly improves the reliability of detecting anomalies in comparison to the baseline as shown by our experimental results. Furthermore, we provide additional analysis of the entropy maximization which shows that caution must be taken when using it in practice in order to ensure its effectiveness.

2 Semantic Segmentation

Artificial neural networks (ANNs) are computational systems inspired by biological neural networks. On a fundamental level, ANNs are constructed from units of computation called neurons which perform nonlinear transformation of a weighted sum of their inputs. Neurons are organized into layers such that the output of one layer serves as the input to the next layer. Deep neural networks (DNNs) consist of a large number of such layers which enables them to solve complex problems. From a strictly mathematical standpoint, neural networks can be interpreted as *universal function approximators* which can approximate a nonlinear input-output mapping of a general nature [6]. This interpretation could explain why neural networks can be used as a solution to many problems from various domains. State-of-the-art semantic segmentation solutions rely on neural networks and their ability to learn complex features from input data during training. Learned knowledge is then used to extrapolate features from a new previously unseen data during inference [6].

Let $\mathbf{x} \in [0,1]^{H \times W \times 3}$ denote a normalized color image of spatial dimensions $H \times W$, where $H \in \mathbb{N}$ and $W \in \mathbb{N}$. Let $\mathcal{I} = \{1, 2, ..., H\} \times \{1, 2, ..., W\}$ denote the set of pixel locations. Let $\mathcal{C} = \{1, 2, ..., C\}$ denote the set of $C \in \mathbb{N}$ predefined classes.

For each image $\mathbf{x}$ and for a given $\mathcal{C}$, there exists $\mathbf{g} = (g_i)_{i \in \mathcal{I}} \in \mathcal{C}^{H \times W}$, called an *inherent segmentation mask*. Let $\mathbf{G} : [0,1]^{H \times W \times 3} \to \mathcal{C}^{H \times W}$ denote mapping such that $\mathbf{g} = \mathbf{G}(\mathbf{x})$. In practice, for each image $\mathbf{x}$ used to train a neural network for semantic segmentation and for a given $\mathcal{C}$, there exists $\mathbf{m} = (m_i)_{i \in \mathcal{I}} \in \mathcal{C}^{H \times W}$, often referred to as the *ground truth segmentation mask*, obtained through an image labeling process where a human, i.e., an expert, manually labels semantic segments in $\mathbf{x}$. Let $\mathbf{M} : [0,1]^{H \times W \times 3} \to \mathcal{C}^{H \times W}$ denote image labeling process such that $\mathbf{m} = \mathbf{M}(\mathbf{x})$.

Let $\mathcal{E}$ denote the set of pixel locations of incorrectly labeled pixels of an image $\mathbf{x}$ during $\mathbf{M}$, i.e., $\mathcal{E} = \{i \mid m_i \neq g_i, m_i \in \mathbf{M}(\mathbf{x}), g_i \in \mathbf{G}(\mathbf{x}), i \in \mathcal{I}\}$. It should be apparent that for a given image $\mathbf{x}$, $\mathbf{M}(\mathbf{x}) \approx \mathbf{G}(\mathbf{x})$ in the sense that $\mathcal{E} \neq \emptyset$. Incorrectly labeled pixels are mostly present at the boundaries of semantic segments. This can be attributed to the fact that labeling interior pixels is easier than labeling boundary pixels. Obviously, the goal of an image labeling process is to reduce the number of incorrectly labeled pixels, i.e., it is desirable that $|\mathcal{E}| \to 0$, because it tends to positively impact network's generalization ability [7].

We define the set of training data used to train a neural network in a supervised manner as

$$\mathcal{D}_{in}^{train} = \{(\mathbf{x}_j, \mathbf{m}_j) \mid \mathbf{m}_j = \mathbf{M}(\mathbf{x}_j)\}_{j=1}^{N_{in}^{train}} \tag{2.1}$$

where N_{in}^{train} denotes the total number of training samples. They are often referred to as *in-distribution training samples*.

Let $\mathbf{F} : [0,1]^{H \times W \times 3} \to \mathbb{R}^{H \times W \times |\mathcal{C}|}$ be a neural network for semantic segmentation with learnable parameters Θ, i.e., weights that produces pixel-wise class scores $\mathbf{y} = \mathbf{F}(\mathbf{x}; \Theta)$ such that $\mathbf{y} = (y_{i,c})_{i \in \mathcal{I}, c \in \mathcal{C}} \in \mathbb{R}^{H \times W \times |\mathcal{C}|}$. Usually, a pixel-wise softmax function defined as

$$\sigma(\mathbf{y})_{i,c} = \frac{e^{y_{i,c}}}{\sum_{k=1}^{|\mathcal{C}|} e^{y_{i,k}}}, \ \forall i \in \mathcal{I}, \forall c \in \mathcal{C} \tag{2.2}$$

is applied on $\mathbf{y}$ to map pixel-wise class scores to pixel-wise class probabilities so that the final result can be used as an input to a loss function such as a commonly used cross-entropy loss [8]. Furthermore, so-called softmax probabilities often serve as a basis for the state-of-the-art anomaly segmentation methods [9].

The predicted segmentation mask is given by $\widetilde{\mathbf{m}} = (\tilde{m}_i)_{i \in \mathcal{I}} \in \mathcal{C}^{H \times W}$, where $\tilde{m}_i = \arg\max_{c \in \mathcal{C}} y_{i,c}, \forall i \in \mathcal{I}$, which is also equivalent to $\tilde{m}_i = \arg\max_{c \in \mathcal{C}} \sigma(\mathbf{y})_{i,c}, \forall i \in \mathcal{I}$. The equivalence comes from the monotonicity of the softmax function [10].

When training a neural network for semantic segmentation with samples from $\mathcal{D}_{in}^{train}$, the main objective is that the network learns parameters Θ such that the predicted segmentation mask $\widetilde{\mathbf{m}}_j$ closely matches the ground truth segmentation mask $\mathbf{m}_j$ for a given in-distribution training sample $(\mathbf{x}_j, \mathbf{m}_j) \in \mathcal{D}_{in}^{train}$. Moreover, we want to train the net-

work in such manner that after the training process finishes, learned parameters Θ provide the network with a satisfactory amount of generalization ability, i.e., during inference, the network should be able to perform well on $(\mathbf{x}^*, \mathbf{m}^*) \notin \mathcal{D}_{in}^{train}$. Let $\mathcal{D}_{in}^{test}$ denote the set of an *in-distribution test samples* such that $\mathcal{D}_{in}^{train} \cap \mathcal{D}_{in}^{test} = \emptyset$. If the network's generalization ability is being evaluated on $(\mathbf{x}^*, \mathbf{m}^*) \in \mathcal{D}_{in}^{test}$ then $\mathbf{m}^* = \mathbf{M}(\mathbf{x}^*)$. If $(\mathbf{x}^*, \mathbf{m}^*) \notin \mathcal{D}_{in}^{train} \cup \mathcal{D}_{in}^{test}$ then it is assumed that the sample comes from the real-world environment in which the network is deployed, therefore, $\mathbf{m}^*$ is undefined due to the absence of an image labeling process $\mathbf{M}$. Regardless, since the network is not learning parameters Θ during inference, $\mathbf{m}^*$ is irrelevant from the network's perspective. It is assumed that the $\mathcal{D}_{in}^{train}$ contains enough representative samples so that even if neural network is presented with an input image that has not been seen during training, it can still correctly classify each pixel with an acceptable amount of accuracy [8].

Standard closed-set neural networks for semantic segmentation assume that samples $(\mathbf{x}^*, \mathbf{m}^*) \notin \mathcal{D}_{in}^{train}$ seen during inference come from an *in-distribution*, i.e., $(\mathbf{x}^*, \mathbf{m}^*) \in \mathcal{D}_{in}$. It should be noted that $\mathcal{D}_{in}^{train} \subset \mathcal{D}_{in}$ and $\mathcal{D}_{in}^{test} \subset \mathcal{D}_{in}$, where $\subset$ represents a proper subset. This assumption usually means that the network expects objects in the input image to belong to one of the classes contained exclusively in $\mathcal{C}$.

However, the assumption that the samples during inference come exclusively from $\mathcal{D}_{in}$ is too strong for real-world applications such as autonomous driving. Methods that relax this assumption are needed in order to make neural networks for semantic segmentation applicable to the real-world environments.

3 Anomaly Segmentation

In the context of semantic segmentation, *anomalies* or *out-of-distribution* (OoD) objects are image segments, or even entire images, whose pixels lie at a location of extreme low density of $\mathbf{p}(\mathbf{x})$, the underlying probability distribution over images in $\mathcal{D}_{in}$, or even outside of it [5]. Let $\mathcal{D}_{out}$ denote a set of images containing such OoD objects. The task of identifying semantically anomalous regions in an image is called *anomaly segmentation* or in the more general context, *OoD detection*.

3.1 Methods for Anomaly Segmentation

Regardless of a specific method used for anomaly segmentation, the main objective is to obtain an *anomaly segmentation score map* $\mathbf{a} = (a_i)_{i \in \mathcal{I}} \in \mathbb{R}^{H \times W}$ which indicates the presence of an anomaly at each pixel location of a given image where the higher the score the more likely is that there is anomaly at that pixel location [5]. The methods differ in the ways they obtain such a map.

The first type of methods utilize thresholding on the maximum softmax probability (MSP) obtained from the output of a neural network on every pixel location. The reasoning behind this method is that the OoD objects and their corresponding pixels tend to have lower MSP than in-distribution pixels, allowing for their detection [9, 11]. Another example of the utilization of the softmax probabilities for detecting OoD pixels are methods based on the information theory standpoint, i.e., a desirable property of a semantic segmentation network would be the attachment of a high prediction uncertainty to the OoD objects which can in turn be quantified with a per-pixel prediction entropy [5, 12]. Another approach is utilization of negative datasets as a proxy for OoD objects and training a separate model prediction head to directly predict the probability of a pixel being OoD [13, 14, 15].

