7th Framework Programme

FP7-ICT-2013-10 N.611145

Active semi-supervised learning in detecting tunnel cracks and other defects and final 3D measurement

Deliverable n.	D3.2	Active semi-supervised learning in detecting tunnel cracks and other defects and final 3D measurement	
Work package	3	Computer Vision for Real Time Defect Detection	
Editor(s)	Nikos Komodakis, Praveer Singh (ENPC)		
Status	Final		
Distribution	Confidential (CO)		
Issue date	06/10/2015	Creation date	01/07/2015

This project has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no 611145

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

REVISION CHART AND HISTORY LOG

REV	DATE	REASON
1	1/7/2015	First Version
2	14/9/2015	Second version for ICCS inputs
3	5/10/2015	Final version and Quality Review

EXECUTIVE SUMMARY

This deliverable is the accompanying report for the active semi-supervised learning algorithms that have been developed under the framework of WP3. The work that this report reflects is outcome of the tasks 3.2 and 3.3 on “Visual Detection of Cracks/Corrosion and Other Material Deterioration through Active Semi-Supervised Learning” and “3D Measures from Multi-View Cameras Views” respectively.

The work involved has included the data collection of defects’ data sets from the tunnels of the participating end-users (EOAE, VSH, LU) that has lead into the determination of the actual defects of interest that the system will be detecting. After the required processing of these data sets, the computer vision algorithms have been developed and configured to visually detect the aforementioned defects following the ROBO-SPECT requirements. The computer vision algorithm developed can be separated into three different parts as divided also into this report. The first is the defects’ detection system (developed by ENPC) that is able to detect tunnel surface defects (i.e. delamination, spalling etc). The second set of algorithms has been developed particularly for the crack detection and crack position localization on the tunnel surface. The third set of algorithms is the 3D calculation algorithm that is responsible for the precise crack localization in 3D coordinates and also the 3D representation of the crack/defect area of the tunnel. Precise evaluation based on defect/crack positive images has been executed to all algorithms to ensure their proper operation and ability to operate in real time but also their validation towards the ROBO-SPECT requirements. Closing, integration tasks have ensured the integration of the aforementioned technologies and algorithms with the robotic system that is responsible for controlling the overall inspection process. It can be noted that this report should be complemented by the D3.3 that describes the vision based 3D measurements in detail and is dedicated to the contribution of 3D vision to the ROBO-SPECT project and the stereo matching algorithm for dense 3D full reconstruction.

INTRODUCTION

This report presents the proposed defect detection algorithm through semi-supervised learning with deep learning features.

To begin with, the state-of-the-art methods for deep learning and semi-supervised learning are investigated in Chapter 1. The reason for using deep learning features is based on the observation that traditional features, like colour and texture, lack sufficient robustness for the challenging task of tunnel defect detection, while deep learning features carry richer information and have been shown to achieve higher performance in object recognition of natural images. The motivation of using semi-supervised learning is from the fact that it is difficult and time-consuming to label a large amount of tunnel images (due to it need civil specialists for labelling) and semi-supervised learning is able exploit unlabelled data to learn a more robust model for defect detection.

In Chapter 2, the proposed defect detection algorithm is detailed. Our proposed algorithm consists of three components, including deep learning, semi-supervised learning and defect segmentation. We first learn an initial deep learning model of convolutional neural networks (CNN) from a large-scale natural image dataset. Such a model is fine-tuned from a small set of labelled tunnel images for defect detection. To achieve a higher performance of defect detection, we also learn the model by using unlabelled tunnel image data further through semi-supervised learning techniques. The defect detection results are refined through the proposed defect segmentation model based on the framework of Markov random field. The proposed algorithm is evaluated on VSH tunnel image dataset. Experiments demonstrated the affectivity of each component and the promising results of the proposed method.

In Chapter 3, we can find an extensive description of the crack detection algorithms that have been developed in WP3. It includes descriptions of the algorithms for the visual inspection of concrete surfaces including the crack detection approach that has been followed and an extensive presentation and evaluation of the recent results.

