SEMI-AUTOMATIC CONSTRUCTION OF SUBSEA INSPECTION DATASETS USING DEEP CONVOLUTIONAL NEURAL NETWORKS

Olavo Argôlo Batista Sampaio

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientador: José Gabriel Rodríguez Carneiro Gomes

Rio de Janeiro
Março de 2020

SEMI-AUTOMATIC CONSTRUCTION OF SUBSEA INSPECTION DATASETS USING DEEP CONVOLUTIONAL NEURAL NETWORKS

Olavo Argôlo Batista Sampaio

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Orientador: José Gabriel Rodríguez Carneiro Gomes

Aprovada por: Prof. José Gabriel Rodríguez Carneiro Gomes
Prof. Mariane Rembold Petraglia
Prof. Priscilla Machado Vieira Lima

RIO DE JANEIRO, RJ – BRASIL
MARÇO DE 2020

Sampaio, Olavo Argôlo Batista

Semi-Automatic Construction of Subsea Inspection Datasets using Deep Convolutional Neural Networks/Olavo Argôlo Batista Sampaio. – Rio de Janeiro: UFRJ/COPPE, 2020.

XIV, 56 p.: il.; 29, 7cm.

Orientador: José Gabriel Rodríguez Carneiro Gomes

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2020.

Referências Bibliográficas: p. 47 – 49.

1. semi-automatic annotation. 2. dataset. 3. deep learning. I. Gomes, José Gabriel Rodríguez Carneiro. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

Dedico este trabalho à minha
família e meus amigos.

Agradecimentos

Dedico este trabalho a Fernanda, Gustavo, Juarez, Lucineia e Maria Julia, que sempre me apoiaram e me incentivaram a começar estes estudos. Dedico também a todo restante da minha família. Vocês são as pessoas mais importantes do mundo para mim.

Agradeço ao meu orientador, Prof. José Gabriel, por me guiar nessa jornada acadêmica.

Agradeço à equipe formada pela Prof. Mariane Petraglia, Felipe Vianna, Karen Olinto e Roberto Estevão pelo trabalho e dedicação e por permitirem minhas contribuições para o projeto.

Agredeço à COPPE/UFRJ e ao CNPq pelo apoio financeiro ao longo deste curso de pós-graduação.

Agradeço ao Roberto Estevão por ter originalmente indicado o artigo sobre o qual esse trabalho se baseia e por outras valiosas dicas.

Agradeço aos meus colegas Andrei Lenine, Adriano Fonseca, Leonardo Mazza, Pedro Bandeira, Pedro Cayres, Renan Rotunno e outros. Obrigado pelas proveitosas discussões e apoio moral ao longo desses anos.

Agredeço aos demais que me ajudaram de alguma forma durante essa etapa.

Sou muito feliz por todo o apoio recebido e desejo usar os conhecimentos que adquiri para causar um impacto positivo na sociedade.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

CONSTRUÇÃO SEMI-AUTOMÁTICA DE BASES DE DADOS PARA INSPEÇÃO SUBMARINA USANDO REDES NEURAIS CONVOLUCIONAIS PROFUNDAS

Olavo Argôlo Batista Sampaio

Março/2020

Orientador: José Gabriel Rodríguez Carneiro Gomes

Programa: Engenharia Elétrica

A inspeção de dutos submarinos requer que especialistas analisem muitas horas de vídeos buscando eventos relevantes nos dutos, uma tarefa demorada e cara. Usando modelos de aprendizado profundo para classificação de imagens, pode-se acelerar a busca por eventos em vídeos, substituindo ou reduzindo o trabalho necessário dos especialistas. Para treinar tais modelos, que podem ter milhões de parâmetros treináveis, bases de dados com muitos exemplos são necessárias. Elas devem ser construídas através da anotação de eventos nos vídeos. Essa tarefa, se feita manualmente por anotadores humanos, é lenta, requer muito esforço humano e é de difícil escalabilidade. Este trabalho adapta e utiliza um método de anotar imagens usando força de trabalho humana em conjunto com redes neurais profundas de uma forma sequencial e iterativa. Esse método é usado para anotar 146 vídeos de inspeção submarina e construir uma base de dados com 457 mil imagens para uma tarefa de classificação hierárquica de três níveis. Essa base de dados é comparada com uma construída apenas usando anotadores humanos, usando ambas para treinar e avaliar modelos classificadores, onde a nova base permite que modelos tenham melhor desempenho em 10 de 14 testes, se comparados aos treinados pela base concorrente. O método também produz uma amplificação de esforço de anotação de 45:1, no melhor caso, e 13:1, no pior, além de ser estimado que ele permita uma que imagens sejam anotadas 4,3 vezes mais rápido do que o método manual anterior.

Palavras-chave: base de dados, aprendizado profundo, anotação automática, anotação semi-automática, ferramenta de anotação.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

SEMI-AUTOMATIC CONSTRUCTION OF SUBSEA INSPECTION DATASETS USING DEEP CONVOLUTIONAL NEURAL NETWORKS

Olavo Argôlo Batista Sampaio

March/2020

Advisor: José Gabriel Rodríguez Carneiro Gomes

Department: Electrical Engineering

Undersea pipeline inspection requires that specialists analyze many hours of video searching for relevant events, a time-consuming and expensive task. Using deep neural models for image classification can accelerate event discovery in videos, substituting or reducing the work required from the specialists. To train such models with millions of parameters, large labeled datasets are required. The datasets must be built by annotating the events on videos. If that is done by human annotators, it becomes a task that is slow, time-consuming and difficult to scale. This work explores and adapts a method of annotating images using human effort in tandem with deep neural networks in an sequential, iterative manner. The method is used to annotate 146 videos of undersea inspection and builds a dataset of 457 thousand images to solve a hierarchical classification task with three levels. This dataset is compared to a dataset built using only human effort by using both to train and evaluate classifier models. The new dataset allows a model to achieve best performance in 10 out of 14 tests in comparison to the performance of models trained from the previous dataset. The method also produces an annotation effort amplification of 45:1 in the best case and 13:1 in the worst, and is estimated to allow the new dataset to be annotated 4.3 times faster than the previous, manual method.

Keywords: dataset, deep learning, automatic annotation, semi-automatic annotation, annotation tool.

Contents

List of Figures

List of Tables

Acronyms

CSV Comma-Separated Values

LSUN Large Scene UNderstanding (SUN)

ROV Remotely Operated Vehicle

SUN Scene UNderstanding

Chapter 1

Introduction

Modern deep neural networks can achieve state-of-the-art results on classification tasks over many different fields. However, the models used have millions of parameters and require training on large labeled datasets. The creation of such datasets is an expensive process that requires many hours of often specialized human effort. This annotation cost grows linearly with the dataset size, but the cost of building a larger dataset is offset by allowing models to achieve better performance.

The experience of developing deep learning models applied to undersea pipeline inspection, detailed in Section 1.4, showed that the most time-consuming task in the project pipeline is the construction, correction, and maintenance of the dataset required to train the models. It demanded the concerted effort of several annotators over several months to annotate each part of the dataset.

Thus, this work explores a method to facilitate the annotation of image datasets applied to the task of image classification on pipeline inspection, using deep neural networks in tandem with human effort. In particular, the dataset is sourced from groups of videos and the associated classes have hierarchical relationships. These characteristics must be taken into consideration during the development of the annotation procedure.

1.1 Motivation

Pipeline inspections are done by oil companies to survey a predetermined extension of a pipeline, searching for relevant features for a given inspection. These features can be known markers such as pipe connectors, riser floaters, oil well "Christmas trees" or written markings on the pipe. They can also be features that characterize problems in the pipeline. For example, damage to the pipe structure, a leakage, previously-made repairs, a corroded electrolytic anode, a pipe torsion, kink or knot. Anything that indicates current or future degradation of pipeline integrity may be identified in an inspection. However, such inspections are carried out remotely and

must be later analyzed by company specialists. The inspections produce a large amount of video footage, which must be analyzed in search for the relevant features or events that the company deems interesting. This means encumbering several highly specialized workers: an expensive and inefficient solution.

For this reason, creating a method to quickly identify which video segments harbor relevant events would allow the experts to use their time more efficiently, focusing their attention only on preselected videos or video segments. This procedure would facilitate the creation of classification datasets for this task, thus improving their availability and size which, in turn, tends to improve model performance. Thus, it is beneficial to explore methods for facilitating image labeling and dataset creation.

This project was developed using the Python programming language and the code is available at github.com/olavosamp/semiauto-video-annotation.

1.2 Objectives

This work seeks to apply a method of fast image annotation to create a classification dataset. To this end, it adapts the iterative labeling procedure described by [1], that uses deep-neural-networks-based annotation with humans in the loop, to annotate an image dataset for the task of pipeline inspection. This procedure is expected to amplify the labeling work done manually, by humans, and create a dataset that can achieve equivalent or better performance than a baseline dataset built using only human annotation while requiring less human effort to create. This allows the creation of bigger datasets in less time and for a smaller cost than the traditional solution of manually annotating all samples.

This semi-automatic annotation procedure using humans and classifier models is an iterative process that involves annotating a small number of difficult images of the dataset with humans, and annotating a large number of easy images with a neural network.

The process begins with a human annotating a small number of images of the dataset, that are then used to train a model. The model evaluates the remaining unlabeled images and separates the images with a high confidence score from the images with low confidence. Those with high confidence are considered annotated, while the remaining examples compose the new unlabeled set. Another batch of images is manually annotated from the unlabeled set and the model is trained from scratch with a larger training set. This loop iterates over the dataset until the set of unlabeled images is small enough to be manually annotated.

The automatic labeling done by the neural network tends to be of inferior quality than human labeling. It has a higher labeling error - the percentage of mislabeled

examples - and often selects less representative images than annotations made by human experts. However, the use of classifier models facilitates an increase in dataset size. It is expected that a large, but weakly annotated, dataset performs better than a smaller but carefully annotated one. This performance can be evaluated by training a model using each dataset and evaluating its performance in a common validation set. The superior dataset is the one that allows a model to perform better in a given measure when trained in that dataset.

Thus, this work seeks to adapt a procedure for faster dataset creation using human annotators and deep neural network models. It also seeks to create a dataset for the task of pipeline inspection using this procedure and compare it against a baseline dataset previously annotated. It is expected that a model trained in the new dataset can achieve performance comparable to one trained in the baseline dataset while requiring less time and human effort to build.

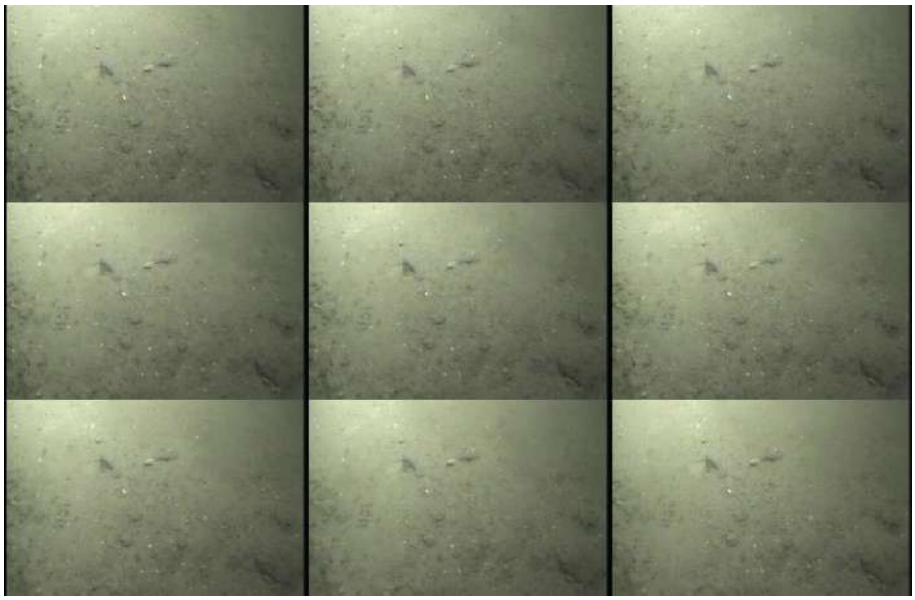


Figure 1.1: A sequence of nine sequential video frames. The time interval between each sequential pair of frames is 1 s.

However, a disadvantage is that each example of a semi-automatically annotated dataset is less valuable than an example of a manually annotated one. One factor is that they are often very similar, because of the constant-rate sampling used to obtain the video frames, as exemplified in Fig. 1.1. Therefore, it is expected that the former needs to be larger than the latter to achieve the same performance.

A remark about dataset availability: since the source videos are proprietary and often contain classified information about the nature of the pipelines or pipeline operations, the dataset created in this work is not made available to the public.

1.3 Classification Task

The classification task studied in this project is the identification of events in undersea pipeline inspections via image classification.

Related tasks are to identify whether a video frame contains a duct, whether a duct contains an inspection event or event features, and the classification of a given event in a set of five classes. These tasks are organized in three sequential, hierarchical, levels. Each level classifies its inputs and feeds the positive examples to the next level.

1.4 Previous Work

The classification task tackled in this work was already subject of (unpublished) research made on a previously developed project, whose results are used in this work as a baseline for comparison. The previous project's goal was to build a classifier system to identify events in pipeline inspections. The system receives a video as input and identifies which frames depict a relevant event. This resulted in the development of a labeling tool, the annotation of image datasets, and the training of classifier models for several tasks.

In the current work, a dataset is created for the same tasks, and the same classifier model architecture is used. The difference is that a new method for building the dataset is explored. It uses the same set of videos as an image source but selects which images will compose the dataset and annotates them differently.

In the previous work, an image dataset – the reference, or baseline dataset – was annotated largely by hand using two methods over a period of two years. Part of the dataset was sampled using variable-rate sampling that prioritizes interesting video segments flagged by human annotators. Another part was sampled uniformly and then human annotators labeled and selected which images would be included in the dataset. Each of these methods was used to annotate part of the source videos or used for the correction of already labeled videos. These methods are described in Section 2.1, and the dataset, in Section 3.2.

1.5 Description

This work is organized in four more chapters. Chapter 2 introduces the theoretical basis of the methods and techniques used in this work, as well as contextualizes the methods used in previous projects that led up to the current work; Chapter 3 details the classification task, iterative annotation process, classifier training, creation of the validation set, and how the dataset comparison is carried out; in Chapter 4 the

results of the iterative annotation, created dataset, model comparison, labeling error analysis, and annotation speed estimates are discussed; and Chapter 5 contains the discussion of results, achieved and missed goals, final remarks, and ideas for future work. Appendix A contains additional data about the iterative annotation process that would take too much room in the previous chapters.

Chapter 2

Concepts of Data Annotation

This chapter briefly discusses select topics that are used in this work. It aims to be an introduction to those themes, and to allow the reader to have an understanding of the methods used and a starting point for further study, if desired. Throughout this work, it is assumed that the reader has basic familiarity with machine learning, particularly deep neural networks. If not, [2] and [3] are good learning materials.

2.1 Annotation Methods

There are different ways to annotate examples and register the data during the process of building a dataset. As this project deals with the construction of image datasets sourced from videos, two methods used to build the reference dataset are discussed in this section. They should allow a human annotator to easily analyze and classify groups of images spending as little effort as possible, as well as registering this information in an accessible way.

The methods are discussed according to the chronological order in which they were used in this and preceding projects. The goal of this section is to contextualize the construction of the reference dataset, used as a baseline for the new dataset. The methods are: the annotation of video intervals and the sorting of image folders. The use of annotation tools is discussed, briefly, in this section, and in detail in Section 2.2.