The second type of methods for anomaly segmentation are based on uncertainty estimation through the usage of Bayesian methods such as Monte-Carlo dropout [16, 17] or the usage of an ensemble of models [18] which capture the model uncertainty by averaging predictions over multiple models. Another type of methods are based on obtaining an anomaly segmentation score map by the means of computing some sort of a statistical distance such as Mahalanobis distance [19, 20]. There are also methods which use Radial Basis Function Networks (RBFNs) in order to learn a feature space using RBF-kernels so that the feature distance to the nearest center quantifies the uncertainty of a prediction, i.e., OoD object [21, 22].

Finally, some methods use generative models for the purpose of anomaly segmentation. The main idea is that the generative model trained exclusively on in-distribution training samples will have less accurate performance on OoD regions in images, consequently, the detection of OoD objects is based on the reconstruction quality of an input image [2, 23, 24].

3.2 Anomaly Segmentation via Entropy Maximization

In this thesis, the focus is on the method that utilizes the per-pixel prediction entropy in order to compute an anomaly segmentation score map $\mathbf{a}$. The methods described in this and the following section are introduced and thoroughly described in [5, 12, 25, 26, 27].

Let $\mathbf{p}_i(\mathbf{x}) = \big(p_i(c|\mathbf{x})\big)_{i\in\mathcal{I},c\in\mathcal{C}} \in [0,1]^{|\mathcal{C}|}$ denote a vector of probabilities such that the $p_i(c|\mathbf{x})$ is a probability of a pixel location $i \in \mathcal{I}$ of a given image $\mathbf{x} \in \mathcal{D}_{in}$ being a pixel that belongs to the class $c \in \mathcal{C}$. For the sake of clarity, it should be noted that if $\mathbf{y} = \mathbf{F}(\mathbf{x})$ represents an output of a semantic segmentation network $\mathbf{F}$, then $p_i(c|\mathbf{x}) = \sigma(\mathbf{y})_{i,c}$, cf. Eq. (2.2). We define $\mathbf{p}(\mathbf{x}) = \big(\mathbf{p}_i(\mathbf{x})\big)_{i\in\mathcal{I}} \in [0,1]^{H\times W\times|\mathcal{C}|}$, the probability distribution over images in $\mathcal{D}_{in}$. When using $\mathcal{D}_{in}^{train}$ to train a semantic segmentation network $\mathbf{F}$, one can interpret that the network is being trained to estimate $\mathbf{p}(\mathbf{x})$, denoted by $\hat{\mathbf{p}}(\mathbf{x})$.

For a given image $\mathbf{x} \in [0,1]^{H\times W\times 3}$ and a pixel location $i \in \mathcal{I}$, the per-pixel prediction entropy is defined as

$$E_i\big(\hat{\mathbf{p}}_i(\mathbf{x})\big) = -\sum_{c\in\mathcal{C}} \hat{p}_i(c|\mathbf{x}) \log\big(\hat{p}_i(c|\mathbf{x})\big), \tag{3.1}$$

where $\hat{\mathbf{p}}_i(\mathbf{x}) = \big(\hat{p}_i(c|\mathbf{x})\big)_{i\in\mathcal{I},c\in\mathcal{C}} \in [0,1]^{|\mathcal{C}|}$ denotes a vector of estimated probabilities by a semantic segmentation network $\mathbf{F}$ such that the $\hat{p}_i(c|\mathbf{x})$ is a probability of a pixel location i of a given image $\mathbf{x}$ being a pixel that belongs to the class $c \in \mathcal{C}$.

As previously mentioned, for a semantic segmentation network in the context of anomaly segmentation, it would be a desirable property if such network could output a high prediction uncertainty for OoD pixels which can in turn be quantified with Eq. (3.1). Note that $E_i\big(\hat{\mathbf{p}}_i(\mathbf{x})\big)$ is maximized by the uniform (non-informative) probability distribution $\hat{\mathbf{p}}_i(\mathbf{x})$ which makes it an intuitive uncertainty measure.

Let $\mathcal{D}_{out}^{train}$ denote a set of *out-of-distribution training samples* defined as

$$\mathcal{D}_{out}^{train} = \big\{(\mathbf{x}_j, \mathbf{m}_j) \mid \mathbf{m}_j = \mathbf{M}(\mathbf{x}_j)\big\}_{j=1}^{N_{out}^{train}}, \tag{3.2}$$

where N_{out}^{train} denotes the total number of such training samples and $\mathbf{M}$ an image labeling process that produced the ground truth segmentation mask $\mathbf{m}_j$ for a given OoD image $\mathbf{x}_j$. Note that $\mathcal{D}_{out}^{train} \subset \mathcal{D}_{out}$, where $\subset$ represents a proper subset.

The set of OoD training samples which is also called *known unknowns* serves as a proxy for OoD images. In practice, $\mathcal{D}_{out}^{train}$ is a general-purpose dataset such as COCO dataset [28] that contains diverse taxonomy exceeding the one found in the chosen domain-specific dataset which represents $\mathcal{D}_{in}^{train}$, e.g., Cityscapes [29], which is a dataset used for the domain of road driving.

It has been shown [12] that one can make the output of a semantic segmentation network $\mathbf{F}$ have a high entropy on OoD pixel locations by modifying a standard closed-set neural network training by employing a multi-criteria training objective defined as

$$\mathcal{L} = (1-\lambda)\mathbb{E}_{(\mathbf{x},\mathbf{m})\in\mathcal{D}_{in}^{train}}\Big[l_{in}\big(\mathbf{F}(\mathbf{x}),\mathbf{m}\big)\Big] + \lambda\mathbb{E}_{(\mathbf{x},\mathbf{m})\in\mathcal{D}_{out}^{train}}\Big[l_{out}\big(\mathbf{F}(\mathbf{x}),\mathbf{m}\big)\Big], \tag{3.3}$$

where $\mathcal{D}_{in}^{train}$ is defined by Eq. (2.1), $\mathcal{D}_{out}^{train}$ is defined by Eq. (3.2) and $\mathbf{F}$ is a semantic segmentation network. In order to control the impact of each part of the overall objective,

a hyperparameter $\lambda \in [0, 1]$ is used for creating a convex combination.

When minimizing the overall objective defined by Eq. (3.3), a commonly used cross-entropy loss is applied for in-distribution training samples defined as

$$l_{in}\big(\mathbf{F}(\mathbf{x}), \mathbf{m}\big) = -\frac{1}{H \cdot W} \sum_{i \in \mathcal{I}} \sum_{c \in \mathcal{C}} \mathbb{1}_{m_i = c} \cdot \log\big(\hat{p}_i(c|\mathbf{x})\big), \ (\mathbf{x}, \mathbf{m}) \in \mathcal{D}_{in}^{train}, \tag{3.4}$$

where $\mathbb{1}_{c=m_i} \in \{0, 1\}$ is the indicator function being equal to one if the class index $c \in \mathcal{C}$ is, for a given pixel location $i \in \mathcal{I}$, equal to the class index m_i defined by the ground truth segmentation mask $\mathbf{m}$ and zero otherwise.

For OoD training samples, a slightly modified cross-entropy loss is applied

$$l_{out}\big(\mathbf{F}(\mathbf{x}), \mathbf{m}\big) = -\frac{1}{H \cdot W} \sum_{i \in \mathcal{I}} \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \mathbb{1}_{m_i = OoD(i)} \cdot \log\big(\hat{p}_i(c|\mathbf{x})\big), \ (\mathbf{x}, \mathbf{m}) \in \mathcal{D}_{out}^{train}, \tag{3.5}$$

where $\mathbb{1}_{m_i=OoD(i)} \in \{0, 1\}$ is the indicator function being equal to one if the pixel location $i \in \mathcal{I}$ is labeled as OoD and zero otherwise.

It can be shown that *minimizing* l_{out} defined by Eq. (3.5) is equivalent to *maximizing* per-pixel prediction entropy $E_i\big(\hat{\mathbf{p}}_i(\mathbf{x})\big)$ defined by Eq. (3.1), hence the name *entropy maximization*. Since $\sum_{c \in \mathcal{C}} \hat{p}_i(c|\mathbf{x}) = 1$, if we apply Jensen's inequality [30] to the inner sum of the Eq. (3.5) and omit the indicator function for clarity, then we obtain

$$-\sum_{c \in \mathcal{C}} \frac{1}{|\mathcal{C}|} \log\big(\hat{p}_i(c|\mathbf{x})\big) \geq -\log\left(\sum_{c \in \mathcal{C}} \frac{1}{|\mathcal{C}|} \hat{p}_i(c|\mathbf{x})\right) \geq -\log\left(\frac{1}{|\mathcal{C}|}\right) = \log\left(|\mathcal{C}|\right) \tag{3.6}$$

Since $E_i\big(\hat{\mathbf{p}}_i(\mathbf{x})\big) \leq \log\left(|\mathcal{C}|\right)$, the equality between Eqs. (3.1) and (3.6) holds when the probability distribution $\hat{\mathbf{p}}_i(\mathbf{x})$ is uniform.

The anomaly segmentation score map $\mathbf{a}$ can then be obtained by normalizing per-pixel prediction entropy, i.e.,

$$\mathbf{a} = (a_i)_{i \in \mathcal{I}} \in [0, 1]^{H \times W}, \ a_i = \frac{E_i\big(\hat{\mathbf{p}}_i(\mathbf{x})\big)}{\log\left(|\mathcal{C}|\right)} \tag{3.7}$$

3.3 Meta Classification

Meta classification is the task of discriminating between a false positive prediction and a true positive prediction. In the context of anomaly segmentation, pixels of OoD objects are considered to be the members of the positive class. Training a network with a modified entropy maximization training objective increases the network's sensitivity towards predicting OoD objects and can result in a substantial number of false positive predictions. Applying meta classification in order to post-process the network's prediction has been shown to significantly improve the network's ability to reliably detect OoD objects. Note that even though we apply meta classification as a post-processing step of the entropy maximization training, the following definition of meta classification could potentially be applied to the other anomaly segmentation methods described in Sec. 3.1.