In Chapter 4, the final 3D measures from multi-view cameras view are briefly introduced and more details about this can be found in Deliverable 3.3.

Finally, User Interface and Integration is illustrated in Chapter 5. The final goal of the project is to have a well integrated system which can easily detect cracks and other defects in the tunnel automatically and can then communicate to the end user leaving in place for him to take the final call. Thus in such a scenario, User interface and communication between the various sub-systems plays a critical role which we shall discuss in detail in this chapter.

1. STATE OF THE ART

Tunnel images may consist of complex scenes and may show low contrast between defects and background regions, traditional features lack sufficient robustness for detecting defects in tunnel images. Therefore, we investigate to exploit the recently suggested deep learning features for defect detection. To train a robust defect detection model, sufficient labeled tunnel images are necessary as well, unfortunately, it is difficult and time consuming to label tunnel images, due to the fact that it requires specialists to label the data. So we aim to exploit semi-supervised learning techniques, which also use unlabeled data to train the model so that fewer labeled data are needed. In the following, we review the state-of-the-art deep learning features and semi-supervised learning techniques, respectively.

1.1 Deep learning

Table 1: A summary of accronyms and there short description

Accronym	Short Description
HOG	Histogram of Oriented Gradients
SIFT	Scale Invarient Feature Transform
GPU	Graphics Processing Unit
CNN	Convolutional Neural Network
DNN	Deep Neural Networks
DBN	Deep Belief Networks
DBM	Deep Boltzman Machine
RBM	Restricted Boltzman Machine
BM	Boltzman Machine
SPN	Sum Product Networks
mcRBM	Mean-covariance Restricted Boltzman Machine
RNN	Recurrent Neural Network
HMM	Hidden Markov Model
HHMM	Hierarchical Hidden Markov Model
CRF	Conditional Random Fields
TDNN	Time Delay Neural Network
HTM	Hierarchical Temporal Memory

Computer Vision researchers have long been using human-engineered feature extraction techniques like HOG, SIFT until the recent success of Convolutional Neural Networks (CNN's) on tasks of Image Classification and detection. Though the major focus in this field has mostly drifted towards learning these features end to end using deep learning models, yet Vision research embodies both the components from low-level features to higher-level semantic information[1][2]. Since 2006, deep learning, which is more recently referred to as representation learning, has emerged as a new area of machine learning research [3][4][5].

Deep learning refers to a class of machine learning techniques, where many layers of information-processing stages in hierarchical architectures are exploited for pattern classification and for feature or representation learning. It is in the intersections among the research areas of neural network, graphical modeling, optimization, pattern recognition, and signal processing. There are three important reasons which have lead to the current popularity of deep learning today : 1) drastically increased chip processing abilities (e.g., GPU units) 2) significantly lowered cost of computing hardware and 3) recent advances in machine learning and

signal/information processing research. Active researchers in this area include those at University of Toronto, New York University, University of Montreal, Microsoft Research, Google, IBM Research, Baidu, Facebook, Stanford University, University of Michigan, MIT, University of Washington, and numerous other places. These researchers have demonstrated successes of deep learning in diverse applications of computer vision, phonetic recognition, voice search, conversational speech recognition, speech and image feature coding, semantic utterance classification, hand-writing recognition, audio processing, visual object recognition, information retrieval, and even in the analysis of molecules that may lead to discovering new drugs as reported recently in [9].

As described earlier, Deep learning is a branch of Machine Learning which uses several layers of neural networks with additional non-linearity's and the entire learning is done end to end in a hierarchical fashion. Depending upon the style of architecture and the supporting techniques, deep learning can be broadly classified into three main architectures -

1) Generative deep architectures
2) Discriminative deep architectures
3) Hybrid deep architectures