The first method, video interval annotation, consists in associating video time intervals with labels. The annotator analyzes the source video and identifies a relevant class feature. Then, they determine the initial and final video timestamps where the desired feature lies. The defined time interval must be continuous and represent well the target class. When the annotator is done registering the desired intervals, this information is used to sample images from the video, with each annotated time interval associated with a class (or group of classes) and yielding images labeled in the same way. Figure 2.1 represents the result of this annotation method saved to

```
VideoName,Class,StartTime,EndTime,Id
20161101215100437@DVR-SPARE_Ch1.wmv,tubo,000000,002226,01
20161101215100437@DVR-SPARE_Ch1.wmv,nada,002226,002423,02
20161101215100437@DVR-SPARE_Ch1.wmv,tubo,002423,002620,03
20161101215100437@DVR-SPARE_Ch1.wmv,nada,002620,002907,04
20161101215100437@DVR-SPARE_Ch1.wmv,tubo,002907,003000,05
```

Figure 2.1: Example a file used to register image labels. This CSV file holds information of video interval annotations and their corresponding labels. The second to fourth columns represent the video interval's label (class), start time, and end time, respectively.

a CSV file.

This method is flexible as it allows the image sampling rate to be chosen independently from the class labeling, and it helps the annotator label groups of similar images that share a temporal immediacy. Its disadvantages include a slow annotation speed and relatively high mislabeling chance. The first is due to requiring the annotator to register subjectively and precisely the time interval, the second arises from the medium used to register the labeling data in the project. The annotator is required to manually register the information of each time interval on a file, thus writing several fields of information in a time-consuming and error-prone process.

This method was used to build part of the reference dataset described in Section 3.2 but was later discarded in favor of manual sorting of image folders. The main reason for this is that the former method was harder to re-annotate since the individual images are undefined until video sampling is performed. Besides, dataset creation is more cumbersome (than in the latter method), since it relies on auxiliary programs to interpret the annotation data and sort the images into classes.

The second method explored consists simply in labeling each example individually. This method requires sampling the videos before the annotation, obtaining a set of unlabeled images. A human annotator then analyzes each image individually and assigns it to a class. The labeling information must then be registered in some way for sharing or use in other steps.

It is a way of annotating images that consists in organizing the dataset examples in folders according to their class. Each image is moved to a folder corresponding to its associated label after annotation. This is advantageous because the medium used to register the annotations is also a common way to organize the examples in a dataset for sharing or model training. However, if the images are moved after annotation, their class information may be lost.

The folder file structure can be saved in a separate file after the dataset is annotated. This "manual sorting of image folders" method was used to annotate part of the reference dataset and required minimal setup, which allowed annotators to contribute more easily, and made the annotation process more interpretable.

Annotation tools may be used to enhance either method. They can be any software that helps human annotators label data. They may promote faster annotation, less labeling error, standardized annotations or any other benefit to the annotation process.

2.2 Annotation Tools

Annotation tools are widely used to help annotate datasets. They are designed to assist a human annotator label data, and to use one in a project, it must either be designed for a specific problem or be selected among one of several existing, freely available options. Some examples are discussed in this section, and then the tool used in this project is described.

Designed for image and object annotation, LabelMe [4] is a simple web-based tool that allows collaborative labeling and label sharing in a freely accessible online platform. It displays one image at a time and the user is able to annotate objects in the scene using a polygonal drawing tool. The platform was used to create the dataset of same name.

Chimera [5] is an image annotation tool that combines machine learning and rule-based classifiers to help the annotator handle large amounts of retail product images. Crowd-sourced annotation evaluation is used to analyze and give feedback to the hand-crafted classification rules. It tackles the challenges of variable human resources scale, uneven annotation loads, and changing data distributions.

In [6], a video annotation tool named *interactive Video Annotation Tool* (iVAT) is presented. It supports manual, semi-automatic and automatic annotation, integrating computer vision algorithms in an incremental learning framework to facilitate annotation. The tool is designed for object segmentation and allows a user to track objects throughout a video segment. It includes annotation quality statistics, annotation timeline visualization, video segment overview, labeling categorization, and three bounding box drawing options.

A web-based open-source system for video annotation that supports image, bounding box and polygon annotation with several drawing options is detailed in [7]. It also uses an R-CNN based object detection model to track objects along frames and propagate bounding box annotations to help the user. The tool is applied to create a driving dataset, the BD100k dataset.

In the previous project, a labeling tool was developed and used to annotate part of the reference dataset. Developing a tool customized for the project allowed more control over its characteristics. It is a desktop program for the Windows operating system, developed using the Python module PyQt5. It displays images individually to the user and allows them to assign a class to the image, skip to the next image,

Figure 2.2: Screen capture of the labeling interface in use. The top and bottom portions of the image are omitted. The rightmost portion of the screen displays buttons used to assign labels for the three classification levels described in Section 3.2. The user selected the classes Duct, for level 1, and Not Event, for level 2.

save and load annotation progress, jump to a specific image, and move annotated images to folders according to their assigned classes. The user can interact with the tool through a visual interface, using mouse, keyboard, and perform navigation and labeling using hotkeys. It saves annotations in CSV files to allow multiple labeling sessions, and for easy sharing. This automates part of the annotation process, thus reducing the cognitive load of each annotation task, and focusing the annotator attention on a single, straightforward task of labeling the image among a pre-selected list of classes. The labeling is made for all three hierarchical levels at the same time. The interface suggests, but it does not enforce, the hierarchical relationships for the user. Figure 2.2 shows a view of the labeling interface during use. It allows the user to annotate more labels than the nine classes used in the current work (They are duct, event, anode, buried, damage, flange, and repair.), among all levels. This allows more flexibility to select only the relevant labels and discard classes that are ambiguous or have too few annotated examples.

As detailed in Section 4.3.2, it is estimated that using the interface led to an increase of up to 100 times in annotation speed during the construction of the reference dataset.

2.3 Class Imbalance Treatment

When dealing with a dataset with an unbalanced class distribution, a situation with one class with many examples and another with a few ones, attempting to train a classifier using that dataset encounters difficulty. The model has to learn adequate parameters to define class separators, but the larger class has a greater effect on the loss function and influences the learning process to favor itself. The trained model is then not able to discriminate between examples from both classes as well as if it was trained with samples of the same size. In this case, the best solution would be to obtain more examples for the smaller class. However, this is not always feasible. Discarding examples from the larger class also results in the two classes being balanced, but, since obtaining training examples is hard, discarding them is not ideal. Two other main solutions are available: sampling each class differently so that each training mini-batch is balanced, and modifying the way the cost function weights examples of each class.

The first option (sampling differently) changes the way the dataset is presented to the model. If each mini-batch is made to harbor a balanced dataset sample, with the same number of examples for each class, none influences the learning process to the detriment of the others and the problem is solved. Usually, to select examples for a mini-batch during training, the training set is sampled randomly and uniformly. Selected examples are either taken out of the set or not, and the process is repeated to select the next mini-batch of examples until all images in the training set are seen by the model. To balance class contributions, the sampling is changed so that each element of a class has a probability inversely proportional to the class size. Therefore, on average, each mini-batch has the same number of examples of each class.

The second option (cost function weighing) consists of weighing each class differently as their examples are evaluated by the cost function. If the weights are selected as to counteract the ratio in class size, the larger class numerical advantage is eliminated.

Both solutions are mathematically equivalent and neither completely solves the problem, as there is still a class with few examples. From the point of view of the training process, if either one of the above methods is implemented, then each of the rarer examples are worth more. But the class still lacks diversity and the model may not be able to generalize well on other samples.

The method used in this work is the cost function weighing, as it is simpler to implement. It is described in this section. The other method can be used interchangeably with minimal impact on the model's parameters.

The error loss function, or cost function, is calculated during model training for

each mini-batch (of size M) as the sum of the values of the cost function J, given as

$$J(\mathbf{X}) = \sum_{i=1}^{M} J(\mathbf{x}_i) \tag{2.1}$$

where $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M] \in \mathbb{R}^{N \times M}$ is the input matrix with M examples and $J(\mathbf{x}_i)$ is the cost function applied to a single example.

To change the impact of each training example on the model's parameters, the error is modified to a weighted sum,

$$J(\mathbf{X}) = \sum_{i=1}^{M} w_i J(\mathbf{x}_i) \tag{2.2}$$

where w_i is the class weight associated with $\mathbf{x}_i$ such that, if M_j is the number of examples of class C_j,

$$w_i = \frac{1}{M_j}, \text{ if } \mathbf{x}_i \in C_j. \tag{2.3}$$

However, for significant differences in class size, this method can lead to weights that are too small and impair learning of the larger classes. In this work, a modified version of the original formula is used, to avoid large differences in class weights. The weights are given by

$$w_i = \sqrt{\frac{1}{M_j}}, \text{ if } \mathbf{x}_i \in C_j. \tag{2.4}$$

The most evident benefit of this method is smaller variance of the model classification results. Discouraging the class weights from achieving extreme values mitigates abrupt variations on the model parameters brought by changes in mini-batch order or initialization during training.

Chapter 3

Semi-Automatic Annotation

In this chapter, the methods used to create and evaluate the semi-automatically annotated (also referred to as semi-automatic, or *semiauto*) dataset are discussed. This project annotates a dataset with a combination of human and computational effort. The classification task tackled is the identification and sorting of a set of inspection events in an undersea pipeline. Examples of inspection events are a damaged or repaired pipe segment, and a bolt flange that is part of the pipe. In this work, duct, pipe segment or pipe section are used interchangeably to represent a part of a larger pipeline contained in a captured image.

The dataset was derived from a set of videos of undersea pipeline inspection. Each video was sampled at a rate of one frame per second. These sampled images formed an initial unlabeled dataset, upon which an iterative annotation process was applied. The process uses manual and automatic annotation to label all samples in classes, following a hierarchical relationship. The manual annotation is made by human effort, while the automatic annotation is provided by a deep neural network classifier. This system allows fast labeling of large numbers of samples, thus reducing the required time and human effort while maintaining an acceptable level of annotation precision and performance on the validation set.

3.1 Video Dataset

The existing video dataset contains 143 videos of undersea pipeline inspections recorded over several years. They are recorded in color but vary in location, nature of the activities carried out, lighting, sand occlusion, presence of interfering sea life, presence of occluding machinery, pipeline condition and type, blur, video quality, resolution, and format. Table 3.1 shows the variation in resolution among the dataset images.

The recordings were made by a subsea Remotely Operated Vehicle (ROV). Each ROV has a single camera of varying characteristics and records a set of videos. The

vehicle hovers over the target pipeline and records the entire inspection run. It eventually stops on pipe segments that the remote operator deems relevant for the inspection, but can also record the undersea floor, empty water, and otherwise uninteresting features.

Table 3.1: Source video count according to resolution. Resolution is represented in pixels, as (width × height).

Resolution	**Number of videos**
352 × 240	2
352 × 288	1
352 × 480	49
640 × 480	18
704 × 576	30
720 × 480	29
720 × 576	14

The videos are treated only as a source of still images. Each frame in a video is regarded independently and their time-related nature is not exploited in this work. Images extracted from the same video tend to be similar and may thus contain the same kind of target feature or event. As such, images from a single video, or a set of similar videos, are allocated together in a single training or validation set to avoid bias.

3.2 Previously Developed Image Dataset

This classification task has been the subject of previous work. In [8], a supervised dataset was annotated for a similar task, but on another type of pipeline with a somewhat different domain. In [9], a supervised dataset was annotated through a combination of variable-rate sampling and manual image annotation and selection. This approach tackled a simpler task than the current work: identifying whether an image contained a pipe segment or not. In this, it the approach yielded 81.7% average class accuracy. Further work approached the task of classifying an image, already tagged as a pipe segment, as containing or not an inspection event, and then classifying the event type. This three-stage hierarchical approach has yielded good results and is the process used in this project.

This work benefits from the previously devised approach for three-stage hierarchical image classification. It splits the classification task into three simpler tasks or levels, following a sequential structure where the outputs of a given level are the inputs of the next one.

First, an input image is classified as containing a pipe segment or not. The positive outputs of that level, duct images, are then fed into the second level, which

classifies them as either containing inspection events or not. Finally, the last level receives images of events and classifies them into one of five possible classes: galvanic anode, duct damage, duct repair, bolt flange, and buried duct.

Figure 3.1: Examples of negative examples at hierarchical Level 1.

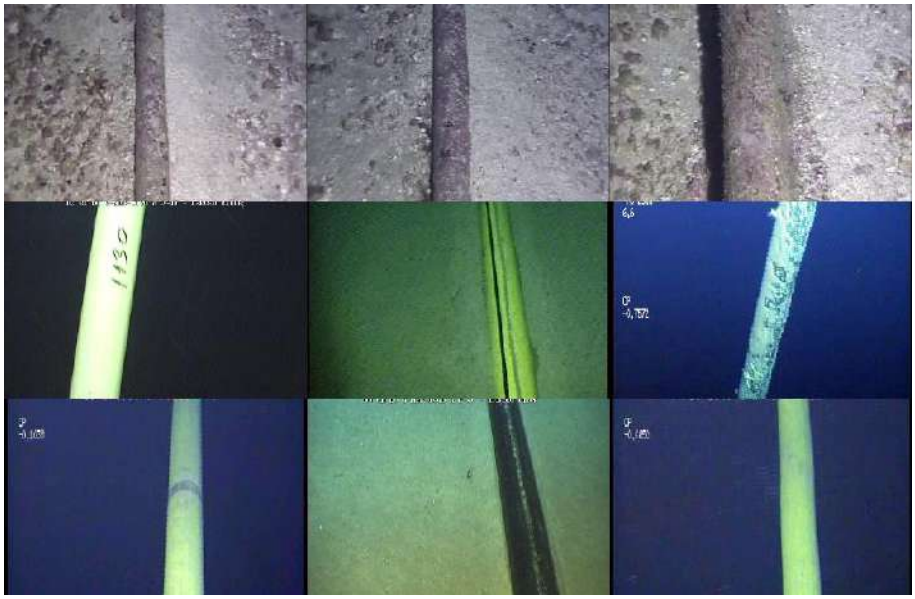


Figure 3.2: Examples of negative examples at hierarchical Level 2.

The following list describes the classes at all three levels, and Figs. 3.1 and 3.2 present image examples for levels 1 and 2.

- *Level 1* (binary):
  - *Duct*: positive class. Presence of a recognizable portion of a single pipe segment in the image;

  - *Not Duct*: absence or images from scenes with confusing information, or many cluttered objects, are included in this class.

- *Level 2* (binary):
  - *Event*: positive class. Presence of an inspection event on a pipe segment. The event corresponds to a particular class at Level 3 or, otherwise, it may correspond to another interesting event set that has not yet been defined as a class at the next hierarchical level;
  - *Not Event*: absence of an inspection event on a pipe segment. The duct does not have any interesting or relevant features.

- *Level 3* (multiclass):
  - *Anode*: presence of a galvanic anode on the pipe segment. This is usually a white thick band over the duct, often near a bolt flange;
  - *Buried*: the pipe segment present on the image is partially buried (occluded by sand), but still recognizable as part of the pipeline. A segment that is completely buried is not detectable in the image and thus not included in this class;
  - *Damage*: presence of any significant damage to the pipe. It may be a scratch or tear of the outer rubber coating, a segment with its outer coating scraped off and so that the inner metallic layer is exposed, an instance of the metallic layer pushed to the outside of the duct or any other apparent damage;
  - *Flange*: presence of a bolt flange on the pipe segment;
  - *Repair*: presence of a repair previously made on the duct. Repairs are usually characterized by a tape wrapped over a part of the pipe, but they can appear in other forms. What problem was being repaired is not important in this case.

This hierarchical procedure splits a single, difficult task, that is, multiclass classification on a broad domain, into easier tasks. Each subsequent level must solve a classification problem inside a smaller domain than the immediately previous one. The second level receives only images of pipe segments, thus skipping the effort of determining whether unrelated machinery is part of the duct or not. The third classification level processes images of inspection events only. The procedure is represented in Table 3.2.

In previous work, one dataset for each level was created through manual annotation, for use in a hierarchical procedure as described. Each dataset was used to

Table 3.2: Diagram of the three-level class hierarchy. Classes in bold are the positive or multiclass options at their respective levels.

Level 1	**Level 2**	**Level 3**
Not Duct		
Duct	Not Event	
	Event	**Anode** **Buried** **Damage** **Flange** **Repair**

train one deep neural network responsible for the classification of images on a single level. This achieved acceptable results, which are discussed in Section 3.6, and will be used as a baseline performance.