Let $\mathcal{D}_{out}^{test}$ denote the set of *out-of-distribution test samples* such that $\mathcal{D}_{out}^{train} \cap \mathcal{D}_{out}^{test} = \emptyset$ and $\mathcal{D}_{out}^{test} \subset \mathcal{D}_{out}$, where $\subset$ represents a proper subset. By introducing the notion of $\mathcal{D}_{out}^{test}$ we emphasize the fact that the meta classification is employed after the entropy maximization training for which samples from $\mathcal{D}_{out}^{train}$ were used.

A set of pixel locations being predicted as OoD for a given image $\mathbf{x} \in \mathcal{D}_{out}^{test}$ and an anomaly segmentation score map $\mathbf{a}$ computed with Eq. (3.7) is defined as

$$\hat{\mathcal{I}}_{out}(\mathbf{x}, \mathbf{a}) = \big\{ i \in \mathcal{I} \mid a_i \geq t, t \in [0, 1] \big\}, \tag{3.8}$$

where t represents a fixed threshold.

Based on $\hat{\mathcal{I}}_{out}(\mathbf{x}, \mathbf{a})$, a set of connected components representing OoD object predictions is defined as

$$\hat{\mathcal{K}}(\mathbf{x}, \mathbf{a}) \subseteq \mathcal{P}\big(\hat{\mathcal{I}}_{out}(\mathbf{x}, \mathbf{a})\big), \tag{3.9}$$

where $\mathcal{P}\big(\hat{\mathcal{I}}_{out}(\mathbf{x}, \mathbf{a})\big)$ denotes the power set of $\hat{\mathcal{I}}_{out}(\mathbf{x}, \mathbf{a})$.

The following discussion is based on [5, 12, 25, 26, 27]. Meta classifier is a lightweight model added on top of a semantic segmentation network $\mathbf{F}$. After training $\mathbf{F}$ for entropy maximization on the pixels of OoD objects, a structured dataset of hand-crafted metrics is constructed. For every OoD object prediction $\hat{k} \in \hat{\mathcal{K}}(\mathbf{x}, \mathbf{a})$, different pixel-wise uncertainty measures are derived solely from $\hat{\mathbf{p}}(\mathbf{x})$ such as normalized per-pixel prediction

entropy in Eq. (3.1), MSP, etc. In addition to metrics derived from $\hat{\mathbf{p}}(\mathbf{x})$, metrics based on the OoD object prediction geometry features are also included such as the number of pixels contained in $\hat{k}$, various ratios regarding interior and boundary pixels, geometric center, geometric features regarding the neighborhood of $\hat{k}$, etc. [12, 25]

After a dataset with the hand-crafted metrics is constructed, a meta classifier is trained to classify OoD object predictions in one of the following two sets,

$$C_{TP}(\mathbf{x}, \mathbf{a}) = \left\{\hat{k} \in \hat{\mathcal{K}}(\mathbf{x}, \mathbf{a}) \mid IoU(\hat{k}, \mathbf{m}) > 0, (\mathbf{x}, \mathbf{m}) \in \mathcal{D}_{out}^{test}\right\}, \text{ and} \tag{3.10}$$

$$C_{FP}(\mathbf{x}, \mathbf{a}) = \left\{\hat{k} \in \hat{\mathcal{K}}(\mathbf{x}, \mathbf{a}) \mid IoU(\hat{k}, \mathbf{m}) = 0, (\mathbf{x}, \mathbf{m}) \in \mathcal{D}_{out}^{test}\right\}, \tag{3.11}$$

where C_{TP} represents a set of true positive OoD object predictions and C_{FP} a set of false positive OoD object predictions [12]. In the Eqs. (3.10) and (3.11), IoU represents the intersection over union [31] of a OoD object prediction $\hat{k}$ with the corresponding ground truth segmentation mask $\mathbf{m}$. Clearly, OoD object prediction is considered a true positive if IoU is greater than zero, which can equivalently be stated that the OoD object prediction is a true positive if at least one pixel location is correctly classified as OoD. This may seem as too rigorous for some applications, however, in the safety-critical scenarios such as autonomous driving, this condition can be considered reasonable.

During inference, meta classifier predicts whether an OoD object predictions obtained from $\mathbf{F}$ are false positive, i.e., have IoU equal to zero. Certainly, the prediction is done without the access to $\mathbf{m}$ and is based on learned statistical and geometrical properties of the OoD object predictions obtained from the known unknowns. OoD object predictions classified as false positive are then removed and the final prediction is obtained.

3.4 Neural Network Meta Classifier

In [12], authors use logistic regression for the purpose of meta classifying predicted OoD objects. Their main argument for the use of logistic regression is that since it is a linear model, it is possible to analyze the impact of each hand-crafted metric used as an input to the model with an algorithm such as Least Angle Regression (LARS) [32]. However, we argue that even though it is desirable to have an interpretable model in order to analyze

the relevance and the impact of its input, it is possible to achieve a significantly better performance by employing a more expressive type of model such as neural network.

Let $\hat{K}$ be a set containing OoD object predictions for every $\mathbf{x} \in \mathcal{D}_{out}^{test}$ defined as

$$\hat{K} = \bigcup_{(\mathbf{x},\mathbf{m}) \in \mathcal{D}_{out}^{test}} \hat{\mathcal{K}}(\mathbf{x}, \mathbf{a}), \tag{3.12}$$

where $\hat{\mathcal{K}}(\mathbf{x}, \mathbf{a})$ represents OoD object predictions defined by Eq. (3.9) and $\mathbf{a}$ is an anomaly segmentation score map defined by Eq. (3.7).

We formally define the aforementioned hand-crafted metrics dataset as $\mu \subset \mathbb{R}^{|\hat{K}| \times N_m}$, where $|\hat{K}|$ represents the total number of OoD object predictions obtained from the semantic segmentation network $\mathbf{F}$ across all OoD test samples in $\mathcal{D}_{out}^{test}$ and where N_m is the total number of hand-crafted metrics derived from each OoD object prediction. It should be clear from the definition that μ is a matrix in which each row corresponds to OoD object prediction and contains hand-crafted metrics derived from the respective OoD object prediction.

We propose that instead of logistic regression as a meta classifier, a lightweight fully-connected neural network is employed. Let $\mathbf{F}^{meta} : \mu \rightarrow [0, 1]$ denote such neural network. We can interpret that $\mathbf{F}^{meta}$ outputs the probability of a given OoD object prediction being false positive based on the corresponding derived hand-crafted metrics according to Eqs. (3.10) and (3.11). Let p^{FP} denote such probability. Since $\mathbf{F}^{meta}$ is essentially a binary classifier, we can train it using a special case of the cross-entropy loss, the binary cross-entropy loss defined as

$$\mathcal{L}^{meta} = -\frac{1}{N} \sum_{i=1}^{N} y_i \log\left(p_i^{FP}\right) + (1 - y_i) \log\left(1 - p_i^{FP}\right), \tag{3.13}$$

where N represents the number of OoD object predictions included in a mini-batch and y_i represents the ground truth label of a given OoD object prediction and is equal to one if given OoD object prediction is a false positive according to Eq. (3.11) and zero if it is a true positive according to Eq. (3.10).

4 Experiments

In this chapter, as a baseline, we use the method [12] based on entropy maximization training supplemented with a meta classifier. First, we describe the experimental setup. Next, we provide further evaluation of the entropy maximization training. Finally, we describe a modification of the baseline and show that the proposed modification is an improvement in comparison to the baseline. The code is publicly available at `https://github.com/JuricaRuntas/meta-ood`.

4.1 Experimental Setup

We use DeepLabv3+ semantic segmentation model [33] with a WideResNet38 backbone [34] trained by Nvidia [35]. The model has 137,103,936 parameters and is pretrained on Cityscapes dataset [29] containing images of urban street scenes.

The pretrained model is fine-tuned according to the Eq. (3.3). We use Cityscapes dataset [29] as $\mathcal{D}_{in}^{train}$ containing 2,975 images while for $\mathcal{D}_{out}^{train}$ we use COCO dataset [28], a general-purpose dataset that contains images of everyday scenes. For the purpose of $\mathcal{D}_{out}^{train}$, we exclude images containing class instances ("person", "bicycle", "car", "motorcycle", "bus", "truck", "traffic light" and "stop sign") that are also found in Cityscapes dataset. After filtering, 46,751 images remain.

The model is trained for 4 epochs on random square crops of height and width of 480 pixels. Images that have height or width smaller than 480 pixels are resized. Before each epoch, we randomly shuffle 2,975 images from Cityscapes dataset with 297 images randomly sampled from the remaining 46,751 COCO images. Hyperparamters are set according to the baseline [12]: loss weight $\lambda = 0.9$, entropy threshold $t = 0.7$. Adam optimizer [36] is used with the learning rate $\eta = 1 \times 10^{-5}$.

In order to evaluate the anomaly segmentation performance, we utilize two datasets used for the purpose of benchmarking anomaly segmentation performance. Both datasets share the same setup as the Cityscapes dataset [29], i.e., they contain images of road-driving and urban street scenes. The first one is the LostAndFound dataset [37] and its test split named LostAndFound Test consisting of 1,203 images and their corresponding pixel level annotations of small obstacles placed in front of the ego vehicle such as cones, various plastic boxes and car parts. The second one is a part of the Fishyscapes dataset [38] named Fishyscapes Static consisting of 30 images and their corresponding pixel level annotations of anomalous objects such as various animals extracted from the Pascal VOC dataset [31] and blended in the images from the Cityscapes validation split.

4.2 Metrics

As already mentioned, in the context of anomaly segmentation, OoD pixels are considered to be the members of the positive class and in-distribution pixels are considered to be the members of the negative class, i.e., anomaly segmentation is in fact a binary classification task since the network has to distinguish between in-distribution and OoD pixels in a given image. Consequently, we can use the usual metrics for evaluation of the binary classification performance.