By machine learning tradition (e.g., [34]), it may be natural to use a two-way classification scheme according to discriminative learning (e.g., neural networks) versus deep probabilistic generative learning (e.g., DBN, DBM, etc.).This classification scheme, however, misses a key insight gained in deep learning research about how generative models can greatly improve learning DNNs and other deep discriminative models via better optimization and regularization. Also, deep generative models may not necessarily need to be probabilistic; e.g., the deep auto encoder. Nevertheless, the two-way classification points to important differences between DNNs and deep probabilistic models. The former is usually more efficient for training and testing, more flexible in its construction, less constrained (e.g., no normalization by the difficult partition function, which can be replaced by sparsity), and is more suitable for end-to-end learning of complex systems (e.g., no approximate inference and learning). The latter, on the other hand, is easier to interpret and to embed domain knowledge, is easier to compose and to handle uncertainty, but is typically intractable in inference and learning for complex systems.

This distinction, however, is retained also in the proposed three-way classification.

Below we briefly review representative work in each of the above three classes.

1.1.1 Generative architectures

Associated with this generative category, we often see "unsupervised feature learning", since the labels for the data are not of concern. When applying generative architectures to pattern recognition (i.e., supervised learning), a key concept here is (unsupervised) pretraining. This concept arises from the need to learn deep networks but learning the lower levels of such networks is difficult, especially when training data are limited. Therefore, it is desirable to learn each lower layer without relying on all the layers above and to learn all layers

in a greedy, layer-by-layer manner from bottom up. This is the gist of “pretraining” before subsequent learning of all layers together.

Among the various subclasses of generative deep architecture, the energy-based deep models including auto encoders are the most common (e.g.,[4][36][37][38]). The original form of the deep auto encoder[21][30], is a typical example in the generative model category. Most other forms of deep auto encoders are also generative in nature, but with quite different properties and implementations. Examples are transforming auto encoders[39], predictive sparse coders and their stacked version, and denoising auto encoders and their stacked versions [27].

Specifically, in denoising auto encoders, the input vectors are first corrupted; e.g., randomizing a percentage of the inputs and setting them to zeros. Then one designs the hidden encoding nodes to reconstruct the original, uncorrupted input data using criteria such as KL distance between the original inputs and the reconstructed inputs. Uncorrupted encoded representations are used as the inputs to the next level of the stacked denoising auto encoder.

Next we move onto another class of generative models called the deep Boltzmann machine or DBM[40][41][42]. A DBM similar to other DNN's is composed of various hidden layers but in fact has no connections between the neurons in the same layer. Each top layer of the DBM captures a high level correlation of the layer just below it.

Restricted Boltzmann Machine's or RBM are a special category of DBM's when we have simply one hidden layer in the entire network. However by stacking many RBM's, we can basically yield a Deep Boltzmann Machine or DBM in which all of the hidden layers can be learnt together by passing the activations from one bottom layer as the output to its next upper layer. Applications of BM's at the bottom layer of DBN can be seen in phone recognition [43]. This model is called mean-covariance RBM or mcRBM and has several other applications like [26].

SPN's or Sum-Product Networks are another set of Deep Generative Models which is arranged in the form an acyclic graph having leaves as the data and the nodes in the form of sum / product operation. Gradient dilution is one of the problem faced while training SPN's which has been solved to some extent in [44][45].

Another very powerful class of deep generative architectures have been RNN's which can generate and model sequential data (e.g.,[46]). A very common problem faced while training RNN's have been the vanishing gradient problem which has been partially tackled in[47] using second-order information. Applications of RNN's can be seen in character level language modeling as in[48] where a Hessian free optimization has been used. Bengio*et al.[49]*andSutskever[50] have come out with new methods in training generative RNNs which can outperform Hessian-free optimization methods. RNN's have shown excellent results on Language models and oral language understanding as reported in *[51]*[52].

Probabilistic generative models have also been used for tasks such speech recognition[53][54][55][56][57][58][59], speech recognition robustness [60][61] as well as for speech synthesis [62][63][64][65][66]and speech generation [15][16][67][68][69][70][71][72][73]. [74][75][76] show more

efficient work of this deep architecture. In the works of [77][78][79][80][81] these can be seen as a more generalized form of Bayesian as well as graphical models.