Dataset size and class distribution for each level are detailed in Table 3.3. Each level may benefit from images that have been already annotated at a previous level, and it also adds new images to the dataset. This means Level 2 and 3 image counts do not necessarily match up with the positive image counts at their immediately previous levels, as would be expected if all datasets were derived from the same group of images.

Table 3.3: Reference (manually annotated, baseline) dataset class distributions for all three classification levels. Total counts for each level are in bold.

	Class	**Image Count**
Level 1	Duct	1794
	Not Duct	784
	Total	**2578**
Level 2	Event	1020
	Not Event	1169
	Total	**2189**
Level 3	Anode	480
	Buried	1109
	Damage	505
	Flange	771
	Repair	458
	Total	**3323**

The annotation process is different for this reference dataset and the new semi-automatically annotated dataset. There are two main differences. First, the video sampling rate is different. In the case of the reference dataset, the images are hand-sampled by human annotators, using the annotation interface or not. In the semi-automatic dataset's case, the videos are sampled uniformly and, then, all images of a given class are selected by the classifier model. This may lead to many images

being very similar, which is generally not desirable but is an acceptable trade-off.

Second, the manual annotation criteria are different: for the reference dataset, the annotators were generally instructed to annotate an image as part of a class if its main element is an object or feature of that class; for the semi-automatic dataset, an image is labeled as part of a class if that class is identifiable within the image, even if it is not the principal element. This difference occurs because each image in the reference dataset is annotated only once and is not subject to the multiclass translation process, described in Section 3.5. The different tools and methods used to annotate the datasets may also exacerbate the differences between them.

3.3 Iterative Annotation

To quickly annotate the 457 thousand images extracted from the video dataset, a semi-automatic iterative annotation procedure is proposed. It is an adaptation from the procedure used in [1] to annotate a large number of unlabeled images obtained from web searches and to create the Large Scene UNderstanding, or Large SUN (LSUN), Dataset [10]. The main differences between [1] and the present work are:

- Classification Task: [1] annotates a dataset for binary classification tasks, while the current work annotates two datasets for binary and one for multiclass classification. In the latter case, this difference is reconciled by annotating each class separately as binary tasks and then merging them into a multiclass dataset;

- Image sources: the unlabeled images used in [1] are obtained from bulk Internet searches using image search engines, while the present work obtains its images from frames sampled from a specialized, proprietary, video dataset;

- Image relationships: the classes in the LSUN dataset are composed of unrelated images that are independently sampled from available search engine image results. Images used in the current work are sampled from videos and are thus intrinsically related. Frames sampled in short sequence and from the same video tend to be very similar, while frames sampled from different videos tend to be less similar;

- Scale: while [1] tackles a large-scale problem of 59 million images to be mapped into 20 classes, this project starts with 457 thousand images and builds datasets of equal or smaller size.

3.3.1 Iteration Step-by-Step

The procedure starts with the creation of an unlabeled set of images to be annotated. To this end, each one of the 143 available videos was sampled at a rate of one frame per second. The resulting images compose an unlabeled dataset of 457 372 images detailed in this subsection. Figure 3.3 illustrates the iterative annotation process.

The iterative loop starts with the manual annotation of a small fraction of the unlabeled dataset, such as 1%. In this case, 4 676 images were manually annotated, a little over 1%. Every instance of manual annotation labels each image for every level of classification. As the classifier models used are independent, an image annotated at one level can also be used in the following. After the manual annotation, the currently annotated set is split into training and validation datasets. This is done with a uniform random split of 80%/20%. Then, a classifier is trained in the training dataset and is, then, used to evaluate every image of the validation set. Each validation image is presented to the trained classifier, which gives a score to the input image.

Next, these scores are normalized and two thresholds are defined for the evaluation scores: an upper threshold, above which images are considered positive class examples and a lower threshold, below which they are considered negative class examples. Images scored between these thresholds are still considered unlabeled. The thresholds are selected on the validation set. The upper threshold is selected such that 99% of the images above it are true positives. The lower threshold is selected such that only 1% of all positive images are below it.

Then, all unlabeled images are evaluated by the trained classifier and classified as positive examples, negative examples or unlabeled, if their scores are above the upper threshold, below the lower threshold or between the two thresholds, respectively. The images labeled by this process are considered already annotated and are removed from the unlabeled set. They are not used to augment the training set, as the size of the automatically labeled images set can become much larger than the size of the manually annotated ones, which might introduce or increase classifier bias. Also, the class distribution of the annotated images is different from the distribution of the images remaining in the unlabeled set, which could further skew the classifier.

Finally, a new iteration begins. The remaining unlabeled images of the last iteration make up the unlabeled set for the current iteration, and a fraction (typically 1%) of the images in the unlabeled set is manually annotated. All images manually annotated in all iterations so far form the labeled set, which is used to train the classifier again from scratch.

The process ends when the remaining unlabeled set is small enough to be completely annotated through manual labeling. In this work, the iteration loop ended

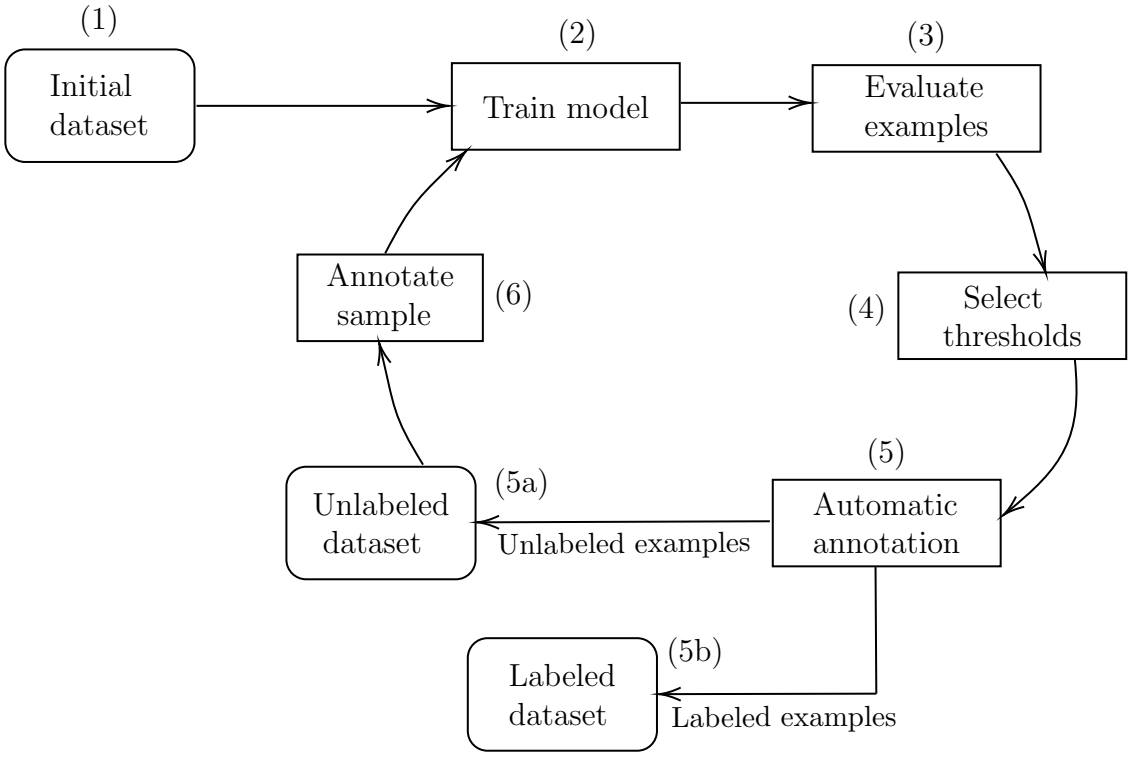


Figure 3.3: Diagram of the iterative annotation process. (1) The process starts with an initially annotated set of images; (2) then, a classifier model is trained on the annotated set; (3) the yet unlabeled examples are given a score by the trained model and, (4) based on the validation set, decision thresholds are selected to ensure 99% precision and 99% recall; (5) unlabeled images with scores greater than the upper or smaller than the lower threshold are considered labeled and compose the labeled dataset (5b), images with scores lying between thresholds are still unlabeled (5a); a sample of the unlabeled set is manually annotated (6) and incorporated to the dataset used for training (2).

if the unlabeled set size reached 4 thousand images or less.

3.3.2 Details

Other implementation details of the iterative annotation are discussed in this subsection.

Classifier

The classifier used for automatic image annotation is a deep neural network using an adapted version of the Resnet18 architecture [11] and using *cross-entropy* loss function, weighted as to mitigate class imbalance, as discussed in Section 2.3. The network architecture adaptation consists of changing the fully-connected layer to two neurons, corresponding to the positive and negative classes. The model was initialized with available pre-trained weights, obtained by [11] by training the model on the ImageNet dataset [12] [13]. The initialized network is then trained on the manually annotated dataset. All network layers are trained, including the convolutional and fully-connected layers. In each new iteration, the model weights are reset to their pre-trained values, so that each trained model is not dependent on the previous one(s).

The training and validation set are split at random, selecting 80% and 20% of the dataset for each set, respectively. As the dataset is composed of frame captures from a group of videos, images sourced from the same video and potentially very similar may end up in different sets. Usually, this is undesirable, as the classifier is evaluated for its capability of generalization outside of the training set. Overfitted models would be unfairly well evaluated because of the contamination of the training set with validation set examples. However, in this case, it does not matter if the results are misreported, because the classifier goal is to correctly label the given set of examples, instead of being able to generalize on the entire dataset.

The first five iterations of the first hierarchical level trained the model for 500 epochs, with a mini-batch size of 256. This number of epochs is higher than necessary. It is used in order to gauge the model behavior during training. In the following iterations of that and subsequent levels, 150 epochs are used for training as, in the first-level training runs, the model stops improving before that point. The mini-batch size used made the process slightly faster than lower values. The optimizer used is the *Adam* optimizer [14] with learning rate $\alpha = 10^{-3}$ and $\beta_1 = 0.9$, $\beta_2 = 0.99$. The model that yields the smallest validation loss among all epochs is selected to be used in the next stages.

An image is evaluated by the network by passing it as input to the selected model and obtaining an output vector, which is an input vector for the *cross-entropy* loss

function. In this case, it is a two-dimensional vector, as all annotation procedures are transformed into binary classification problems, as discussed in Section 3.5. This vector is given as input to a *softmax* function, whose output is a score value for each class. The value for the positive class is recorded as the image evaluation score, which is used for annotation in the next stage.

The model was implemented using the Pytorch [15] framework. It provided the computational framework for model training and evaluation, deep neural network model implementations, and data augmentation utilities. The hardware used for training the network was a local machine with the following specifications: 64 GB of RAM DDR4/3000 Kingston HyperX, NVidia GeForce GTX1080/11 GB Ti video card, 3.6 GHz Intel Extreamer X99M Core i7 6850 CPU with 12 cores, SSD M2 Samsung EVO and HD WD REd SATA III hard drives.

Image Preprocessing

Input images are preprocessed and subject to data augmentation using some of the transformations used in [11], as follows. In the training phase, images are subjected to a random crop of resolution 224×224 pixels, then they are flipped horizontally with a 50% chance and, finally, normalized with the mean and standard deviation of ImageNet images, per channel. For the validation phase, images are resized to a resolution of 256×256 pixels, center cropped to 224×224 pixels and normalized in the same way as in the training phase.

Threshold Selection

To perform the automatic image classification, upper and lower thresholds must be selected. These are used to classify the unlabeled images into positive and negative examples and must be chosen based on a sample that follows the same class distribution as the unlabeled set. The sample used is the same validation set selected from the 80/20% random split used for training the model. If the unlabeled set follows the validation set distribution, the model is able to correctly label its images with high confidence. If not, it will have difficulty classifying the unlabeled examples.

Having selected the validation set, its images pass through the inference process of the trained network and each is assigned a score. The images are ordered by their scores and classified as a labeled positive, a labeled negative or an unlabeled class example. Images with an evaluation score higher than the upper threshold are classified as positive examples and images with a score lower than the lower threshold are classified as negative examples. These labeled images are then aggregated to the automatically labeled set, but not used in future iterations.

Images with a score between both thresholds are still considered unlabeled and

compose the unlabeled set for the next iteration. Selecting adequate thresholds makes it so only images that the model classifies with high confidence are automatically labeled, while images that are still challenging to the classifier move on to the next iterations. However, high model confidence does not guarantee a correct classification, as the network can, for example, attribute high score to an image belonging to a negative class, if it cannot generalize well enough for that family of images.

The threshold selection step is as follows. The upper threshold is a value $T_{upper} \in (0, 1)$ such that $\frac{P_{upper}}{I_{upper}} \geq 99\%$, where P_{upper} is the number of true positives images with a score greater than T_{upper} and I_{upper} is the total number of images with a score greater than T_{upper}. This means that the precision of the automatically labeled images is made to be greater than 99%. Likewise, the lower threshold is a value $T_{lower} < T_{upper}$, $T_{lower} \in (0, 1)$ such that $\frac{P_{lower}}{P_{total}} \leq 1\%$, where P_{lower} is the number of true positives images with a score lower than T_{lower} and P_{total} is the total number of true positives in the validation set. In this case, this selection guarantees that the recall of the automatically labeled set is greater than 99%. The labeling precision and recall metrics can be modified by choosing different thresholds. This carries a trade-off, as stricter thresholds will result in better labeling metrics, but will annotate a smaller number of images each iteration, requiring more iterations to annotate the dataset. In [1], the upper threshold was selected aiming for an annotation precision of 95%, allowing for faster annotation while maintaining acceptable precision and recall. In the current project, a precision of 99% is chosen because the smaller dataset size allows for it to be annotated in a reasonable time frame, even while using stricter labeling thresholds.

The algorithm used to select the thresholds is a simple brute force sweep of possible values. To select the upper threshold, values are tested starting from 1.0 and descending until the condition of 99% or more ground truth positives above the upper threshold is satisfied or the bound 0.0 is reached. Likewise, to select the lower threshold, the sweep starts from 0.0 and tests values for the greater threshold that still satisfies the condition of less than 1% ground truth positives under the lower threshold or the 1.0 bound is reached. In both cases, the step size is 0.001. As speed is not a concern, in this case, this simple method suffices.

3.4 Manual Annotation

In this work, the manual annotation of the images during the semi-automatic annotation procedure was done by two human annotators. A verification step was carried out after the development of the dataset, using annotations made by another three human annotators to evaluate labeling error, described in Section 4.3.1. These vol-

unteers are not the same workers as the annotators for the semi-automatic labeling part. No annotator that participated in this project is an expert on undersea ducts, but they are trained to classify the data according to domain and project-specific demands.

All manual annotation in this project was done through a previously developed labeling interface to speed up human annotation, described in Section 2.2, and is used with minimal adaptation in the current work. This interface displays the images to be labeled sequentially, presenting the possible classes at each hierarchical level.

The reference dataset used in this project for comparison against the semi-automatic dataset, described in Section 3.2, was annotated using different methods, using slightly different criteria and by different annotators than the semi-automatic dataset. This may cause differences in class characterization, as some images that may be labeled as one class, following given criteria, may be labeled as another if following different ones. As detailed further in Section 3.6, two validation sets are used during dataset comparison to mitigate this.

3.5 Binary and Multiclass Treatment

As discussed in Section 3.2, the datasets for hierarchical levels 1 and 2 are annotated for binary classification tasks, while the dataset for Level 3 is annotated for a multiclass classification task. However, the semi-automatic annotation procedure is done through binary annotation. It creates one binary dataset for hierarchical Level 1, one for Level 2 and one for each of the five Level 3 classes. While Level 1 and Level 2 datasets are ready to be used for evaluation, a procedure to create a multiclass dataset from the Level 3 binary datasets becomes necessary.