We are mostly interested in the receiver operating characteristic (ROC) and precision recall (PR) curves [39, 40]. The area under ROC (AUROC) and the area under PR (AUPRC) summarize the respective curves into a single number whose values are in the range between zero and one. Area under the curve (AUC) measures the degree of separability between distributions of a positive and a negative class achieved by a given binary classifier. ROC curve does not place more emphasis on one class over the other in the case of class imbalance. In our case, the class imbalance is indeed present since the number of OoD pixels in the images is clearly much smaller than the number of in-distribution pixels. On the other hand, PR curve places more emphasis on the members of the positive class, i.e., OoD pixels [39]. Nevertheless, we use both metrics in order to evaluate the binary classification performance. We additionally evaluate the false positive rate at 95% true positive rate (FPR_{95}) [41].

In order to evaluate the impact of the entropy maximization training on the origi-

nal semantic segmentation task of the in-distribution images, we compute intersection-over-union (IoU) [31] of the model predictions on the Cityscapes validation set. More precisely, IoU is computed for the 19 evaluation classes in the Cityscapes dataset and averaged to produce the mean IoU (mIoU) score.

4.3 Evaluation of the Entropy Maximization Training

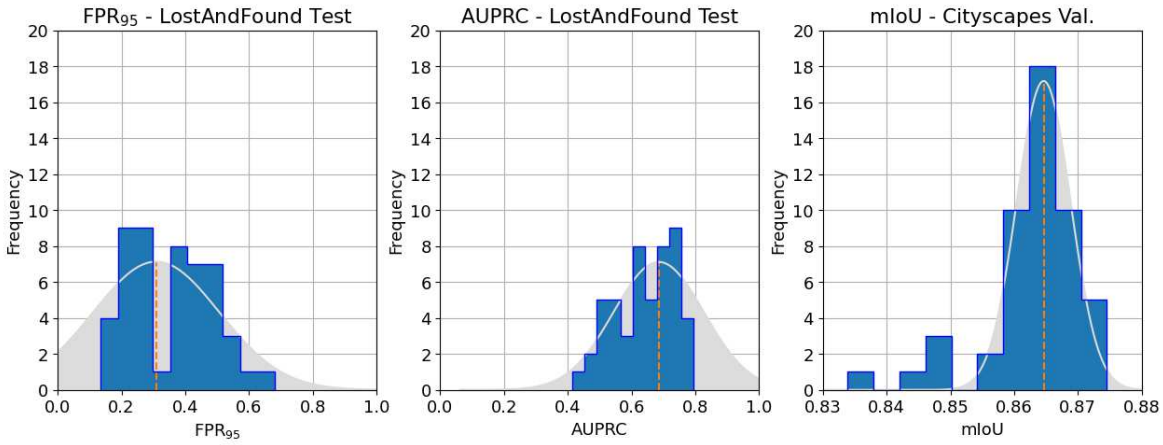


Figure 4.1: Histograms of the obtained metric scores after repeating the entropy maximization training 50 times. The left and the middle histogram show FPR_{95} and AUPRC scores for the anomaly segmentation of the LostAndFound Test. The right histogram shows mIoU score for the semantic segmentation of the Cityscapes validation set obtained after each run. Gray areas represent fitted Gaussian distributions, dashed orange lines represent estimated means and the histograms are represented with a blue color.

In addition to the results and analysis reported in [12], we perform further evaluation of the entropy maximization training. This is motivated by the fact that when performing entropy maximization training, a certain amount of proxy OoD images are randomly sampled from $\mathcal{D}_{out}^{train}$. It is reasonable to assume that such training procedure could be positively or negatively impacted depending on which exact proxy OoD images are included in the random sample. We are interested in the average case when the entropy maximization training is repeated numerous times. The results are illustrated with Fig. 4.1 while Table 4.1 summarizes and compares them to the results reported in [12].

4.4 Evaluation of Neural Network Meta Classifier

As proposed in Sec. 3.4, we substitute the logistic regression with a lightweight fully-connected neural network whose architecture is shown in Table 4.2. The network is

Metric	FPR_{95}		AUPRC		mIoU		$mFPR_{95}$	mAUPRC	mmIoU
Source	Baseline	Reproduced	Baseline	Reproduced	Baseline	Reproduced	Reproduced	Reproduced	Reproduced
LostAndFound Test	0.09	0.15	0.76	0.75	-	-	0.35	0.64	-
Fishyscapes Static	0.09	0.17	0.87	0.64	-	-	0.36	0.42	-
Cityscapes Val.	-	-	-	-	0.89	0.87	-	-	0.86

Table 4.1: Evaluation results of the entropy maximization training. For the columns FPR_{95}, AUPRC and mIoU, Baseline contains scores reported in [12] while Reproduced contains our (best) results obtained while performing 50 runs of the entropy maximization training. While the columns FPR_{95}, AUPRC and mIoU contain results obtained from a single best model, the columns $mFPR_{95}$, mAUPRC, mmIoU contain metrics averaged over 50 obtained models.

trained on the hand-crafted metrics derived from OoD object predictions of the images in LostAndFound Test. The OoD object predictions are obtained from the semantic segmentation model that was beforehand fine-tuned for anomaly segmentation using the entropy maximization training. Derived hand-crafted metrics, i.e., corresponding OoD object predictions are leave-one-out cross validated according to the Eqs. (3.10) and (3.11). The network is trained using Adam optimizer with learning rate $\eta = 1 \times 10^{-3}$ and weight decay $\gamma = 5 \times 10^{-3}$ for 50 epochs with a mini-batch size $N = 128$. Note that in our case, the total number of hand-crafted metrics is $N_m = 75$.

Table 4.2 shows the architecture of the proposed neural network meta classifier. For the sake of comparison, logistic regression meta classifier has 76 parameters (75 parameters for the 75 used hand-crafted metrics and one for the intercept).

#	Layer type	# of input features	# of output features	# of parameters
1.	FC	75	75	5,700
	ReLU	75	75	0
2.	FC	75	75	5,700
	ReLU	75	75	0
3.	FC	75	75	5,700
	ReLU	75	75	0
4.	FC	75	1	76
	Sigmoid	1	1	0
			Total	17,176

Table 4.2: Architecture of the neural network meta classifier. FC stands for fully-connected layer. For the sake of completeness, we also include the used activation function (ReLU) and a sigmoid function (Sigmoid) used for mapping the output on $[0, 1]$ interval.

Performance comparison of meta classifiers is given in Table 4.3 which shows that the proposed neural network meta classifier outperforms logistic regression. For the sake of

Model type	Logistic Regression		Neural Network
Source	Baseline	Reproduced	Ours
AUROC	0.9444	0.9342	**0.9680**
AUPRC	0.7185	0.6819	**0.8418**

Table 4.3: Performance comparison of meta classifiers. Note that the given results are based on the meta classifiers trained on OoD object predictions obtained with entropy threshold $t = 0.7$. Performance of the logistic regression (Reproduced) and the neural network meta classifier is further examined in Fig. 4.2.

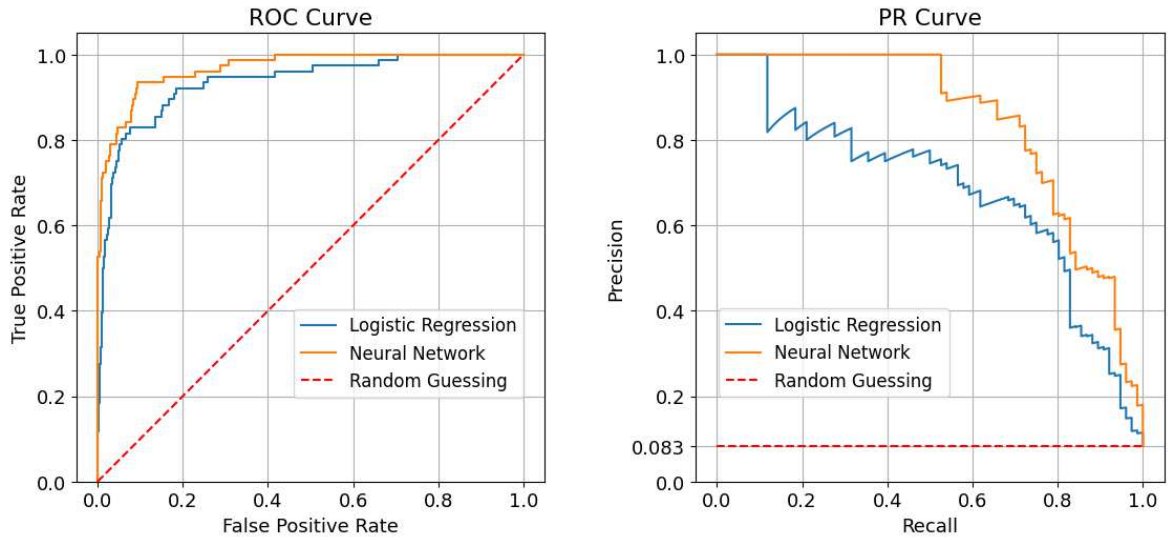


Figure 4.2: ROC and PR meta classifier curves for OoD object predictions of LostAndFound Test images. Note that the AUROC can be seen as the probability that the meta classifier will allocate a higher score to a randomly chosen OoD object than it will to a randomly and independently chosen in-distribution object [40], hence random guessing is represented with a dashed red line whose AUROC is 0.5. On the PR curve, random guessing is represented as a constant dashed red line whose value is equal to the ratio of the number of OoD objects and the total number of predicted OoD objects.

comparison, we provide the results reported in [12]. ROC and PR curves are displayed in Fig. 4.2 where we plot the performance of our neural network meta classifier and of the reproduced logistic regression meta classifier.

4.5 Neural Network Meta Classifier Architectures

In this section, we further examine potential neural network meta classifier architectures. In our attempt to stay unbiased, another run of the entropy maximization training is performed that yields a different semantic segmentation model than the one used in the previous section. Clearly, different model generates different OoD object predictions

and their respective hand-crafted metrics than the ones used to train meta classifiers whose performance is reported in Table 4.3 and Fig. 4.2. Note that for the rest of this section, we fix hyperparameters regarding neural network meta classifier to the values used in the previous section. Also note that in all the following experiments, we are trying to keep neural network meta classifier as lightweight as possible. We arbitrarily set an upper limit on the number of learnable parameters to be around 30,000, although one could try to go even higher.