Combining the shallow HMM's with the higher layers of deep models is also another methodology used in speech recognition tasks as illustrated in the recent book [82] on “Hierarchical HMM or[83] for HHMM and [84]for Layered HMM.

Temporally recursive and deep generative models as in[85]can also be used for human motion modeling, and in[86]for natural language or scene parsing.

1.1.2 Discriminative architectures

We have seen in the past, various discriminative techniques being applied to HMMs (e.g., [87][88][89][90][91][92][93][94]) or CRFs (e.g., [95][96][97][98][99][100])[101]. Deep Structured CRF's have seen applications in phone recognition [102], spoken language identification[103], and natural language processing [96]. To have a more detailed overview of varied applications of discriminative models, see [33][104][105][106][108][109][110][111][112][11][114][115][116][117].

CNN's or Convolutional Neural Networks are another type of discriminative model composed of layers undergoing convolutions on the activations from the preceding layers and max pooling layers followed by DNN's on top. In order to reduce the number of parameters weight sharing is performed in convolutional layers and to cater into invariance properties we have these pooling layers with more principled way of adding invariance as seen in[39]. CNN's are effectively the most widely used models in important computer vision tasks such as image recognition and detection[118][119][120][121] or speech recognition[122][123][124][125][126].

A special case of CNN's are the Time Delay neural networks (TDNN) where weight sharing is limited to only the time domain[127][129]. Frequency domain is another dimension where weight sharing and pooling inputs has shown promising results [122][123][126].

HTM's or Hierarchical Temporal Memory[17][128][130]is another variant of CNN where we use both bottom up and top down information flow and temporal dimension is used for supervisory tasks too.

Architectures for detection based speech recognition tasks as proposed in [131][132][133][134] which uses the DBN–DNN technique, also come under discriminative deep architecture category with learning being performed mostly through back propagation[135]. All of these encompass simplified units from the early works of[136][137].

1.1.3 Hybrid generative–discriminativearchitectures

The term “hybrid” for this third category refers to the deep architecture that either comprises or makes use of both generative and discriminative model components. Inman existing hybrid architectures published in the literature(e.g., [21][23][25][138]), the generative component is exploited to help with discrimination, which is

the final goal of the hybrid architecture. How and why generative modeling can help with discrimination can be examined from two view points:

1) The optimization viewpoint where generative models can provide excellent initialization points in highly nonlinear parameter estimation problems (the commonly used term of “pretraining” in deep learning has been introduced for this reason); and/or2) The regularization perspective where generative models can effectively control the complexity of the overall model.

The study reported in [139] provided an insightful analysis and experimental evidence supporting both of the viewpoints above.

When the generative deep architecture of DBN is subject to further discriminative training using back prop, commonly called “fine-tuning” in the literature, we obtain an equivalent architecture of the DNN. The weights of the DNN can be “pretrained” from stacked RBMs or DBN instead of the usual random initialization. See[24] for a detailed explanation of the equivalence relationship and the use of the often confusing terminology.

Another example of the hybrid deep architecture is developed in [23], where again the generative DBN issued to initialize the DNN weights but the fine tuning is carried out not using frame-level discriminative information(e.g., cross-entropy error criterion) but sequence level one. This is a combination of the static DNN with the shallow discriminative architecture of CRF. Here, the overall architecture of DNN–CRF is learned using the discriminative criterion of the conditional probability of full label sequences given the input sequence data. It can be shown that such DNN–CRF is equivalent to a hybrid deep architecture of DNN and HMM whose parameters are learned jointly using the full-sequence maximum mutual information (MMI) between the entire label sequence and the input vector sequence. A closely related full-sequence training method is carried out with success for a shallow neural network [140] and for a deep one [141].

Here, it is useful to point out a connection between the above hybrid discriminative training and a highly popular minimum phone error (MPE) training technique for the HMM [89]. In the iterative MPE training procedure using extended Baum–Welch, the initial HMM parameters cannot be arbitrary. One commonly used initial parameter set is that trained generatively using Baum–Welch algorithm for maximum likelihood. Furthermore, an interpolation term taking the values of generatively trained HMM parameters is needed in the extended Baum–Welch updating formula, which may be analogous to “fine tuning" in the DNN training discussed earlier. Such I-smoothing[89] has a similar spirit to DBN pretraining in the “hybrid” DNN learning.