A simple prioritization scheme is carried out. A ranking of the five Level 3 classes is established, and then, if a given image is annotated with multiple labels through the five binary datasets, it is labeled as the highest-ranked annotated class.

The priority ranking is chosen according to technical criteria. The most important inspection events in terms of urgency of identification and technical interest are ranked higher. The class priority list is:

1. Damage
2. Anode
3. Buried
4. Flange
5. Repair (for details on this low priority, the reader is referred to the comments about Table 4.4 in Section 4.1.3)

As an example, assume a given image has been labeled as belonging to the *buried*, *flange* and *repair* classes during the semi-automatic annotation. In the conversion to a multiclass coding, this image would be labeled as an example of the *buried* class, as it is the highest-ranking label in the priority list.

Note that this is only necessary for images automatically annotated by the classifier model. Manually annotated images are assigned a single class at hierarchical Level 3 and do not need to be converted to a multiclass coding.

3.6 Dataset Evaluation

Once the dataset is annotated in a semi-automatic fashion following the procedure described in Section 3.3, it must be evaluated in a comparison with the reference dataset. Each dataset is used to train a deep neural network classifier and is then evaluated by testing the trained network in a common validation set. The classifier's metrics in the validation set are used to compare the datasets. It is hoped that the semiauto dataset provides data necessary for the classifier model to be trained and then to achieve performance equal or acceptably worse than the performance obtained by training the same model in the reference dataset.

This evaluation classifier may use the same network architecture as the classifier used in the automatic labeling phase, but it neither uses their weights as a starting point nor is trained for the same task. The model used in this step is initialized using ImageNet pre-trained weights. The task of the classifier used in the semi-automatic annotation step is to correctly label a fixed set of images, generalization to other image sets is not required. However, the task of the classifier used in the current evaluation step is to classify images of a group, given a small sample of that group. Generalization outside the training set is required at this step, unlike in the labeling step. Thus, the training and validation sets must be carefully selected in this step as to not unfairly boost the models results.

Each hierarchical level is evaluated independently, involving a pair of semiauto and reference datasets. Thus, three comparisons are made, one for each of Level 1, Level 2, and Level 3.

3.6.1 Validation Set

The validation set is selected following the split of the reference dataset. This split into training and validation sets was selected in a previous project by assigning images to each set according to their source videos. As each video produces images that tend to be similar, images sourced from the same video are assigned to the same set. If this precaution is not made, a video can be used in both sets, con-

taminating the validation set with samples that are too similar to training samples, unfairly boosting the evaluation results. The videos composing the validation set were chosen to maintain a reasonable class distribution balance at the last hierarchical classification level and, also, to supply the most representative images to the training set. Thus, this split is assumed to be reasonable for the semi-automatic labeled dataset as well.

As both datasets are sourced from the same set of videos, the same criteria used to choose the training and validation split into the reference dataset can be applied to the semiauto dataset. The validation set images of the semi-automatic dataset are sourced from the same videos used to compose the reference dataset's validation set. Thus, two validation sets are available, even though they are sourced from the same group of videos: a validation set derived from the reference dataset and a validation set derived from the semi-automatic dataset. As they are subsets of the same pool of available frames from the same set of videos, they differ in what images are selected to compose each set. The semi-automatic dataset's validation set has more images, but they are selected less carefully than those in the reference dataset.

Chapter 4

Evaluation of Datasets and Methods

In this chapter, the results of the iterative annotation and dataset comparison are discussed. The annotation process produces labeled examples that are used to create a new dataset, which is evaluated and compared to an existing baseline, reference dataset. It is expected that a model trained in the new dataset can achieve performance comparable to a model trained in the reference dataset.

4.1 Semi-Automatically Annotated Dataset

In this section, we describe the results of the annotation process, the resulting image datasets, the split into training and validation sets, and resulting class distributions.

4.1.1 Iterative Annotation

Following the procedure described in Section 3.3, the entire video dataset is annotated at three hierarchical levels. This procedure is carried out by two human annotators: one labeled the initial set of 4 675 images and the other, the remaining images.

Each iteration labels a different number of images, but a recurring trend is that the first iteration automatic labeling step annotates the vast majority of the unlabeled dataset, while the remaining images are labeled in the remaining iterations. A likely explanation is that, since the unlabeled dataset is sampled uniformly from videos, the majority of its images are very similar among themselves, and thus, easy to classify. It follows that, in the first labeling iteration, the classifier is able to annotate these easy images, and the remaining unlabeled images are more difficult examples of the set. The results of Level 1 annotation are discussed here, while Levels 2 and 3 are left for Appendix A.

The results of each labeling iteration for hierarchical Level 1 are shown in Table 4.1. Each row represents one iteration of the process, indicated by the first column, while the first row, iteration zero, indicates its the initial, unlabeled, state. The manual, auto, and unlabeled columns represent, respectively, the number of images annotated manually by humans, automatically by a classifier model, and of images still not annotated that will be carried over to the next iteration. The percentage columns indicate the relative percentages of each row: manual and auto percentages indicate the percentage of annotated images over the previous iteration's unlabeled set; the annotated percentage indicates the percentage of images annotated in the row over the initial number of unlabeled images, 457 thousand. The total row shows the sum of manual and annotated images over all iterations for the manual and auto columns; for the percentage columns, it shows the percentage of each column's sum of images over the initial unlabeled dataset. This means that a little over 2% of the annotated images were manually annotated, and almost 98% were automatically annotated.

Table 4.1: Level 1 iterative annotation results. The process took 9 iterations to complete. The *Manual* column does not indicate 100% annotated images because duplicated images were discarded during the annotation.

It	**Manual**	**Auto**	**Unlabeled**	**Manual %**	**Auto %**	**Annotated %**
0	0	0	457372	0.00 %	0.00 %	0.00 %
1	4675	408685	44012	1.02 %	89.36 %	90.38 %
2	440	16844	26917	1.00 %	38.27 %	3.78 %
3	269	7410	19306	1.00 %	27.53 %	1.68 %
4	193	8319	10898	1.00 %	43.09 %	1.86 %
5	108	2618	8204	0.99 %	24.02 %	0.60 %
6	100	1661	4677	1.22 %	20.25 %	0.39 %
7	100	872	5518	2.14 %	18.64 %	0.21 %
8	100	1428	4032	1.81 %	25.88 %	0.33 %
9	4031	0	0	99.98 %	0.00 %	0.88 %
Total	**10016**	**447839**	**-**	**2.19 %**	**97.92 %**	**100,11 %**

Observing the annotated percentage column, over 90% of the unlabeled set is annotated in the first iteration and the remaining images are labeled over the remaining eight. This reinforces the idea that the majority of the unlabeled images are classified easily. Indeed, this seems to be a strength of model-assisted annotation: the model can quickly label the easy images, directing the efforts of the human annotators to the harder examples. The data also shows that the manually annotated images represent only 2.2% of all annotated examples. This shows the need to train the models using only the manual set, as the automatic set would overwhelm the more valuable manually annotated set's contributions during training.

The iterative annotation data of Levels 2 and 3 follows the same trend as Level

1 and is available in the Appendix A, in Tables A.1 and A.2.

The data for Levels 1 and 3 indicate values over 100% annotated images, in relation to the initial unlabeled set. This occurs for the following reasons: the initial manually annotated image set contains duplicate images that were discarded in later stages; part of the starting manually annotated set contains images carried over from previous levels, which may not be useful to the next task; some images may be annotated twice, having manual and automatic annotations. These are oversights or artifacts from the annotation process implementation that do not affect the final results. Annotations that are duplicated or not relevant to a given level are discarded after the iterative annotation process, with manual labels taking precedence over automatic ones.

4.1.2 Automatic Image Classification

This subsection exemplifies the process of automatic image classification carried on by the model during the iterative annotation process, discussed in Section 3.3.1, through output histograms, and shows the results of the first iteration of the Level 1 dataset annotation.

Figure 4.1 shows a histogram of image scores of the validation set after being evaluated by the trained model. These are the positive class values of the network after passing through the *softmax* function. The values are normalized between 0 and 1 and displayed as two overlapping 100-bin histograms, one for positive and the other for negative outputs. The red and blue vertical lines, respectively, represent the upper and lower threshold values. The chosen upper threshold, in this case, is $T_{upper} = 0.89$ and the lower one, $T_{lower} = 0.09$. These values achieve 99.09% ground truth positive examples above the upper threshold (over examples above the upper threshold) and 0.96% ground truth examples under the lower threshold (over all ground truth positive examples), thus satisfying the desired threshold conditions.

The validation set in this iteration is composed of a set of 20% randomly selected examples of the manually annotated dataset obtained up until this point. In this case, there is only one iteration of labeling.

In each iteration, all manually annotated examples up until the current point are used to train the model. These examples are randomly split 80/20% in training and validation sets for model training. This validation set is also used for threshold selection. As per the data on Table 4.1, 4 675 images are available for training in the first iteration. In this case, duplicate images were discarded and the dataset size is reduced to 4 310 images. Thus, the training set is composed of 3 448 (80% of total) images, and the validation set, of 862 images (20% of total). The latter is the number of examples in Fig. 4.1.

These thresholds are applied to the unlabeled set, as shown in Fig. 4.2. Images above the upper threshold calculated on the validation set are classified as positive, while those below the lower threshold are classified as negative. In the case, there are 452 697 examples in the unlabeled set, that also is the number of examples in the histogram. Of this total, 408 655 examples were labeled, with 366 196 positive and 42 459 negative labeled images. 44 042 images were left unlabeled and proceeded to the next iterations. This matches the number of unlabeled images left after the first iteration, in the second row of Table 4.1 (30 images were duplicates and thus discarded).

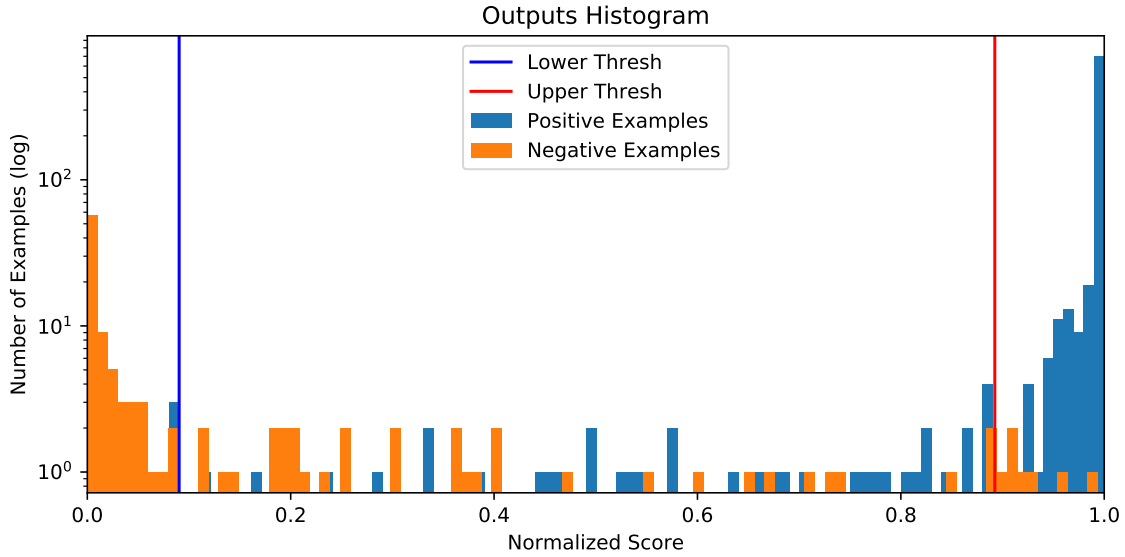


Figure 4.1: Histogram of positive and negative output scores for the first iteration of annotations of the Level 1 dataset, with 100 bins each. The data comprises only the validation set. The vertical axis is in logarithmic scale.

4.1.3 Dataset

With the results of Section 4.1.1, annotated examples for the three hierarchical classification levels are available. These examples are compiled and used to build a new image dataset for the desired tasks. Note that, since the video dataset is proprietary, and the images are sourced from these videos, the dataset is not made publicly available. In this section, it will be discussed as a product of the annotation process adapted for this work.

The annotated images are then transformed into binary datasets, according to the procedure detailed in Section 3.5. This only applies to hierarchical Level 3, a multiclass task, as Levels 1 and 2 are binary tasks annotated in a binary fashion, and thus need no further treatment to compose their datasets.

The class distributions for the annotations of hierarchical Levels 1 and 2 are

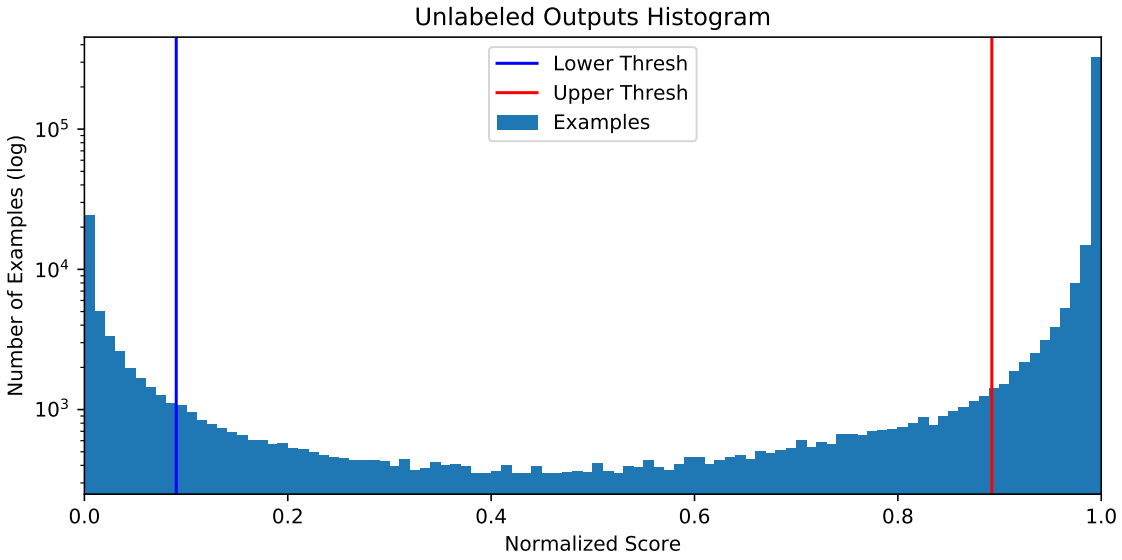


Figure 4.2: Histogram of output scores for the first iteration of annotations of the Level 1 dataset, with 100 bins. The data comprises the unlabeled set. The vertical axis is in logarithmic scale.

described in Table 4.2. The data also corresponds to the class distribution for these levels corresponding datasets. The class distributions for Level 3 annotations are shown in Table 4.3.

Table 4.2: Semi-automatic class distributions of annotations for hierarchical Levels 1 and 2. These are the same as the distributions for the corresponding datasets.

		Automatic		**Manual**		**Total**		
		Count	%	Count	%	**Count**	**%**	**Diff**
Duct	Positive	388642	87 %	7885	79 %	**396243**	**87 %**	**8%**
	Negative	59195	13 %	2117	21 %	**61115**	**13 %**	
	Total	447837	100 %	10002	100 %	**457358**	**100 %**	
Event	Positive	123457	32 %	7119	49 %	**130576**	**33 %**	**17%**
	Negative	258232	68 %	7274	51 %	**265506**	**67 %**	
	Total	381689	100 %	14393	100 %	**396082**	**100 %**	

In both tables, the rows are grouped by class, and each row has the count of *positive*, *negative* and *total* examples for that class. They also show the percentages of each row count in relation to the total amount of examples of the corresponding class. The data is further split by the *automatic* and *manual* columns. The *Total* column is the sum of both manual and automatic examples, as well the final result of the annotation process. For Levels 1 and 2, it also indicates the dataset's class distributions. The *Diff* column indicates the difference between manual and automatic positive percentages. The percentages of the *Automatic* and *Manual* columns are in relation to the sum of examples in each category. The percentages of the *Total* column are in relation to the number of images available to the iterative annotation

Table 4.3: Semi-automatic dataset class distributions for the annotations of hierarchical Level 3 classes.