First, we experiment with the usage of different number of fully-connected layers, i.e., we explore how depth affects the performance of a neural network meta classifier. We keep the total number of neurons per layer fixed and equal to 75. The results are shown in Table 4.4.

Architecture	AUROC	AUPRC	# of parameters
Logistic Regression	0.9444	0.7176	76
NN-L-0	0.9276	0.5820	76
NN-L-1	0.9600	0.8098	5,776
NN-L-2	0.9673	0.8560	11,476
NN-L-3	**0.9708**	**0.8639**	**17,176**
NN-L-4	0.9662	0.8383	22,876
NN-L-5	0.9667	0.8374	28,576
NN-L-6	0.9700	0.8378	34,276

Table 4.4: Performance comparison of neural network meta classifier architectures differing in the number of fully-connected layers while the total number of neurons per layer is fixed and equal to 75. Note that the NN-L-0 is a neural network with a single neuron that has the same number of parameters as logistic regression. It is outperformed by logistic regression most likely because of the set training hyperparameters. NN-L-3 is the architecture shown in Table 4.2. If the goal is to keep neural network meta classifier as lightweight as possible, one could also argue that NN-L-2 seems reasonable since its AUROC differs less than 1% (0.35%) in comparison to AUROC achieved by NN-L-3 while their respective AUPRC scores also differ less than 1% (0.8%).

We continue our experiments with analysis of the usage of different number of neurons per fully-connected, i.e., we explore how width affects the performance of a neural network meta classifier. From the results in Table 4.4, we conclude that NN-L-3 offers significantly better performance in comparison to logistic regression meta classifier and choose to fix the number of layers to be equal to three. Table 4.5 shows the performance of a neural network meta classifier consisting of three fully-connected layers with a different number of neurons per layer. Note that a given number of neurons is the same in every layer.

Architecture	AUROC	AUPRC	# of parameters
Logistic Regression	0.9444	0.7176	76
NN-L-3-N-15	0.9604	0.7829	1,636
NN-L-3-N-25	0.9666	0.8206	3,226
NN-L-3-N-35	0.9639	0.8349	5,216
NN-L-3-N-45	0.9659	0.8412	7,606
NN-L-3-N-55	0.9661	0.8435	10,396
NN-L-3-N-65	0.9671	0.8297	13,586
NN-L-3-N-75	**0.9708**	**0.8639**	**17,176**
NN-L-3-N-85	0.9650	0.8415	21,166
NN-L-3-N-95	0.9705	0.8645	25,556
NN-L-3-N-110	0.9693	0.8605	32,891

Table 4.5: Performance comparison of neural network meta classifier architectures differing in the number of neurons per layer. Note that a given number of neurons is the same in all three layers. The results suggest that increasing layer width can improve the performance. However, we again choose NN-L-3-N-75 as the best option due to the smaller number of parameters in comparison to NN-L-3-N-95 and NN-L-3-N-110. One could also argue that choosing even smaller models such as NN-L-3-N-55 or NN-L-3-N-45 is reasonable due to the significant reduction of the number of parameters. Note that NN-L-3-N-75 is the same architecture as NN-L-3 in Table 4.4.

Inspired by the success of NN-L-3 in Table 4.4 and NN-L-3-N-75 in Table 4.5, we further investigate how asymmetrical architecture affects the performance of neural network meta classifier. For this purpose, we fix the number of layers to be equal to three and the total number of neurons to be equal to 225 (as in NN-L-3) while also keeping the total number of parameters to be around 17,176. Results are shown in Table 4.6.

Architecture	AUROC	AUPRC	# of parameters
Logistic Regression	0.9444	0.7176	76
NN-L1-130-L2-45-L3-30	0.9674	0.8596	17,186
NN-L1-60-L2-140-L3-30	0.9722	0.8552	17,361
NN-L1-40-L2-80-L3-135	0.9670	0.8516	17,391

Table 4.6: Performance comparison of asymmetrical neural network meta classifier architectures consisting of three layers and having the total number of neurons equal to 225.

We also tried to substitute ReLU activation function with a more traditional activation functions such as sigmoid and tanh which resulted in a worse performance than the one reported in Tables 4.4, 4.5, 4.6 and for that reason, we omit those results.

5 Discussion

In this chapter, first we discuss the noticed variability of the entropy maximization training and attempt to give an explanation of its cause. Then, we provide further discussion of our proposed neural network meta classifier and show its advantages and drawbacks when using it instead of logistic regression meta classifier.

5.1 Variability of the Entropy Maximization Training

For the rest of this section, we assume that $\mathcal{D}_{in}^{train}$ defined by Eq. (2.1) is a dataset used for the domain of road driving such as the Cityscapes dataset that we used in our experiments. This implies that the in-distribution images contained in $\mathcal{D}_{in}$ are in fact the images of road driving scenes. While we focus on a road driving, we believe that the following discussion can be easily generalized and applied to the other domains.

In Fig. 4.1, the presence of a significant variance is clearly visible for FPR_{95} and AUPRC metrics. It should be obvious that the entropy maximization training formally described with Eq. (3.3) and more practically described in Sec. 4.1 contains a certain degree of randomness due to the utilization of random samples of proxy OoD images from $\mathcal{D}_{out}^{train}$ defined by Eq. (3.2). In our experiments, we used a subset of COCO dataset for the purpose of $\mathcal{D}_{out}^{train}$ which contains OoD images containing objects that are not instances of any class found in the Cityscapes dataset.

The exact proxy OoD images included in a random sample of $\mathcal{D}_{out}^{train}$ during entropy maximization training are unknown and not further examined. However, in our attempt to give an explanation for the cause of variability of the entropy maximization training, we argue that such blind sampling procedure is in fact the cause of variability.

We introduce the notion of *high informative* and *low informative* proxy OoD images.

What we mean by high and low informative is illustrated with Fig. 5.1. We have noticed empirically that the high informative proxy OoD images have two important characteristics that differentiate them from the low informative proxy OoD images: *spatially clear separation between objects* and *clear object boundaries*.

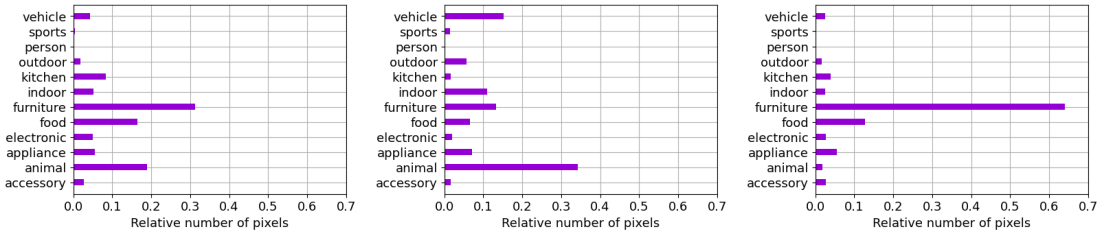
(a) Examples of *high informative* proxy OoD images. (b) Examples of *low informative* proxy OoD images.

Figure 5.1: Examples of high and low informative proxy OoD images. The first row contains the proxy OoD images while the second row contains ground truth segmentation masks such that the white regions represent pixels labeled as OoD according to the indicator function in Eq. (3.5). The black regions are considered as background and are ignored during training.

The hypothesis is that the low informative proxy OoD images have little to no impact on the entropy maximization training or can even negatively impact the training procedure. On the other hand, high informative proxy OoD images are the ones from which the semantic segmentation network can learn to reliably output a high entropy score on OoD pixels of images seen during inference, denoted by $\mathcal{D}_{out} \setminus \mathcal{D}_{out}^{train}$, where $\setminus$ represents a set difference.

In attempt to prove our hypothesis, we perform the entropy maximization training on the subsets of the generated COCO OoD proxy. We consider it difficult to universally quantify the mentioned characteristics of the high informative proxy OoD images, however, we notice a significant correlation between the percentage of the labeled OoD pixels and the desirable properties found in the high informative OoD proxy images. We use the generated COCO OoD proxy which we denote as $\mathcal{D}_{out}^{train}$ for the creation of the two disjoint sets such that the first contains images from $\mathcal{D}_{out}^{train}$ that have at most 20% of pixels labeled as OoD (denoted as L-20%-OoD) and the second that contains images from $\mathcal{D}_{out}^{train}$ that have at least 80% of pixels labeled as OoD (denoted as M-80%-OoD). The results are displayed in Tables 5.1 and 5.2. Table 5.1 shows that performing the entropy maximiza-

tion training using M-80%-OoD results in a little to no improvement in comparison to the model trained exclusively on the in-distribution images. On the other hand, using L-20%-OoD produced even better results than the ones obtained with the usage of COCO OoD proxy.



(a) COCO OoD proxy (b) L-20%-OoD (c) M-80%-OoD

Figure 5.2: Relative number of pixels per supercategory for used OoD proxies.

Metric	FPR_{95}				AUPRC			
Source	DLV3+W38	Reproduced	L-20%-OoD	M-80%-OoD	DLV3+W38	Reproduced	L-20%-OoD	M-80%-OoD
LostAndFound Test	0.35	0.15	**0.09**	0.13	0.46	0.75	**0.78**	0.48
Fishyscapes Static	0.19	0.17	**0.12**	0.31	0.25	0.64	**0.73**	0.25

Table 5.1: Results for the entropy maximization training using COCO OoD proxy subsets. The displayed results are generated from the best model obtained while performing 10 runs of the entropy maximization training (columns L-20%-OoD and M-80%-OoD). For the sake of comparison, we also include the best results from Table 4.1 in column Reproduced and the results from the model used for fine-tuning [35] in column DLV3+W38. mIoU for Cityscapes validation set is equal to 0.87 regardless of which OoD proxy was used.

Metric	$mFPR_{95}$		mAUPRC	
Source	L-20%-OoD	M-80%-OoD	L-20%-OoD	M-80%-OoD
LostAndFound Test	**0.25**	0.44	**0.74**	0.44
Fishyscapes Static	**0.23**	0.65	**0.68**	0.20

Table 5.2: Averaged results for the entropy maximization training using COCO OoD proxy subsets, Fig. 5.2(b) and Fig. 5.2(c). The results are averaged over 10 runs. Using L-20%OoD as a OoD proxy clearly results in better anomaly segmentation performance. Note that the numbers in this table cannot be compared to the ones in Table 4.1 due to the different number of runs. However, it is worth mentioning that we had not been able to achieve 10 consecutive runs that would match the numbers reported in this table when performing 50 runs of the entropy maximization training using COCO OoD proxy in Fig. 5.2(a). mIoU for Cityscapes validation set is equal to 0.87 regardless of which OoD proxy was used.

Fig. 5.2 shows the relative number of pixels per supercategory in each of the three used OoD proxies. One could question whether our additional splitting of the COCO OoD proxy resulted in the loss of the class diversity and consequently resulted in better performing models due to the specialization in a supercategory such as animals in

Fig. 5.2(b). This may be the case for Fisyscapes Static since the majority of the images contain extracted animals from the Pascal VOC dataset [31], although they are limited in the visual diversity, e.g., the same dog is blended in the multiple images, however, it is unlikely that is the case for the LostAndFound Test since only 10% of the images have animals in them and they are exclusively dogs. We also argue that the specialization in the furniture supercategory in Fig. 5.2(c) is irrelevant since the images in M-80%-OoD containing furniture and objects from the other supercategories have most of the pixels labeled as OoD like the examples in Fig. 5.1(b) or even more. However, if the specialization is indeed a factor that affects the entropy maximization training, then that could potentionally have a far-reaching implications mentioned at the end of this section. Finally, the reason behind splitting COCO OoD proxy in L-20%-OoD and M-80%-OoD is to clearly show that our hypothesis is in fact reasonable. However, further analysis is needed for the images not contained in either of the subsets.

The main drawback of the entropy maximization training in comparison with other anomaly segmentation methods mentioned in Sec. 3.1 is the inevitable impairment of the original task performance, i.e., the semantic segmentation of the in-distribution images. This comes from the fact that the model is being trained or fine-tuned on the mixture of OoD proxy images and in-distribution images as defined by Eq. (3.3). The impairment is illustrated with mIoU histogram of the Cityscapes Validation in Fig. 4.1. For the sake of comparison, the model used for fine-tuning [35] trained exclusively on in-distribution images of Cityscapes dataset has mIoU equal to 0.90. The models obtained while performing our experiments have mIoU 0.87 regardless of which OoD proxy was used.

To summarize this discussion, we showed that the variability of the entropy maximization training is related to which exact OoD images are randomly sampled from $\mathcal{D}_{out}^{train}$. In order to decrease the variability, a naive approach would be to increase the number of OoD images that are randomly sampled before each epoch of the training or even increase the number of training epochs. However, one must take caution with such approaches due to the catastrophic forgetting [42]. We argue that the more careful selection of the images included in $\mathcal{D}_{out}^{train}$ could be beneficial to the entropy maximization training as shown in this section. One could also go even further and question which classes of the images should be included in $\mathcal{D}_{out}^{train}$ if we consider $\mathcal{D}_{in}$ fixed. This approach

would be very similar to the usual fine-tuning procedure where we deliberately choose the training examples on which we want our starting model to be additionally trained. For example, are the images of vehicles such as airplanes or various dishes, e.g, the first column in Fig. 5.1(b), relevant OoD images in the context of road driving. This kind of reasoning entails the question of what is the semantic segmentation network actually learning during the entropy maximization training. Further analysis that goes beyond the scope of this thesis is needed in order to gain a better insight and is left for the potential future work.

5.2 Neural Network vs. Logistic Regression

In this section, we further examine and analyze the proposed neural network meta classifier in comparison to the logistic regression used in [12].

We start our analysis with a concept that, in this context, we call *meta overfitting*. We consider this as the main drawback of applying meta classification in order to remove false positive predictions regardless of the chosen meta classifier. What we mean by meta overfitting is illustrated with Fig. 5.3. Meta classifier, regardless of the type (logistic regression, neural network or something else), is a model that has to learn to remove false positive OoD object predictions based on the various statistical and geometrical properties of the known OoD objects as described in Sec. 3.3. Clearly, one can immediately notice that this entails a similar problem as the one encountered while performing entropy maximization training where the generalization ability of the model depends on the used OoD proxy as discussed in the previous section. Fig. 5.3 shows that the semantic segmentation network after the entropy maximization training successfully segments OoD objects, however, neural network and logistic regression meta classifiers attempt to remove them. Formally, from the perspective of Eqs. (3.10) and (3.11), meta classifiers correctly mark and remove them because those OoD object predictions are not marked as OoD in the respective ground truth segmentation masks. However, we argue that such overfitting behavior is not desirable. Note that the neural network meta classifier, however, shows a greater generalization ability in the examples in Figs. 5.3 and 5.5.

Fig. 5.4 contains examples of neural network meta classifier outperforming logistic regression. Note that these examples are meant to illustrate how neural network meta

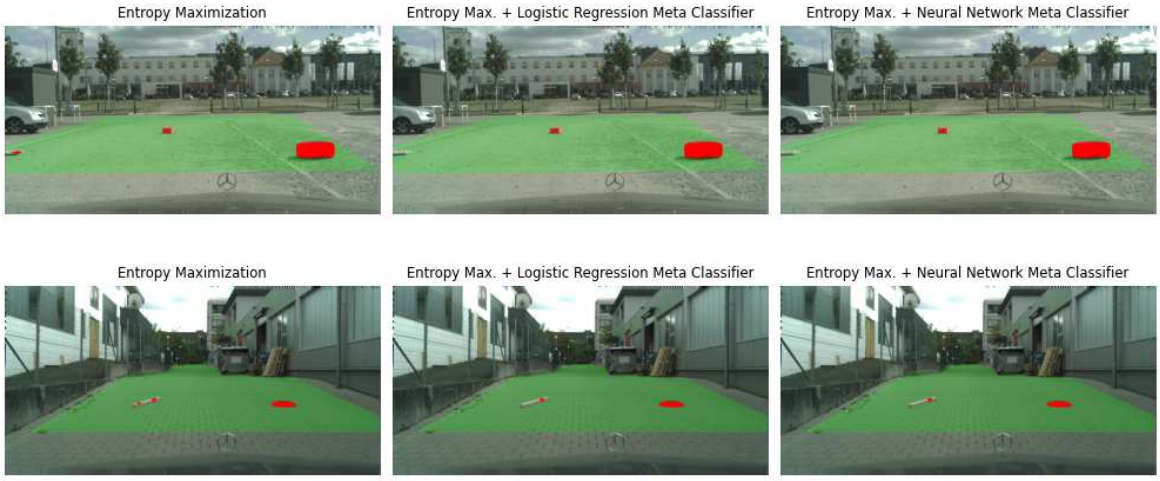


Figure 5.3: Examples of meta overfitting. In the first row, after the entropy maximization training, a semantic segmentation network predicted that the white traffic sign located at the leftmost part of the region of interest (marked with green color) is OoD object. Both neural network and logistic regression meta classifiers marked that OoD object prediction as false positive and removed it, although, neural network meta classifier kept a few barely visible pixels marked as OoD (and hence kept the OoD object prediction). In the second row, semantic segmentation network predicted that the wooden beam is OoD object. Both classifiers marked a part of the OoD prediction as false positive and removed it. Note that neither the white traffic sign nor the wooden beam are marked as OoD in the respective ground truth segmentation masks.

classifier is more successful in removing false positive OoD object predictions. However, we find that the neural network meta classifier especially outperforms logistic regression when OoD object predictions consist of a very small number of pixels which is harder to visually notice. Removal of such small OoD object predictions is the biggest advantage over using logistic regression as a meta classifier. Fig. 5.5 illustrates examples where the logistic regression outperforms neural network meta classifier.

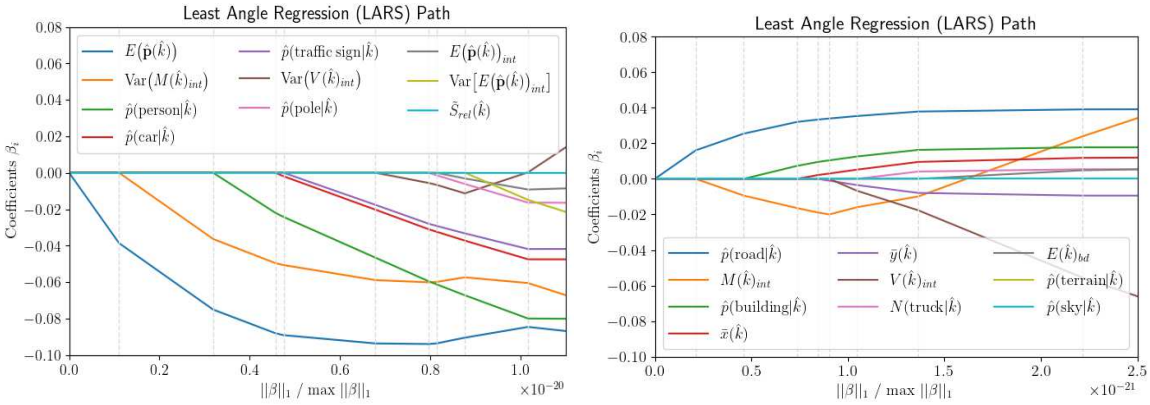
5.3 Interpretability of Neural Network Meta Classifier

In general, we can consider a neural network as a black box for which we have no exact knowledge of its inner workings. On the other hand, linear models such as logistic regression can be analyzed with methods such as Least Angle Regression (LARS) [32]. The main idea is to analyze the impact of each hand-crafted metric on the performance of logistic regression using LARS and evaluate its performance using the hand-crafted metrics most correlated with the response. We can then observe how neural network meta classifier responds to them. Fig. 5.6 shows LARS paths for ten hand-crafted metrics most correlated with the response of logistic regression, i.e., the ones which con-

Figure 5.4: Examples of neural network meta classifier outperforming logistic regression. In the first row, false positive OoD object prediction in the middle of the ROI is removed by neural network meta classifier, however logistic regression meta classifier does not remove it. In the second row, semantic segmentation network predicted OoD object near the blue car, most likely due to the contours drawn by the shadow being mistaken as object boundaries. Neural network meta classifier successfully removes it.