		Automatic		**Manual**		**Total**		
		Count	%	Count	%	**Count**	**%**	**Diff**
Anode	Positive	5080	4 %	557	7 %	**5637**	**4 %**	**3%**
	Negative	117697	96 %	7164	93 %	**124861**	**96 %**	
	Total	122777	100 %	7721	100 %	**130498**	**100 %**	
Buried	Positive	37339	31 %	5501	63 %	**42840**	**33 %**	**32%**
	Negative	83879	69 %	3234	37 %	**87113**	**67 %**	
	Total	121218	100 %	8735	100 %	**129953**	**100 %**	
Damage	Positive	16674	14 %	4288	41 %	**20962**	**16 %**	**27%**
	Negative	103266	86 %	6281	59 %	**109547**	**84 %**	
	Total	119940	100 %	10569	100 %	**130509**	**100 %**	
Flange	Positive	10336	9 %	3251	33 %	**13587**	**10 %**	**25%**
	Negative	110388	91 %	6492	67 %	**116880**	**90 %**	
	Total	120724	100 %	9743	100 %	**130467**	**100 %**	
Repair	Positive	109203	90 %	1547	17 %	**110750**	**85 %**	**75%**
	Negative	11548	10 %	7789	83 %	**19337**	**15 %**	
	Total	120751	100 %	9336	100 %	**130087**	**100 %**	

for that level. For Table 4.2, the percentages of the *Total* column are calculated in relation to the number of starting images for that level. These numbers are 457 358 images for Level 1, 396 243 for Level 2 and 123 457 for Level 3.

A measure of how effective the annotation method is in reducing human effort is the ratio between automatically and manually annotated images. For the hierarchical Level 1, this ratio is 44.77 automatically annotated images for every manual image, or 44.77:1. For Level 2, the ratio is 26.52:1 and, for Level 3, the average ratio between all five classes is 13.29:1. These results agree with the idea that each hierarchical level is progressively more difficult than the previous, which is reflected in the smaller automatic labeling ratio of higher levels.

The manual class distributions are obtained from uniform sampling of the source images and are annotated by a method with low annotation error: human specialist annotators. Thus, they can be assumed to be a good approximation of the dataset's class distribution. If the difference between the automatic and manual distributions is too large, it may be indicative of a problem in the automatic annotation process for that class. However, the difference in class distributions percentages is not a precise indicator of difference in class distribution, in some cases. A small percentage difference can still be significant if the percentage of positive annotations is also small. Measures that directly evaluate difference in sample distribution would be more informative in identifying divergences between manual and automatic annotations, but they were not further explored in this work.

After the annotation of Level 3, the Level 3 multiclass dataset was composed

according to the procedure and class priority list described in Section 3.5. The result is a multiclass dataset with unbalanced class distributions. The dataset is also smaller than the total number of images annotated at Level 3. This occurs because some images have not been labeled as any of the five relevant classes, but as other types of events not relevant for this task and, thus, are not included in the dataset.

Since the annotations for the Repair class flag 85% of the dataset as positive examples, it is left as the lowest priority class in the binary conversion process (Section 3.5). Even then, it is the second largest class in the Level 3 dataset, as detailed in Table 4.4.

Table 4.4: Class counts and percentages for Level 3 multiclass dataset. Percentages are relative to the total image count.

	Count	**Percentage**
Anode	5637	4,76 %
Buried	42774	36,16 %
Damage	20962	17,72 %
Flange	13282	11,23 %
Repair	35649	30,13 %
Total	118304	100,00 %

4.1.4 Validation Sets

With the semi-automatic dataset ready, its validation set can be separated following the procedure of Section 3.6.1. In the reference dataset, 17 videos are used as a source for the validation set images. The images in the semiauto dataset associated with those videos are selected to build its validation set. The resulting class distribution between training and validation sets for the semi-automatic dataset is shown in Table 4.5. The same information about the reference dataset is shown in Table 4.6.

Each table shows the number and percentage of images for the training set, validation set, and the entire dataset. The percentages are over the total amount of images in each group.

The foremost difference between the two datasets (semiauto and reference) is their size. The semi-automatic dataset is much larger, with 454 thousand more images, or 177 times more images, than the reference dataset at Level 1; 393 thousand more images, or 181 times more, at Level 2; and 115 thousand more images, or 37 times more, at Level 3. Its greater size tends to confer an advantage over training models on the smaller dataset, as explored in Section 4.2.

Both validation sets have class imbalances. The reference dataset was assembled by hand with the general goal of obtaining the same proportion of image contri-

butions from each video, selecting images that harbor unique and different events, selecting most kinds of events from the dataset and maintaining class balance. These goals are achieved partially with the data and annotations available at the time. The reference dataset suffers from a very small number of examples in some classes, more so in the validation set, and some class imbalance, especially at Level 1.

The semiauto dataset is split into training and validation sets following the guideline of the reference dataset. The lack of examples is mitigated, as the dataset is larger, but the class imbalance becomes more pronounced. At Level 1, its training set has a distribution of positive examples of 88%, an imbalance which is reversed for Level 2, but that is also undesirable. The class distribution at Level 3 is also worsened, with Anode and Flange classes corresponding to 6% and 5% of the total number of examples, respectively.

These problems are mitigated by training with the weighted cost function and chosen performance measures, but the harshest cases cannot be fully compensated. These deficiencies may reduce the classifier's performance or make it more difficult to be evaluated.

Table 4.5: Class distributions for train and validation sets of the semi-automatic dataset.

		Train		**Validation**		**Total**	
Semiauto Dataset		Count	%	Count	%	Count	%
Level 1	**Duct**	367492	88 %	28751	74 %	396243	87 %
	Not Duct	51060	12 %	10055	26 %	61115	13 %
	Total	418552	100 %	38806	100 %	457358	100 %
Level 2	**Event**	122374	34 %	8202	21 %	130576	33 %
	Not Event	234641	66 %	30865	79 %	265506	67 %
	Total	357015	100 %	39067	100 %	396082	100 %
Level 3	**Anode**	4370	4 %	1267	6 %	5637	5 %
	Damage	15549	16 %	5413	27 %	20962	18 %
	Buried	34540	35 %	8234	42 %	42774	36 %
	Flange	12353	13 %	929	5 %	13282	11 %
	Repair	31661	32 %	3988	20 %	35649	30 %
	Total	98473	100 %	19831	100 %	118304	100 %

4.2 Dataset Evaluation

This section shows the results of the evaluation process detailed in Section 3.6. The measures used for comparison were the cross-entropy loss and the average accuracy score of each class, both obtained on the validation set. The models were trained and evaluated five times. The resulting statistics are shown in Table 4.7. Columns *Dataset* and *Val* (for validation) indicate, respectively, the dataset used to train

Table 4.6: Class distributions for train and validation sets of the reference dataset.

		Train		Validation		Total	
Reference Dataset		Count	%	Count	%	Count	%
Level 1	**Duct**	1430	71 %	362	65 %	1792	70 %
	Not Duct	584	29 %	198	35 %	782	30 %
	Total	2014	100 %	560	100 %	2574	100 %
Level 2	**Event**	873	47 %	145	47 %	1018	47 %
	Not Event	1003	53 %	164	53 %	1167	53 %
	Total	1876	100 %	309	100 %	2185	100 %
Level 3	**Anode**	366	14 %	92	15 %	458	14 %
	Damage	359	14 %	124	20 %	483	15 %
	Buried	881	35 %	197	31 %	1078	34 %
	Flange	618	24 %	117	19 %	735	23 %
	Repair	324	13 %	99	16 %	423	13 %
	Total	2548	100 %	629	100 %	3177	100 %

the model, and the validation set used to evaluate the trained model. On the *Val* column, the letter R indicates that the reference set was used for validation, and the letter S indicates that the semiauto set was used. The columns *Loss* and *Loss std* indicate the mean and standard deviation of the cross-entropy loss. Columns *avg acc* and *avg std* indicate the mean and standard deviation of the class-average accuracy.

Table 4.7: Evaluation statistics for reference and semi-automatic datasets on both validation sets over five evaluation runs. The best results of each validation set and level are shown in bold.

	Dataset	**Val**	**Loss**	**Loss Std**	**Avg Acc**	**Acc Std**
Level 1	Reference	R	0.269	0.015	89.10 %	1.2 %
	Semiauto	R	**0.199**	**0.009**	**91.35 %**	**0.9 %**
	Reference	S	0.268	0.029	88.34 %	1.5 %
	Semiauto	S	**0.159**	**0.010**	**93.70 %**	**0.6 %**
Level 2	Reference	R	**0.140**	0.032	**94.45 %**	1.1 %
	Semiauto	R	0.291	**0.013**	88.94 %	**0.7 %**
	Reference	S	0.424	0.039	78.88 %	5.3 %
	Semiauto	S	**0.144**	**0.009**	**94.30 %**	**0.6 %**
Level 3	Reference	R	**1.157**	**0.085**	**63.20 %**	**3.4 %**
	Semiauto	R	1.528	0.098	54.88 %	5.3 %
	Reference	S	1.640	0.228	53.09 %	4.1 %
	Semiauto	S	**0.493**	**0.055**	**68.20 %**	**3.6 %**

Comparing the evaluation results, it seems that the model trained in the semiauto dataset has an advantage over the reference-trained model at Level 1, but otherwise performs similarly. Training in the semiauto dataset allows the model to achieve better average accuracy and loss at Level 1, and when evaluating on the semiauto

validation sets at Levels 2 and 3. However, at these levels, the reference model (i.e. model trained on the reference dataset) performs better in loss and accuracy on the reference validation set. The semiauto model also achieves a lower standard deviation than the baseline model for both measures on all validation sets except at Level 3 on the semiauto validation set.

A trend at all levels is that each model performs better on its own validation set. That is to be expected, as the datasets were annotated differently. This is caused by the video sample rate, which is different between the two, but also by slightly different criteria for class annotation, as described in Section 3.2.

Table 4.8: F1 scores for Level 3 task classes of the best models selected from five evaluation runs. Best results of each validation set are shown in bold.

Dataset	**Val**	**Anode**	**Damage**	**Buried**	**Flange**	**Repair**
Reference	R	**0.764**	**0.619**	0.797	0.601	**0.597**
Semiauto	R	0.697	0.413	**0.811**	**0.729**	0.247
Reference	S	0.509	0.826	0.601	0.252	0.217
Semiauto	S	**0.786**	**0.852**	**0.966**	**0.665**	**0.651**

The Level 3 results elicit a more detailed analysis. Table 4.8 contains the F1 scores [1] of the best models in each dataset-validation combination for Level 3 task, while their confusion matrices are shown in Figs. 4.3 and 4.4. Note that the results of the confusion matrices tend to be better from those reported in Table 4.7 because they are obtained only from the best runs, while Table 4.7 reports the average results of five runs. However, they still allow general dataset trends to be observed. Also note that the best results from the confusion matrices and Table 4.8 may differ, as while the table shows the F1 scores, the matrices show class accuracy.

Each figure shows the results of the two models – trained on the reference and semiauto datasets – and evaluated in the same validation set. Observing the results on the reference validation set in Fig. 4.4 and on Table 4.8, the reference-trained model performs better on all classes, except Buried and Flange. On the Buried class, it performs similarly to the competing model, but on the Flange class, the semiauto model achieves a significantly higher F1 score than the reference. It seems that the reference dataset does not have images in sufficient numbers or representative quality to allow the model to learn to identify this class as well. The semi-automatic dataset appears to have an advantage simply because its larger size contributes more examples to the training set, which may allow the model to better learn to distinguish these classes.

The hardest class for both models is Repair. The reference-trained model classifies with an F1 score of 0.60, a passable performance, but the semiauto model

[1] The F1 Score is a measure of accuracy for binary classification that corresponds to the harmonic mean of precision and recall and is defined as $F_1 = 2\frac{P \times R}{P+R}$, where R is recall and P is precision.

fails, with an F1 score of 0.25. This situation is mirrored on the semi-automatic validation set, with the reference model performing very poorly, and the semiauto model, passably well. This points to a difference in labeling criteria: the Repair class may have been annotated differently in each dataset. Thus, a model trained with the examples of one dataset cannot learn to classify the examples on another.

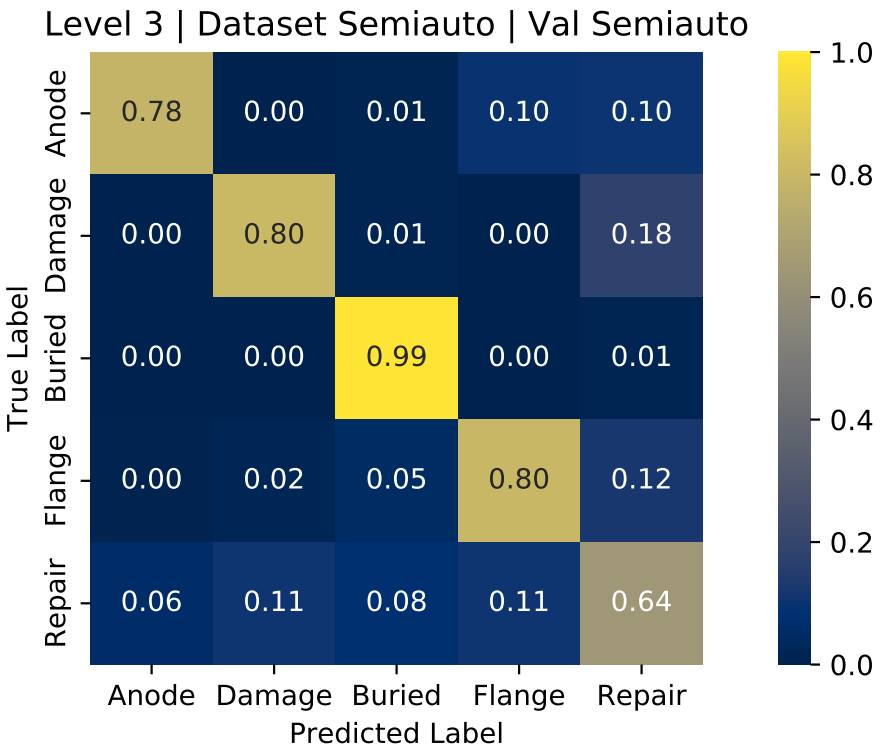
The Damage class appears to be difficult for both models. Both misclassify a large fraction of examples as belonging to the Repair class. This makes sense, as both classes present similar features on the pipe surface in some cases. Some types of pipe damage are accompanied by attempted repairs. However, the classes are still discernible on the majority of circumstances where this overlap does not occur. This suggests that the reference validation set contains examples that are poorly selected or too different from those on the training sets. The fact that both models perform well on the semiauto validation set also suggests this.

Observing the semiauto validation set results on Fig. 4.3 and Table 4.8 shows that the semiauto model wins on all classes, in this set. The combination of a larger number of examples than the reference set and evaluation on its own validation set may have conferred an advantage to the semiauto-trained model.