Figure 5.5: Examples of logistic regression outperforming neural network meta classifier. In the first row, logistic regression meta classifier successfully removes OoD object predictions not marked as OoD according to the ground truth segmentation mask (papers on the left and a plastic bag on the right) while neural network meta classifier does not remove the plastic bag. We find this interesting because in this example, neural network meta classifier shows a better generalization ability. In the second row, segmentation network marks part of the road as OoD. It is successfully removed by logistic regression while the neural network meta classifier does not remove it.

tribute the most in classifying OoD object predictions according to Eqs. (3.10) and (3.11). Fig. 5.6(a) shows LARS path for the hand-crafted metrics constructed from the output of semantic segmentation network using the entropy threshold $t = 0.3$ for which authors in [12] showed that it yields the best meta classification performance. However, the en-



(a) LARS path for the entropy threshold $t = 0.3$ (b) LARS path for the entropy threshold $t = 0.7$

Figure 5.6: LARS paths for the hand-crafted metrics constructed from the output of a semantic segmentation network. Note that we apply LARS on the hand-crafted metrics obtained with the entropy threshold $t = 0.3$ and $t = 0.7$ and display the top ten hand-crafted metrics most correlated with the response of logistic regression. A detailed description of the hand-crafted metrics can be found in [12].

tropy threshold $t = 0.7$ results in the best overall anomaly segmentation performance and Fig 5.6(b) shows LARS path for the hand-crafted metrics obtained while using such threshold.

One can certainly interpret LARS as a way of sorting the hand-crafted metrics based on the impact on the response of logistic regression. We are interested in the values of AUROC and AUPRC metrics when we train meta classifiers on the subsets of the hand-crafted metrics dataset μ. First, we take a subset of μ that contains only values of the most correlated hand-crafted metric according to LARS, i.e., $\hat{p}(\mathrm{road}|\hat{k})$ in Fig. 5.6(b). Then, we add the second most correlated hand-crafted metric according to LARS to the previous subset and so on. We accumulate the hand-crafted metrics in order in which they are given by LARS and evaluate AUROC and AUPRC. We perform the same process with neural network meta classifier. Fig. 5.7 shows the resulting graphs regarding AUROC and AUPRC metrics.

One can immediately notice that in Fig. 5.7, the neural network meta classifier behaves in the similar manner as the logistic regression meta classifier. For the logistic regression meta classifier, after we take a subset of μ containing 21 most correlated hand-crafted metrics according to LARS, adding remaining hand-crafted metrics results in little to no improvement in performance. We can see that the neural network meta classifier exhibits a similar behavior, although in more unstable manner. The obvious differ-

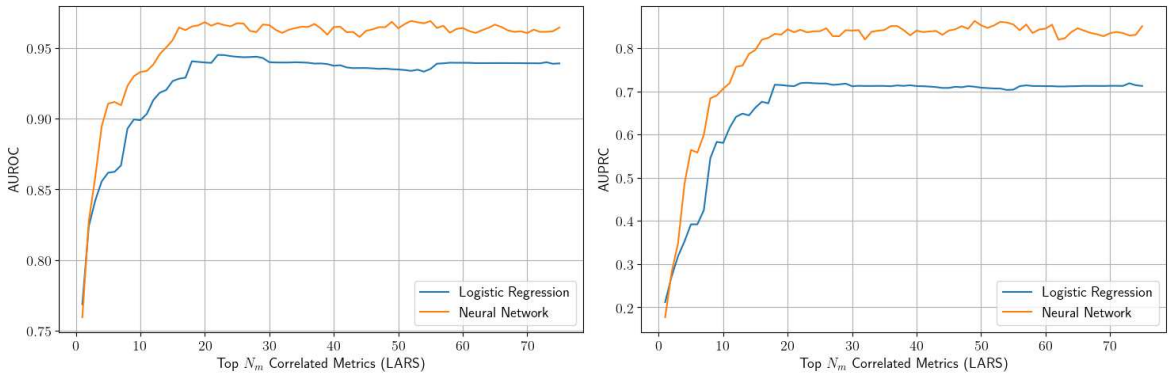


Figure 5.7: Performance comparison of logistic regression meta classifier and neural network meta classifier when trained on subsets of the hand-crafted metrics dataset μ. Note that the total number of hand-crafted metrics is equal to 75. For each value N_m on the x-axis, we train the meta classifiers on the subset of μ such that we take the first N_m metrics showing the most correlation with the response according to LARS.

ence in performance can be most likely attributed to the fact that neural network meta classifier is more expressive and better aggregates the hand-crafted metrics. We argue, at least from what can be seen in Fig. 5.7, that the hand-crafted metrics having the most impact on the performance of logistic regression meta classifier also do so in the case of neural network meta classifier. However, it is questionable if our observation depends on the used hand-crafted metrics dataset μ. Further evaluation is needed in order to show that such claim holds, e.g., performing the entropy maximization training and crafting μ numerous times followed by the evaluation shown in Fig. 5.7.

Finally, the reason why using a neural network instead of logistic regression results in a significantly better performance as shown in Table 4.3 and Fig. 4.2 is based on the fact that neural network is a nonlinear model and it is more expressive due to the increased number of parameters as shown in Table 4.2 which means it can better approximate an arbitrary function. The main drawback is the loss of interpretability as mentioned in Sec. 3.4. It is questionable if the loss of interpretability is in fact a drawback. For example, in the safety critical environments such as road driving where autonomous driving systems are employed, one could argue that it is more important that the meta classifier performs as accurately as possible than to be as interpretable as possible. This kind of opinion is clearly present in the current state-of-the-art solutions for many other tasks where neural networks are employed. The only difference in our case is that this is a rare occasion were we have ability to choose between a model that is interpretable and has a solid performance and a model that is not interpretable and offers a significantly better

performance than the interpretable one. In complex tasks such as semantic segmentation, we do not have that kind of a choice due to the fact that traditional interpretable methods are not of a great use in the complex scenarios such as road driving. Nevertheless, it is up to the system designer to decide what option is more favorable in a given scenario.

6 Conclusion

In this thesis, we explored the anomaly segmentation method called *entropy maximization*. Using such method can increase the network's sensitivity towards predicting OoD objects and can result in a substantial number of false positive predictions [12], hence the *meta classification* post-processing step is applied in order to improve network's ability to reliably detect OoD objects. We proposed a modification of the original meta classification procedure described in [12] where we substitute a logistic regression meta classifier with a neural network meta classifier. Our experimental results show that employing a neural network meta classifier results in a significantly better performance in comparison to logistic regression meta classifier. We presented additional analysis of our proposed modification which examined the potential model architectures and compared it to the logistic regression.

The main advantages of using a neural network meta classifier is significantly improved ability of detecting false positive OoD object predictions, especially the ones consisting of a very small number of pixels. Also, a neural network meta classifier exhibits a greater generalization ability in comparison to the logistic regression meta classifier. A potential drawback of using a neural network as a meta classifier is the loss of interpretability. Whether interpretability or performance is more favorable in a given scenario is up to the system designer to decide.

Furthermore, we provided additional analysis of the entropy maximization training which showed that such method heavily relies on the used OoD proxy containing OoD images and in order to ensure its effectiveness, caution must be taken when choosing images that are going to be included in such proxy. Our experimental results demonstrated that certain OoD proxy images can be more or less beneficial to the entropy maximization training in terms of how well can a semantic segmentation neural network detect

OoD objects. Even though we provided additional analysis of the entropy maximization training, a more comprehensive analysis is needed in order to gain a better insight in how OoD proxy should be constructed in order to ensure a reasonable OoD detection performance. Also, it would be interesting to see how meta classification can be applied as a post-processing step of the other anomaly segmentation methods. We leave this ideas for the potential future work.

References

[1] J. Janai, F. Güney, A. Behl, and A. Geiger, “Computer vision for autonomous vehicles: Problems, datasets and state-of-the-art,” *CoRR*, vol. abs/1704.05519, 2017. [Online]. Available: http://arxiv.org/abs/1704.05519

[2] G. D. Biase, H. Blum, R. Siegwart, and C. Cadena, “Pixel-wise anomaly detection in complex driving scenes,” *CoRR*, vol. abs/2103.05445, 2021. [Online]. Available: https://arxiv.org/abs/2103.05445

[3] S. Cakir, M. Gauß, K. Häppeler, Y. Ounajjar, F. Heinle, and R. Marchthaler, “Semantic segmentation for autonomous driving: Model evaluation, dataset generation, perspective comparison, and real-time capability,” 2022. [Online]. Available: https://arxiv.org/abs/2207.12939

[4] K. Wong, S. Wang, M. Ren, M. Liang, and R. Urtasun, “Identifying unknown instances for autonomous driving,” *CoRR*, vol. abs/1910.11296, 2019. [Online]. Available: http://arxiv.org/abs/1910.11296

[5] R. Chan, S. Uhlemeyer, M. Rottmann, and H. Gottschalk, “Detecting and learning the unknown in semantic segmentation,” 2022. [Online]. Available: https://arxiv.org/abs/2202.08700

[6] S. S. Haykin, *Neural networks and learning machines*. Upper Saddle River, NJ: Pearson Education, 2009.

[7] H. Song, M. Kim, D. Park, and J. Lee, “Learning from noisy labels with deep neural networks: A survey,” *CoRR*, vol. abs/2007.08199, 2020. [Online]. Available: https://arxiv.org/abs/2007.08199

[8] R. Szeliski, *Computer Vision Algorithms and Applications*, 2nd ed., ser. Texts in Computer Science. Cham: Springer International Publishing, 2022.