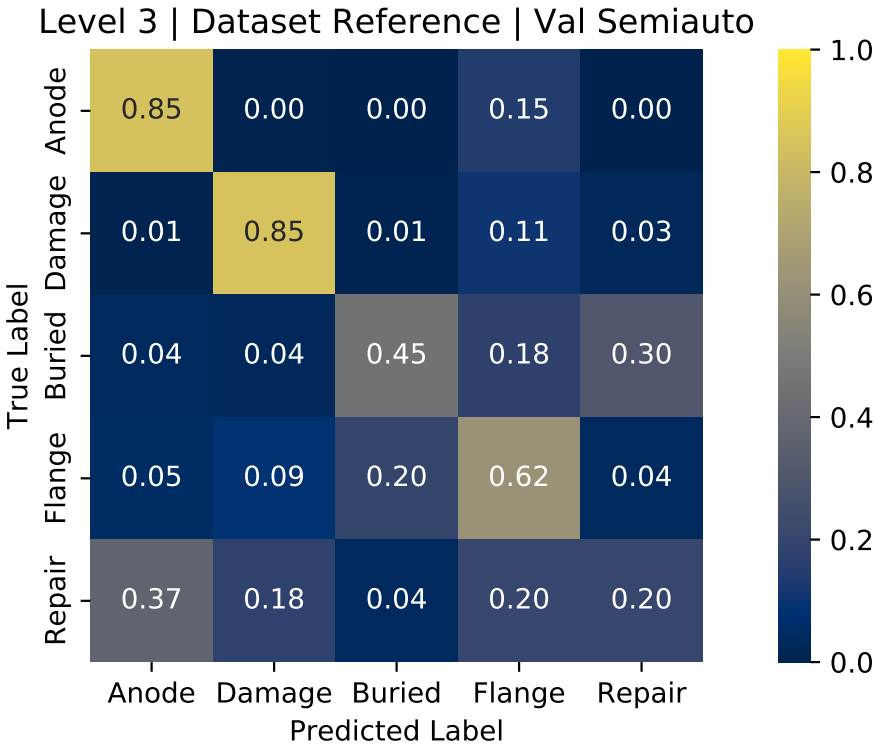
On the Flange class, the reference model performs very poorly when evaluated on the semiauto validation set. Since other combinations between dataset and validation all achieve a reasonable F1 score of 0.6 and higher, this may indicate a poor selection of training examples on the reference dataset.

Overall, on the semi-automatic dataset, the Anode, Buried, and Flange classes appear to be well-annotated. The Repair class shows strong indications that the network could not adequately label this class. While the task may be more difficult by its overlap with the Damage class, the Anode and Flange classes have a stronger overlap, but both models perform reasonably well on them. The reason for the poor performance on the Repair class may be simply because it is harder to classify than the other classes, and the network needed more manually annotated examples to be able to label the dataset well.

Accounting for all levels, the results meet the desired performance standards for the semi-automatically annotated dataset. The model trained on it achieves reasonable performance on all levels, surpassing the reference dataset on Level 1 and attaining equivalent performance on Levels 2 and 3, with each model performing on its own validation set, as expected. Since the goal was for the semi-automatic dataset to reach, at least, equivalent performance as the baseline dataset, these results satisfy that goal.

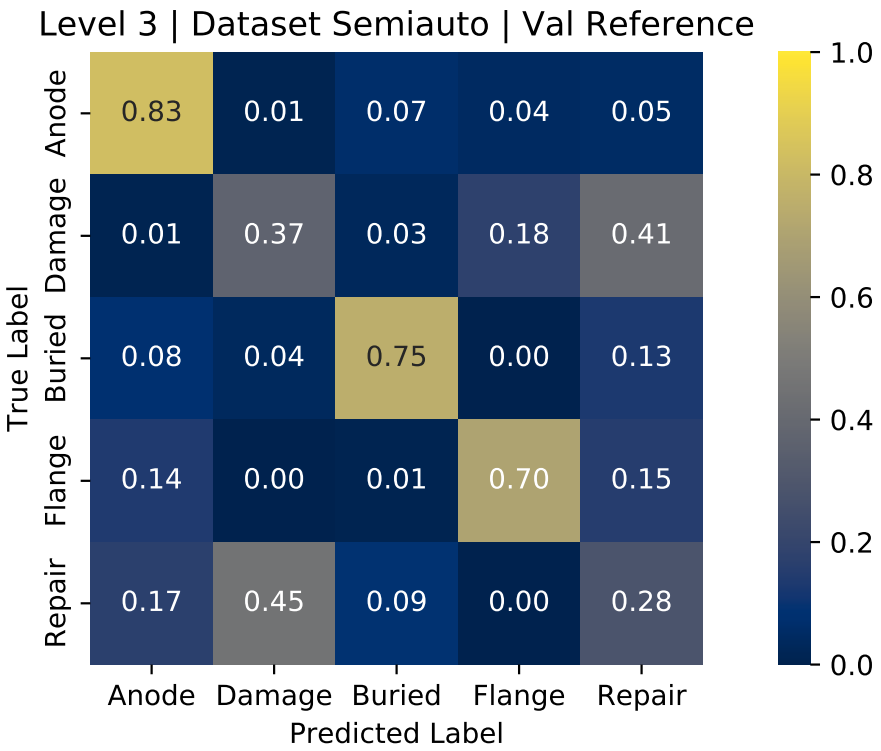


(a) Results of the semiauto dataset, evaluated on the semiauto validation set. Mean and standard deviation of class accuracy: $80.2 \pm 10.9\%$

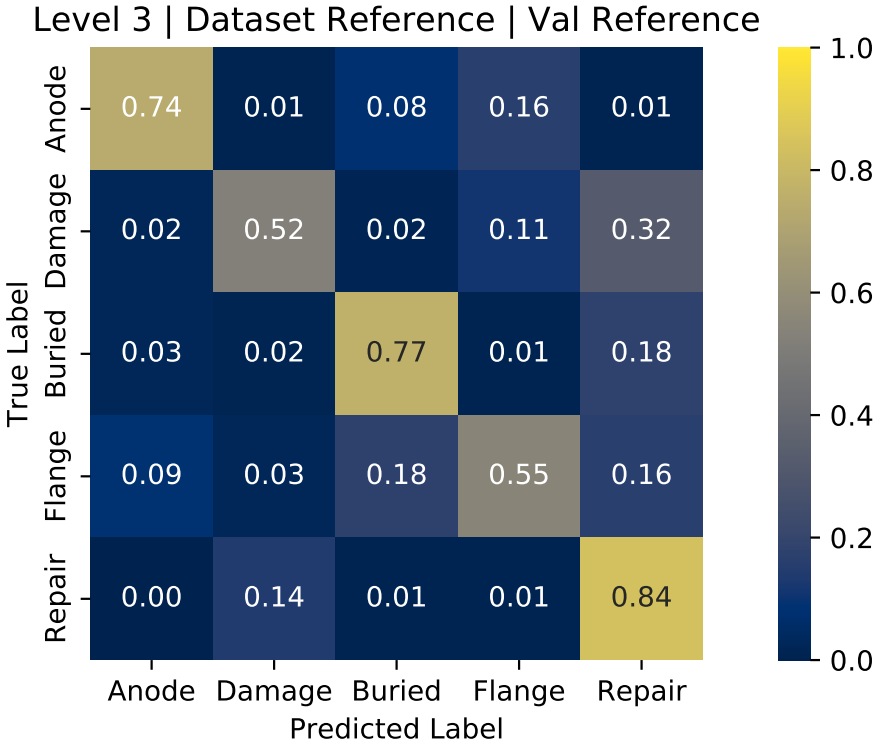


(b) Results of the reference dataset, evaluated on the semiauto validation set. Mean and standard deviation of class accuracy: $59.3 \pm 24.6\%$

Figure 4.3: Confusion matrices for the Level 3 task, evaluated on the semiauto validation set, row normalized. The values shown are the results of the best model selected over five runs.



(a) Results of the semiauto dataset, evaluated on the reference validation set. Mean and standard deviation of class accuracy: $58.6 \pm 21.7\%$



(b) Results of the reference dataset, evaluated on the reference validation set. Mean and standard deviation of class accuracy: $68.3 \pm 12.5\%$

Figure 4.4: Confusion matrices for the Level 3 task, evaluated on the reference validation set, row normalized. The values shown are the results of the best model selected over five runs.

4.3 Annotation Measures

This section details efforts to measure the quality of the automatic labeling carried out in this project. These measures are subjective, as they rely on human annotation, but they are still informative when coupled with those on Section 4.2.

4.3.1 Labeling Error

To evaluate the quality of the dataset created by the iterative process, its labeling error is estimated. A human annotator labels samples of the dataset and these labels are compared to the labels given by the model. If the labels agree, the example is counted as correctly labeled, else it is counted as an error.

Two specialists participated in this error check annotation. They did not participate in the annotation of the semi-automatic dataset. This avoids perpetuating annotator bias, but disagreement on labeling criteria may increase error counts. Two annotators is a small sample, and it would be better to perform this procedure with more independent human annotators, none of which would have taken part in the construction on the evaluated dataset. Only two annotators participated because of time and manpower constraints.

The same annotation interface and procedure used in the iterative annotation step is used in this stage. The human annotator is asked to label each sample without regard to what dataset it was sampled from. The images are presented without label identification as to not inform to the annotator which label was assigned by the model.

A random sample of 1% of each level is obtained for labeling. Levels 1 and 2 datasets produced one sample each, while the Level 3 dataset produced five samples, one for each class. Each is evaluated separately, as this measure is about the annotation process and not the dataset. Thus, each Level 3 class is annotated and evaluated independently. These class labels are interpreted as binary for the purpose of calculating the labeling error. The images evaluated in this step are only those automatically annotated by the annotator model, and not those manually labeled as part of the iterative annotation process. This is done to evaluate only the quality of the annotation done by the model, as the manually labeled images are assumed to be well-annotated. Also, since the majority of the semi-automatic dataset examples are automatically annotated, it is more important to analyze this group. The analysis of the manual images would also be relevant as a baseline comparison, to assert the extent of the criteria difference impacts the perceived labeling error. This is not explored in this work because of time constraints.

For each dataset, an example is counted as correctly labeled if both the dataset's and the annotator's labels match the target label – they agree on a positive label –

or if both do not match – they agree on a negative label. If neither case occurs, the example is counted as incorrectly labeled. The target label is a positive example for Levels 1 and 2 and, for Level 3, each of its five classes.

Table 4.9: Labeling error of semi-automatic dataset samples for each level and F1 scores of the semiauto model on reference validation set. F1 values come from the best models over five evaluation runs. The data shows a correlation of -0.80 between labeling error and F1 score.

Level	Class	Labeling error	F1 Score
1	Duct	7.6 %	0.94
2	Event	12.1 %	0.88
3	Anode	3.8 %	0.70
	Buried	21.0 %	0.41
	Damage	13.9 %	0.81
	Flange	6.9 %	0.73
	Repair	94.5 %	0.25

This procedure is carried out for each annotator to obtain two sets of labeling errors. The reported results are the mean between both annotators results. Table 4.9 shows the labeling error for the semi-automatic dataset and correlated measures. Each row displays, for each class, the labeling error percentage and F1 score of the semiauto dataset evaluated on the reference validation set. The score is obtained from the best model out of five runs. This combination of training and validation is chosen because, first, the dataset samples that are evaluated at this step are taken from the semiauto dataset. Second, the human annotator took part in the labeling of the reference dataset, so using the corresponding validation set to evaluate allows the same criteria to be used to measure labeling error and model performance. Indeed, calculating the Pearson's correlation between those two measures yields a result of -0.80, strong correlation.

These results indicate that, when a class is poorly labeled by the model, the latter's performance in the reference dataset is also poor. The most evident example is the Repair class. Its labeling error is 94.5%, a value that shows that the model fails to correctly label this class, and, as such, its F1 score on the reference validation set is of 0.25, a low value. Similarly, well-annotated classes such as Duct (Level 1 positive class), Event (Level 2 positive class), and Anode also performs well on the reference dataset.

4.3.2 Annotation Speed

This subsection discusses the measure of dataset annotation times, including human labeling and model training. The numeric figures presented are estimates and averages, as only some annotators rigorously timed their work. Presented here are

annotation times for the previous dataset, the manual annotation required in the iterative annotation procedure, and the automatic annotation performed by deep neural networks. The first case concerns the labeling work made in a previous project to build the reference dataset, while the latter two concern the annotations made in this project to build the semi-automatically annotated dataset.

The measure used for annotation speed is frames per minute of annotation (frames/min). Using this measure instead of annotation time removes the need to report exact time estimates for the labeling work carried out, as the data is not informative without context, is dependent on video frame rate, and is often incomplete.

The annotation times of the labeling work done to build the reference dataset are scarce. Consistent records were not maintained at that time, so the available figures are based on limited data. Only two human annotators out of the five that worked on the project recorded their annotation times. Even then, they vary greatly between videos, stage of the project, and whether the annotation interface is used. The latter is the main factor in variations in annotation speed. The average reported speed without the interface – that is, using one of the methods described in Section 2.1 without tool assistance – is 47 frames/min. The average reported speed using the interface is 4 700 frames/min, a hundred-fold increase. The average speed of all annotations is 2 370 frames/min. Note that, when using the annotation interface to label the reference dataset, the annotator visualizes all frames from a given video and chooses only a fraction of those to actually label and include in the dataset. Thus, the high annotation speeds reported are masked, as they consider frames that are skipped and not included in the dataset as annotated.

The annotation work done for the semiauto dataset is better recorded. Four human specialists recorded their times. Two were involved in the creation of the dataset and two, in its evaluation after creation. The reported annotation speeds ranged from 14.7 to 120 frames/min, with the average being 42.7 frames/min over all annotators. This value is lower than the interface-assisted annotation of the reference dataset because, in that case, videos are labeled individually, by sampling its frames sequentially and labeling those images. This sequential presentation allows human annotators to easily label groups of images belonging to the same class. In contrast, the manual annotation work done to build the semi-automatic dataset requires the annotators to evaluate a set of randomly sampled images from different videos, which may vary in respect with class, scene, contrast, resolution, portrayed objects, among other visual features. Thus, each image requires an individual evaluation from the human annotator and cannot be analyzed as part of a group of images. This slows down the labeling work significantly, as reflected by the reported figures.

The automatic annotation is done entirely by the classifier models. The process

being measured in this case is a part of the iterative annotation process described in Section 3.3. The steps 2, 3, 4, and 5 of Fig. 3.3 are measured, that is, from model training to the selection of labeled and unlabeled examples. Of these steps, only the model training and evaluation is be considered, as the other parts of the process consume a negligible amount of time in comparison. In this case, not only the speed with which examples are processed by the model but also the time spent per epoch is reported, as the latter measure is informative about the network training process.

Using the hardware described in Section 3.3.2, for the Level 1 semiauto dataset, the model spent 19.6 minutes on training and 2.0 minutes on validation. This amounts to an annotation speed of a bit over 21 170 frames/min. This shows exactly how much faster automatic annotation is, in comparison to manual annotation. This process occurs at the same speed for the other hierarchical levels. From Tables 4.2 and 4.3, the fraction of manual and automatically annotated images in relation to total semi-automatic dataset size, averaged along all levels, is 4% manual and 96% automatic images. Thus, the average annotation speed of the process, considering both manual and automatic annotation steps, is 20 270 frames/min.

However, manually labeled examples tend to be more valuable than automatically annotated ones. The former are more likely to be correctly annotated, and the annotator can select only the most representative examples to include in the dataset.

Chapter 5

Conclusion

This work succeeded in creating a new dataset from the proprietary source videos available. It demonstrated the use of an adapted version of the iterative annotation used in [1], using humans and neural networks to annotate a group of images to solve a set of hierarchical classification tasks. Finally, it favorably compared the new dataset with the existing reference dataset, both created from the same source but using different methods for image annotation and selection.

It was shown that an image dataset for classification can be created from a set of videos using the iterative annotation method. This iterative, model-assisted, dataset construction process is faster and requires less human effort than if carried out using only human annotation, and produced a dataset that is 131 times larger than the reference, on average. The new, semi-automatically annotated, dataset was evaluated on classification tasks alongside the existing reference dataset, which had been constructed intermittently over two years using only human labeling. The semi-automatic dataset achieved a lower validation loss than the reference dataset in all tasks when evaluated on its own validation set, as can be seen in Section 4.2. When evaluated on the reference dataset's validation set, the results were mixed, but acceptable: the semi-automatic dataset led to a lower validation loss on one task, and higher on the other two. Furthermore, in the multiclass task, the new dataset allowed for a higher F1 score on two out of five classes in comparison to the reference dataset, when evaluated on the latter's validation set. On its own validation set, the semi-automatic-dataset-based neural network scored higher on all classes. When comparing both validation set's results on each level and considering Level 3 classes results individually, the model trained on the semi-automatic dataset achieved better performance in 10 out of 14 tests.

These results show that the semi-automatically annotated dataset is capable of leading to performance that is superior than, or comparable to, the one obtained from the manually annotated dataset.

The labeling error verification of Section 4.3.1 suggests that some classes could

not be correctly annotated by the neural network. The labeling error of the Repair class was 94% when measured by an independent human annotator. Each dataset-trained model also performed poorly on this class when evaluated on its cross-validation set, that is, the validation set of the competing dataset. These factors point to a discrepancy in the annotation of the Repair class. The semi-automatic labelling models were not able to correctly label images of this class. The class may be more difficult than the others, and thus the model may need more manually annotated examples on each labeling iteration to be able to learn to classify it correctly. A difference in annotation criteria between the reference and the semi-automatic datasets would also explain poor cross-set performance and high perceived annotation error.

The set goal of reducing human effort was also achieved. Comparing the effort amplification ratio, the proportion of automatically annotated to manually annotated images on each level, yields favorable results. In the process of annotating the first hierarchical level, a ratio of 45:1 automatic to manual images was achieved, while the second, a ratio of 26:1. The average ratio for the five classes of the third level is 13:1. The original method described in [1] reported a ratio of 40:1 automatic to manually annotated images, so the ratios obtained are within reasonable range. It is expected that successive levels present more difficult tasks than the previous, which is reflected in the diminishing ratios of levels 2 and 3 when compared to the first level. The results are also masked by the fact that each hierarchical level utilizes the previous level's annotated examples, so that levels 2 and 3 start the iteration process with a number of annotated examples. While it is beneficial to avoid annotating images a second time, they are counted twice on this evaluation. However, it is also not entirely honest to discard these examples when calculating the effort amplification ratio, as each level uses the previous dataset as a starting point in the annotation process, and more images would need to be labeled otherwise.

Another strength of this method is its high annotation speed. Considering both the manual and automatic parts of the annotation process, it can label around 20 270 frames/min, while human annotators were estimated to reach 120 frames/min. This allows the combined human and computer efforts to label a great number of images quickly, reducing the time, human effort, and cost to build a dataset. This speed comes at a cost in the lower quality of annotations, in comparison to manual labeling, but this downside seems to be compensated by the greater number of available examples. The model performance on the new dataset shows that this trade-off may lead to better datasets.

Further work includes exploring modifications and improvements of the iterative annotation method. The set of examples used to train the model in each iteration could be modified by discarding the examples from the manually annotated set that

falls within the thresholds. Since the unlabeled set to be annotated in the following iteration is composed by images whose scores fall between the thresholds, using only training examples that also fulfill these criteria may be beneficial. It would allow the model to train using examples that follow more closely the distribution of the unlabeled set and may improve automatic annotation quality. However, discarding manually annotated examples may deprive the model of valuable information without providing enough benefit. Thus, it remains an approach to be explored.

More binary to multi-label conversion methods could also be explored. The project currently handles the transformation of five binary sets of annotation into a single multiclass dataset by ranking the classes according to project specifications (that may be subjective) and keeping the highest-ranking class label whenever an example has more than one. Changing this priority list also changes the final class distribution of the multiclass dataset, and the priority list thus becomes a parameter of the project. This ranking could also be selected using an objective measure, such as the proportion of positive examples a class labels. If a class that labels the majority of the dataset is ranked high enough on the priority list, then the final dataset will likely be very unbalanced in its favor. Ranking the classes in inverse proportion to the percentage of positive examples would contribute to class balance in the final dataset.

It would be informative to verify labeling error percentages for reference dataset and for the manually annotated portion of the semi-automatic dataset. Currently, only the automatic images of the latter are labeled by independent annotators to check the number or labeling errors. Without having the same done for the manually annotated images, the automatic classification error counts must be evaluated without a baseline for comparison. Having such a baseline would help understand whether the classes for which the classifier has poor performance, such as Repair, also have high labeling error percentages in the manual set, or whether this is a phenomenon brought by the automatic annotation process.

It would also be valuable to verify labeling error percentages on a known, labeled, dataset. Applying the iterative annotation method to a dataset that is already annotated, with a known labeling error, would allow the method to be better evaluated. Using data with known labels, the semi-automatic annotation is carried out, ignoring the labels and treating the data as if it were unlabeled. After the annotation, the new labels are compared with the existing ones, thus allowing the annotation error to be measured at each step of the process. This would help understand which stages of the annotation method need most improvement.

Other neural network architectures could be used to annotate images. The larger size of the semi-automatic dataset could benefit from models with a higher number of trainable parameters than the currently used ResNet-18 network. Deeper ver-

sions of the ResNet network are good candidates, but other architectures such as InceptionNet v4 [16] or ResNeXt [17] could also be tested.

An evaluation of the automatic images contribution could be made. The automatically annotated images are the majority of the dataset, but they may contribute comparatively less than the manually annotated images. To verify this difference, those two sets of images could be used to train a model and evaluate a model in the same way that this project did for the semi-automatic dataset. If the model trained using only the manual images performs as well as if trained on the semi-automatic dataset, that could mean that the automatic images are not contributing to its performance, only increasing dataset size and training time. Evaluating this would help better understand the question.

Bibliography

[1] YU, F., SEFF, A., ZHANG, Y., et al. "LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop", jun 2015. Disponível em: <http://arxiv.org/abs/1506.03365>.

[2] HAYKIN, S. *Neural Networks and Learning Machines*. N. 2. 3rd ed. Hamilton, Ontario, Canada, PEARSON, 2008. ISBN: 978-0-13-147139-9.

[3] GOODFELLOW, I., BENGIO, Y., COURVILLE, A. "Deep Learning: Machine Learning Book", p. 785, 2016. Disponível em: <http://www.deeplearningbook.org/>.

[4] RUSSELL, B. C., TORRALBA, A., MURPHY, K. P., et al. "LabelMe: A Database and Web-Based Tool for Image Annotation", *International Journal of Computer Vision*, v. 77, n. 1-3, pp. 157–173, may 2008. ISSN: 0920-5691. doi: 10.1007/s11263-007-0090-8. Disponível em: <http://link.springer.com/10.1007/s11263-007-0090-8>.

[5] SUN, C., RAMPALLI, N., YANG, F., et al. "Chimera: Large-scale classification using machine learning, rules, and crowdsourcing", *Proceedings of the VLDB Endowment*, v. 7, n. 13, pp. 1529–1540, 2014. ISSN: 21508097. doi: 10.14778/2733004.2733024.

[6] BIANCO, S., CIOCCA, G., NAPOLETANO, P., et al. "An interactive tool for manual, semi-automatic and automatic video annotation", *Computer Vision and Image Understanding*, v. 131, pp. 88–99, 2015. ISSN: 1090235X. doi: 10.1016/j.cviu.2014.06.015. Disponível em: <http://dx.doi.org/10.1016/j.cviu.2014.06.015>.

[7] YU, F., XIAN, W., CHEN, Y., et al. "BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling", pp. 1–16, 2018. Disponível em: <http://arxiv.org/abs/1805.04687>.

[8] GONÇALVES, H. D. A. "Visão Computacional Aplicada à Inspeção de Dutos Submarinos". 2019. Disponível em: <http://monografias.poli.ufrj.br/monografias/monopoli10028270.pdf>.

[9] SAMPAIO, O. A. "Construção de uma Base de Dados para Reconhecimento de Eventos em Vídeos". 2018. Disponível em: <http://monografias.poli.ufrj.br/monografias/monopoli10024539.pdf>.

[10] XIAO, J., HAYS, J., EHINGER, K. A., et al. "SUN database: Large-scale scene recognition from abbey to zoo", *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3485–3492, 2010. ISSN: 10636919. doi: 10.1109/CVPR.2010.5539970.

[11] HE, K., ZHANG, X., REN, S., et al. "Deep residual learning for image recognition", *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, v. 2016-Decem, pp. 770–778, 2016. ISSN: 10636919. doi: 10.1109/CVPR.2016.90.

[12] LI FEI-FEI, WEI DONG, JIA DENG, et al. "ImageNet: A large-scale hierarchical image database", *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. ISSN: 1063-6919. doi: 10.1109/cvprw.2009.5206848.

[13] RUSSAKOVSKY, O., DENG, J., SU, H., et al. "ImageNet Large Scale Visual Recognition Challenge", *International Journal of Computer Vision*, v. 115, n. 3, pp. 211–252, 2015. ISSN: 15731405. doi: 10.1007/s11263-015-0816-y. Disponível em: <http://dx.doi.org/10.1007/s11263-015-0816-y>.

[14] KINGMA, D. P., BA, J. "Adam: A Method for Stochastic Optimization", *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pp. 1–15, dec 2014. Disponível em: <http://arxiv.org/abs/1412.6980>.

[15] PASZKE, A., GROSS, S., MASSA, F., et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: Wallach, H., Larochelle, H., Beygelzimer, A., et al. (Eds.), *Advances in Neural Information Processing Systems 32*, Curran Associates, Inc., pp. 8024–8035, 2019. Disponível em: <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>.

[16] SZEGEDY, C., IOFFE, S., VANHOUCKE, V., et al. "Inception-v4, inception-ResNet and the impact of residual connections on learning", *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, pp. 4278–4284, 2017.

[17] XIE, S., GIRSHICK, R., DOLLÁR, P., et al. “Aggregated residual transformations for deep neural networks”, *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, v. 2017-Janua, pp. 5987–5995, 2017. doi: 10.1109/CVPR.2017.634.

Appendix A

Additional Results

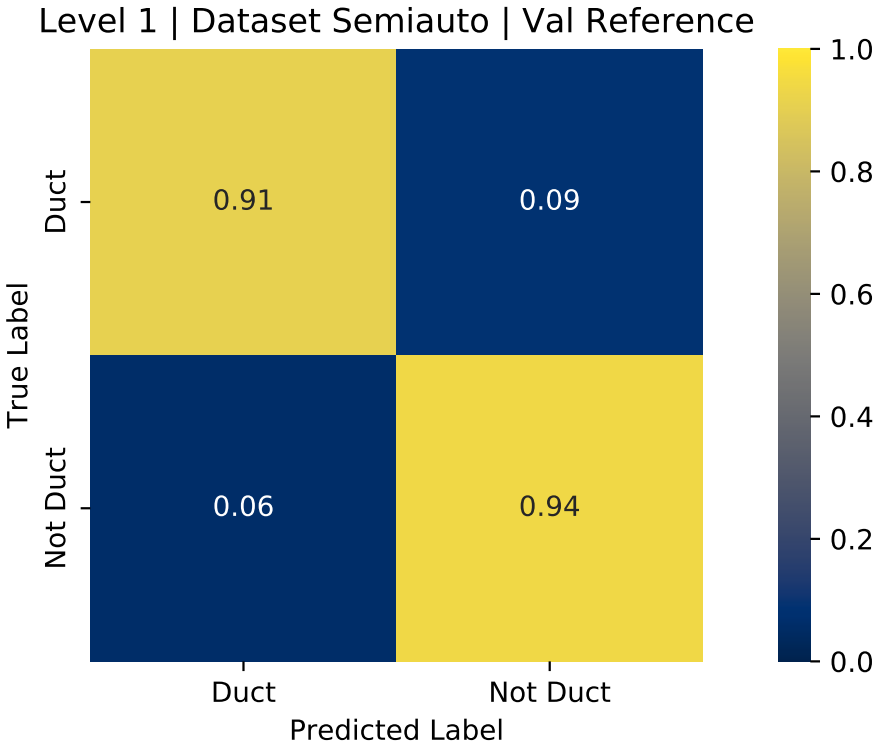
This appendix contains results that were too lengthy or not essential to the discussion on Chap. 4, but should still be made available.

A.1 Iterative annotation details

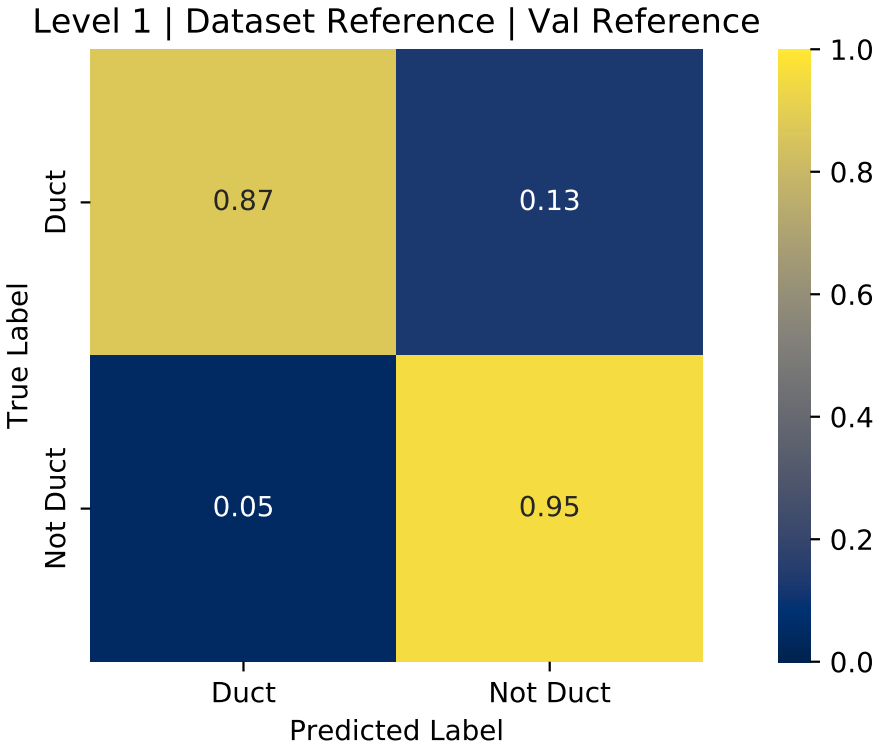
Tables A.1 and A.2 show the results of the iterative annotation process for levels 2 and 3. The *it* column indicates the iteration number for that row; *manual* and *auto* columns indicate the number of manually and automatically annotated images, respectively; the *unlabeled* column shows the number of images still not annotated at the end of the iteration; *manual* and *auto* percentage columns show the percentage of manually and automatically annotated images over the number of unlabeled images of the previous iteration; *annotated* percentage indicates the percentage of annotated images over the initial amount of unlabeled images. In Table A.2, the first column indicates the class of the row group. The presence of values over 100% is discussed in Section 4.1.1.

A.2 Confusion Matrices for Levels 1 and 2

Figures A.1 and A.2 show the confusion matrices for the level 1 task on both validation sets. Figures A.3 and A.4 show the same for the level 2 task. These results are obtained from the best models out of five evaluation runs. The values shown are the percentage of images predicted correctly for each class. The diagrams are row-normalized.

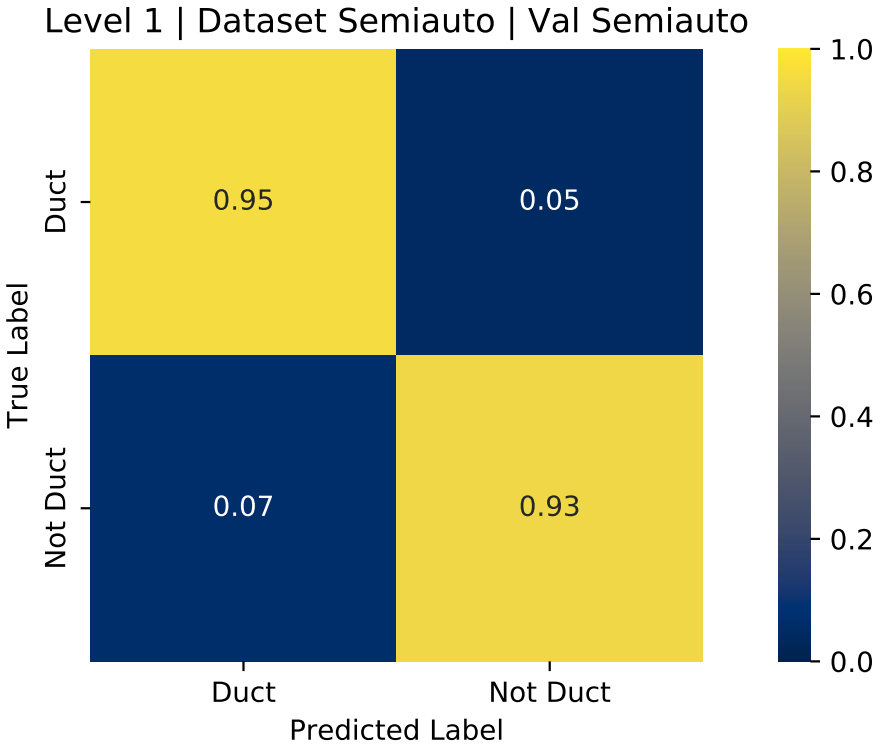


(a) Results of the semiauto dataset, evaluated on the reference validation set.