[9] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. [Online]. Available: https://openreview.net/forum?id=Hkg4TI9xl

[10] B. Gao and L. Pavel, "On the properties of the softmax function with application in game theory and reinforcement learning," 2018. [Online]. Available: https://arxiv.org/abs/1704.00805

[11] S. Liang, Y. Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. [Online]. Available: https://openreview.net/forum?id=H1VGkIxRZ

[12] R. Chan, M. Rottmann, and H. Gottschalk, "Entropy maximization and meta classification for out-of-distribution detection in semantic segmentation," *CoRR*, vol. abs/2012.06575, 2020. [Online]. Available: https://arxiv.org/abs/2012.06575

[13] P. Bevandic, I. Kreso, M. Orsic, and S. Segvic, "Simultaneous semantic segmentation and outlier detection in presence of domain shift," *CoRR*, vol. abs/1908.01098, 2019. [Online]. Available: http://arxiv.org/abs/1908.01098

[14] ——, "Dense outlier detection and open-set recognition based on training with noisy negative images," *CoRR*, vol. abs/2101.09193, 2021. [Online]. Available: https://arxiv.org/abs/2101.09193

[15] ——, "Discriminative out-of-distribution detection for semantic segmentation," *CoRR*, vol. abs/1808.07703, 2018. [Online]. Available: http://arxiv.org/abs/1808.07703

[16] Y. Gal and Z. Ghahramani, “Dropout as a bayesian approximation: Representing model uncertainty in deep learning,” 2016.

[17] A. Kendall, V. Badrinarayanan, and R. Cipolla, “Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding,” *CoRR*, vol. abs/1511.02680, 2015. [Online]. Available: http://arxiv.org/abs/1511.02680

[18] B. Lakshminarayanan, A. Pritzel, and C. Blundell, “Simple and scalable predictive uncertainty estimation using deep ensembles,” 2017.

[19] K. Lee, K. Lee, H. Lee, and J. Shin, “A simple unified framework for detecting out-of-distribution samples and adversarial attacks,” 2018.

[20] T. Denouden, R. Salay, K. Czarnecki, V. Abdelzad, B. Phan, and S. Vernekar, “Improving reconstruction autoencoder out-of-distribution detection with mahalanobis distance,” *CoRR*, vol. abs/1812.02765, 2018. [Online]. Available: http://arxiv.org/abs/1812.02765

[21] J. van Amersfoort, L. Smith, Y. W. Teh, and Y. Gal, “Simple and scalable epistemic uncertainty estimation using a single deep deterministic neural network,” *CoRR*, vol. abs/2003.02037, 2020. [Online]. Available: https://arxiv.org/abs/2003.02037

[22] Y. Li and J. Kosecka, “Uncertainty aware proposal segmentation for unknown object detection,” *CoRR*, vol. abs/2111.12866, 2021. [Online]. Available: https://arxiv.org/abs/2111.12866

[23] K. Lis, K. K. Nakka, P. Fua, and M. Salzmann, “Detecting the unexpected via image resynthesis,” *CoRR*, vol. abs/1904.07595, 2019. [Online]. Available: http://arxiv.org/abs/1904.07595

[24] Y. Xia, Y. Zhang, F. Liu, W. Shen, and A. L. Yuille, “Synthesize then compare: Detecting failures and anomalies for semantic segmentation,” *CoRR*, vol. abs/2003.08440, 2020. [Online]. Available: https://arxiv.org/abs/2003.08440

[25] M. Rottmann, P. Colling, T. Hack, F. Hüger, P. Schlicht, and H. Gottschalk, “Prediction error meta classification in semantic segmentation: Detection

via aggregated dispersion measures of softmax probabilities," *CoRR*, vol. abs/1811.00648, 2018. [Online]. Available: http://arxiv.org/abs/1811.00648

[26] M. Rottmann and M. Schubert, "Uncertainty measures and prediction quality rating for the semantic segmentation of nested multi resolution street scene images," *CoRR*, vol. abs/1904.04516, 2019. [Online]. Available: http://arxiv.org/abs/1904.04516

[27] P. Oberdiek, M. Rottmann, and G. A. Fink, "Detection and retrieval of out-of-distribution objects in semantic segmentation," *CoRR*, vol. abs/2005.06831, 2020. [Online]. Available: https://arxiv.org/abs/2005.06831

[28] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," *CoRR*, vol. abs/1405.0312, 2014. [Online]. Available: http://arxiv.org/abs/1405.0312

[29] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," *CoRR*, vol. abs/1604.01685, 2016. [Online]. Available: http://arxiv.org/abs/1604.01685

[30] S. S. Dragomir, J. Pečarić, and L. E. Persson, "Properties of some functionals related to jensen's inequality," *Acta Mathematica Hungarica*, vol. 70, no. 1, pp. 129–143, Mar 1996. https://doi.org/10.1007/BF00113918

[31] M. Everingham, S. M. A. Eslami, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, pp. 98 – 136, 2014. [Online]. Available: https://api.semanticscholar.org/CorpusID:207252270

[32] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *The Annals of Statistics*, vol. 32, no. 2, Apr. 2004. https://doi.org/10.1214/009053604000000067

[33] L. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder

with atrous separable convolution for semantic image segmentation," *CoRR*, vol. abs/1802.02611, 2018. [Online]. Available: http://arxiv.org/abs/1802.02611

[34] Z. Wu, C. Shen, and A. van den Hengel, "Wider or deeper: Revisiting the resnet model for visual recognition," *CoRR*, vol. abs/1611.10080, 2016. [Online]. Available: http://arxiv.org/abs/1611.10080

[35] Y. Zhu, K. Sapra, F. A. Reda, K. J. Shih, S. D. Newsam, A. Tao, and B. Catanzaro, "Improving semantic segmentation via video propagation and label relaxation," *CoRR*, vol. abs/1812.01593, 2018. [Online]. Available: http://arxiv.org/abs/1812.01593

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[37] P. Pinggera, S. Ramos, S. Gehrig, U. Franke, C. Rother, and R. Mester, "Lost and found: Detecting small road hazards for self-driving vehicles," *CoRR*, vol. abs/1609.04653, 2016. [Online]. Available: http://arxiv.org/abs/1609.04653

[38] H. Blum, P. Sarlin, J. I. Nieto, R. Siegwart, and C. Cadena, "The fishyscapes benchmark: Measuring blind spots in semantic segmentation," *CoRR*, vol. abs/1904.03215, 2019. [Online]. Available: http://arxiv.org/abs/1904.03215

[39] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *Proceedings of the 23rd International Conference on Machine Learning*, ser. ICML '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 233–240. https://doi.org/10.1145/1143844.1143874

[40] J. Hogan and N. M. Adams, "On averaging ROC curves," *Transactions on Machine Learning Research*, 2023, survey Certification. [Online]. Available: https://openreview.net/forum?id=FByH3qL87G

[41] J. Henriksson, C. Berger, M. Borg, L. Tornberg, S. R. Sathyamoorthy, and C. Englund, "Performance analysis of out-of-distribution detection on trained neural networks," *Information and Software Technology*, vol. 130, p. 106409, Feb. 2021. https://doi.org/10.1016/j.infsof.2020.106409

[42] M. McCloskey and N. J. Cohen, “Catastrophic interference in connectionist networks: The sequential learning problem,” *Psychology of Learning and Motivation*, vol. 24, pp. 109–165, 1989. [Online]. Available: https://api.semanticscholar.org/CorpusID:61019113

Abstract

Uncertainty of image segmentation regarding out-of-distribution samples

Jurica Runtas

Semantic segmentation is a computer vision task in which each pixel of an image is assigned into one of the predefined classes. Deep neural networks (DNNs) are a contemporary solution for such task and are usually trained to operate on a predefined closed set of classes. In open-set environments, it is possible to encounter anomalies, i.e., semantically unknown objects. Road driving is an example of such environment in which, from a safety standpoint, it is important to ensure that a DNN is able to indicate when it is operating outside of its learned semantic domain. One of the methods used for that purpose is entropy maximization paired with a post-processing step called meta classification which increases the reliability of anomaly detection. We propose a meta classification approach that significantly improves the reliability of anomaly detection in comparison to the original approach by substituting logistic regression meta classifier with a lightweight neural network meta classifier. We analyze its performance and examine potential model architectures, advantages and drawbacks of using it instead of logistic regression. Furthermore, we additionally evaluate the entropy maximization and show that caution must be taken when using it in practice in order to ensure its effectiveness.

Keywords: computer vision, semantic segmentation, anomaly segmentation, entropy maximization, meta classification, autonomous driving systems, open-set environments

Sažetak

Nesigurnost segmentacije slike s obzirom na uzorke izvan distribucije

Jurica Runtas

Semantička segmentacija je zadatak računalnog vida u kojem se svaki piksel slike dodjeljuje u jednu od unaprijed definiranih klasa. Duboke neuronske mreže (DNM) su suvremeno rješenje tog zadatka i obično su trenirane tako da rade na zatvorenom skupu unaprijed definiranih klasa. U okruženjima otvorenog skupa, moguće je susresti anomalije, tj., semantički nepoznate objekte. Cestovna vožnja je primjer takvog okruženja u kojem je, sa stajališta sigurnosti, bitno osigurati da DNM može indicirati da operira izvan svoje naučene semantičke domene. Jedna metoda koja se koristi za tu svrhu je maksimizacija entropije uparena s korakom postprocesiranja zvanim meta klasifikacija koji povećava pouzdanost detekcije anomalija. U ovom radu, predlažemo pristup meta klasifikaciji koji značajno povećava pouzdanost detekcije anomalija u odnosu na originalni pristup tako što zamjenjuje logističku regresiju s jednostavnom neuronskom mrežom kao meta klasifikatorom. Analiziramo njegovu performansu i dajemo pregled potencijalnih arhitektura, prednosti i nedostataka korištenja neuronske mreže umjesto logističke regresije. Nadalje, dodatno evaluiramo metodu maksimizacije entropije i ukazujemo na to da je potreban oprez prilikom korištenja te metode u praksi kako bi se osigurala njena učinkovitost.

Ključne riječi: računalni vid, semantička segmentacija, segmentacija anomalija, maksimizacija entropije, meta klasifikacija, sustavi autonomne vožnje, okruženja otvorenog skupa