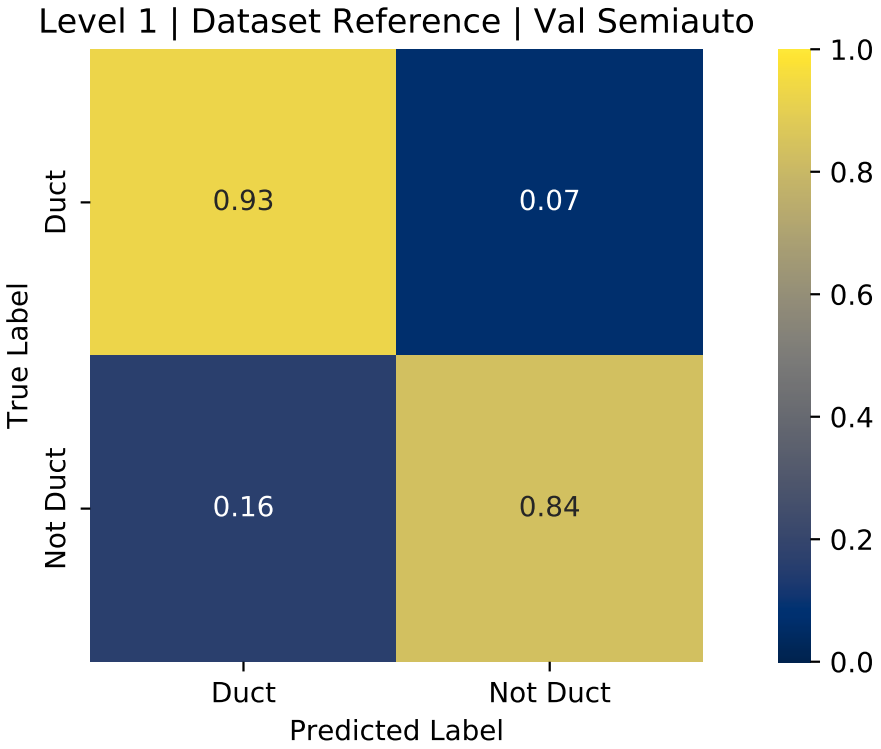


(b) Results of the reference dataset, evaluated on the reference validation set.

Figure A.1: Confusion matrices for the level 1 task, evaluated on the reference validation set, row normalized. The values shown are the results of the best model selected over five runs.

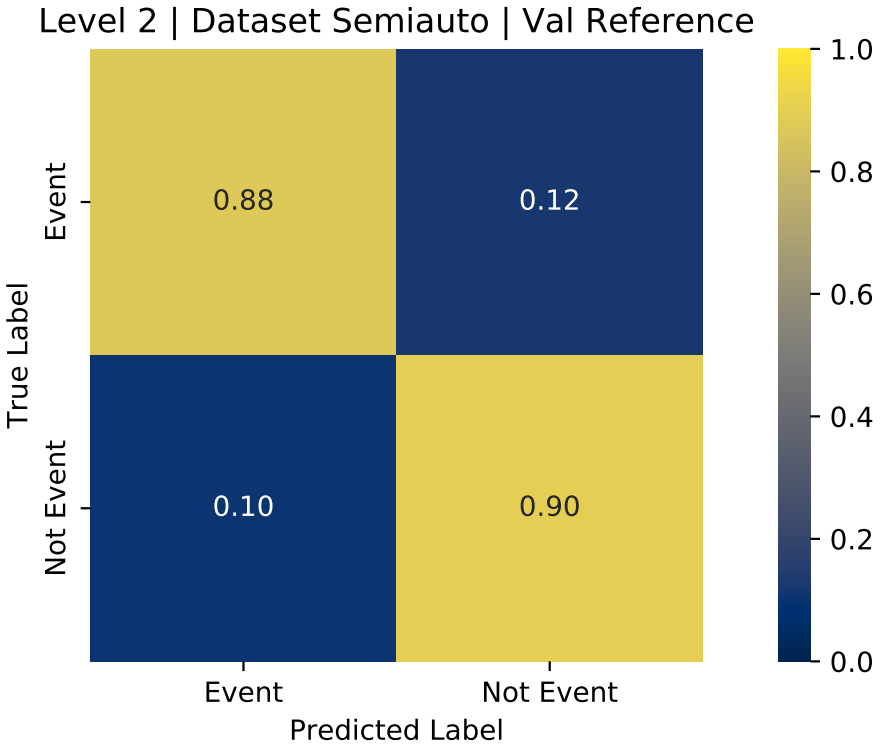


(a) Results of the semiauto dataset, evaluated on the semauto validation set.

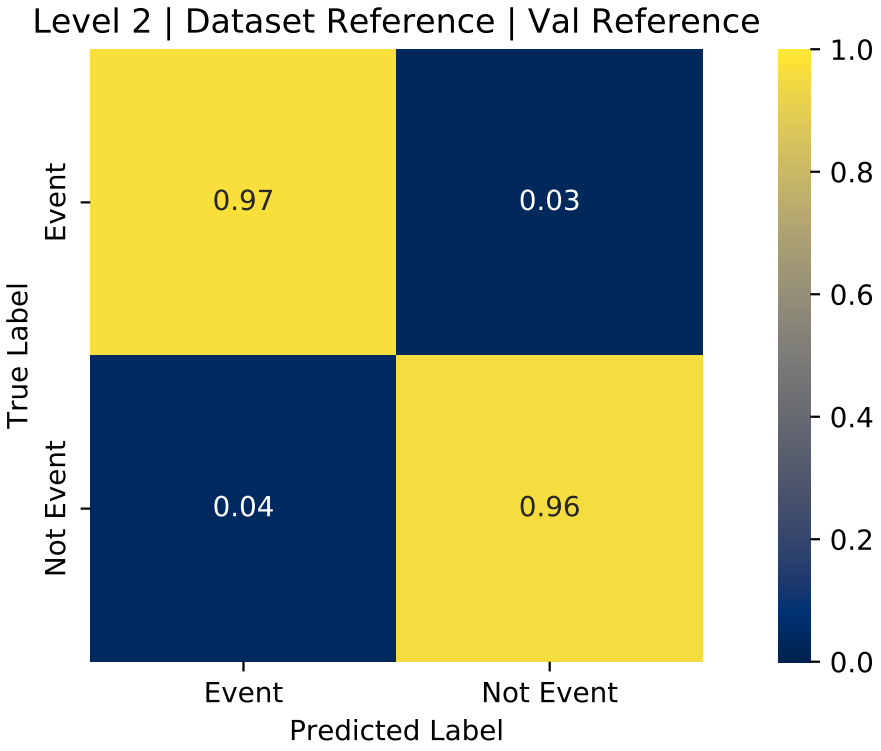


(b) Results of the reference dataset, evaluated on the semiauto validation set.

Figure A.2: Confusion matrices for the level 1 task, evaluated on the semiauto validation set, row normalized. The values shown are the results of the best model selected over five runs.

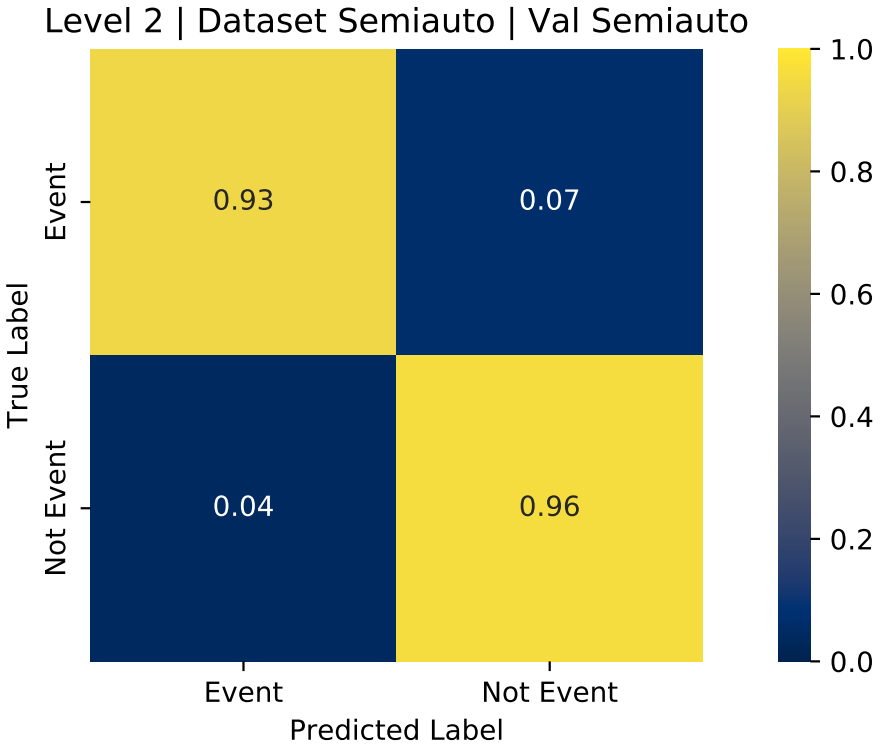


(a) Results of the semiauto dataset, evaluated on the reference validation set.

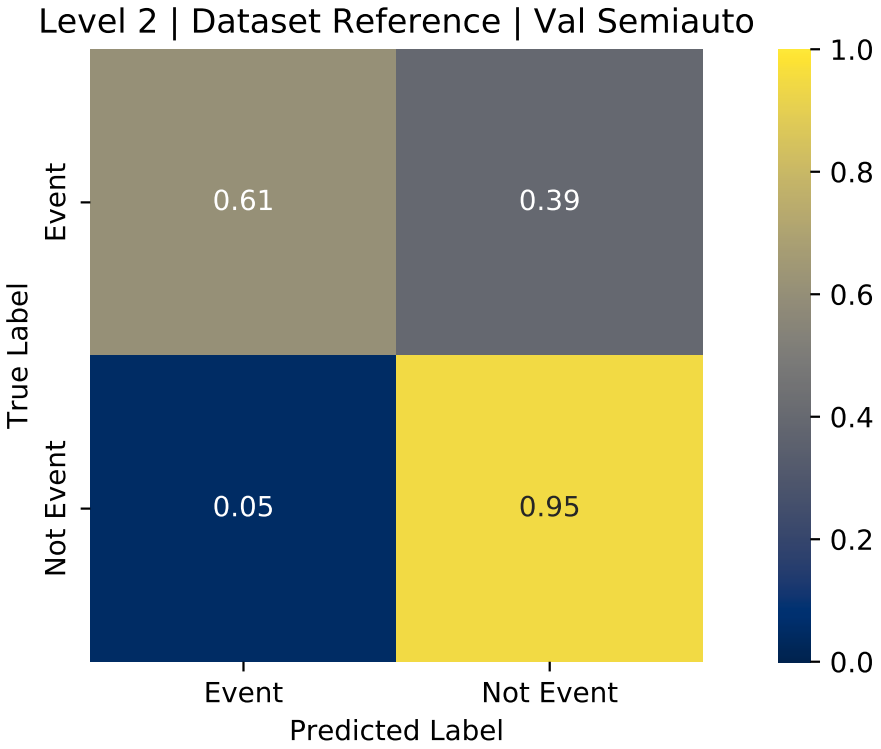


(b) Results of the reference dataset, evaluated on the reference validation set.

Figure A.3: Confusion matrices for the level 2 task, evaluated on the reference validation set, row normalized. The values shown are the results of the best model selected over five runs.



(a) Results of the semiauto dataset, evaluated on the semauto validation set.



(b) Results of the reference dataset, evaluated on the semiauto validation set.

Figure A.4: Confusion matrices for the level 2 task, evaluated on the semiauto validation set, row normalized. The values shown are the results of the best model selected over five runs.

Table A.1: Level 2 iterative annotation results. The process took 16 iterations to complete.

It	**Manual**	**Auto**	**Unlabeled**	**Manual %**	**Auto %**	**Annotated %**
0	0	0	396.243	0.00%	0.00%	0.00%
1	7.755	342.259	46.204	1.96%	86.38%	88.33%
2	462	5.973	39.770	0.12%	1.51%	1.62%
3	397	5.410	33.963	0.10%	1.37%	1.47%
4	338	9.459	24.170	0.09%	2.39%	2.47%
5	241	4.586	19.393	0.06%	1.16%	1.22%
6	193	3.890	15.260	0.05%	0.98%	1.03%
7	151	2.037	13.072	0.04%	0.51%	0.55%
8	130	1.586	11.356	0.03%	0.40%	0.43%
9	112	1.761	9.483	0.03%	0.44%	0.47%
10	100	1.342	8.041	0.03%	0.34%	0.36%
11	100	1.347	6.594	0.03%	0.34%	0.37%
12	100	406	6.088	0.03%	0.10%	0.13%
13	100	479	5.510	0.03%	0.12%	0.15%
14	100	381	5.029	0.03%	0.10%	0.12%
15	100	773	4.157	0.03%	0.20%	0.22%
16	4.157	0	0	1.05%	0.00%	1.05%
Total	**14.536**	**381.689**	**-**	**3.67%**	**96.33%**	**100.00%**

Table A.2: Level 3 iterative annotation results. Results for each of the 5 classes are displayed in row groups. Rows in bold indicate totals for that group.

	It	Manual	Auto	Unlabeled	Manual %	Auto %	Annotated %
Anode	0	0	0	123.457	0.0%	0.0%	0.0%
	1	7.119	122777	680	5.8%	99.4%	105.2%
	2	680	0	0	0.6%	0.0%	1.1%
		7.799	**123457**	**-**	**6.3%**	**99.4%**	**105.8%**
Buried	0	0	0	123.457	0.0%	0.0%	0.0%
	1	7.119	118699	4.758	5.8%	96.1%	101.9%
	2	100	2519	4.658	0.1%	2.0%	2.1%
	3	2.139	0	2.519	1.7%	0.0%	1.7%
		9.358	**121218**	**-**	**7.6%**	**98.2%**	**105.8%**
Damage	0	0	0	123.457	0.0%	0.0%	0.0%
	1	7.119	112501	10.957	5.8%	91.1%	96.9%
	2	108	998	9.860	0.1%	0.8%	0.9%
	3	100	5227	4.533	0.1%	4.2%	4.3%
	4	100	1227	3.209	0.1%	1.0%	1.1%
	5	3.209	0	0	2.6%	0.0%	2.6%
		10.636	**119953**	**-**	**8.6%**	**97.2%**	**105.8%**
Flange	0	0	0	123.457	0.0%	0.0%	0.0%
	1	7.119	117567	5.490	5.8%	95.2%	101.0%
	2	100	663	4.727	0.1%	0.5%	0.6%
	3	100	2097	2.533	0.1%	1.7%	1.8%
	4	2.533	0	0	2.1%	0.0%	2.1%
		9.852	**120327**	**-**	**8.0%**	**97.5%**	**105.4%**
Repair	0	0	0	123.457	0.0%	0.0%	0.0%
	1	7.119	113406	10.051	5.8%	91.9%	97.6%
	2	100	7345	2.606	0.1%	5.9%	6.0%
	3	2.606	0	0	2.1%	0.0%	2.1%
		9.825	**120751**	**-**	**8.0%**	**97.8%**	**105.8%**