Along the line of using discriminative criteria to train parameters in generative models as in the above HMM training example, we here briefly discuss the same method applied to learning other generative architectures. In [142],the generative model of RBM is learned using the discriminative criterion of posterior class/label probabilities when the label vector is concatenated with the input data vector to form the overall visible layer in the RBM. In this way, RBM can be considered as a stand-alone solution to classification problems and the authors derived a discriminative learning algorithm for RBM as a shallow generative model. In the more recent work of [146], the deep generative model of DBN with the gated MRF at the lowest level is learned for feature extraction and then for recognition of difficult image classes including occlusions. The generative ability of the

DBN model facilitates the discovery of what information is captured and what is lost at each level of representation in the deep model, as demonstrated in[146]. A related work on using the discriminative criterion

of empirical risk to train deep graphical models can be found in [81].

A further example of the hybrid deep architecture is the use of the generative model of DBN to pre-train deep convolutional neural networks (deep DNN)[123][144][145]). Like the fully-connected DNN discussed earlier, the DBN pretraining is also shown to improve discrimination of the deep CNN over random initialization.

The final example given here of the hybrid deep architecture is based on the idea and work of[147][148], where one task of discrimination (speech recognition) produces the output (text) that serves as the input to the second task of discrimination (machine translation). The overall system, giving the functionality of speech translation – translating speech in one language into text in another language – is a two-stage deep architecture consisting of both generative and discriminative elements. Both models of speech recognition(e.g., HMM)and of machine translation (e.g., phrasal mapping and non-monotonic alignment) are generative in nature. But their parameters are all learned for discrimination. The framework described in [148] enables end-to-end performance optimization in the overall deep architecture using the unified learning framework initially published in[90]. This hybrid deep learning approach can be applied to not only speech translation but also all speech-centric and possibly other information-processing tasks such as speech information retrieval, speech understanding, cross lingual speech/text understanding and retrieval, etc. (e.g.,[11][109][149][150][153]).

1.2 Semi-supervised learning

Semi-supervised learning is initially motivated by its practical value in learning faster, better, and cheaper. In many real world applications, it is relatively easy to acquire a large amount of unlabeled data. For example, documents can be crawled from the Web, images can be obtained from surveillance cameras, and speech can be collected from broadcast. However, their corresponding labels for the prediction task, such as sentiment orientation, intrusion detection, and phonetic transcript, often requires slow human annotation and expensive laboratory experiments. This labeling bottleneck results in a scarce of labeled data and a surplus of unlabeled data. Therefore, being able to utilize the surplus unlabeled data is desirable.

Recently, semi-supervised learning also finds applications in cognitive psychology as a computational model for human learning. In human categorization and concept forming, the environment provides unsupervised data (e.g., a child watching surrounding objects by herself) in addition to labeled data from a teacher (e.g., Dad points to an object and says “bird!”). There is evidence that human beings can combine labeled and unlabeled data to facilitate learning.

One of the earliest examples of the empirical advantages of SSL was co-training, a method first developed for text mining problems [1]and later extended in various forms to other applications[157][158]. Therein, multiple classifiers are first estimated using conditionally independent feature sets of training data. The performance

advantages of this method rely heavily on the existence of independent and complementary classifiers. Theoretical results show that some mild assumptions on the underlying data distribution are sufficient for co-training to work [159][160]. However, performance can dramatically degrade if the classifiers do not complement each other or the independence assumption does not hold [161]. Though co-training is conceptually similar to semi-supervised learning due to the way it incorporates unlabeled data, the classifier training procedure itself is often supervised.

The extension of traditional supervised support vector machines (SVMs) to the semi-supervised scenario is another widely used SSL algorithm. Instead of maximizing separation (via a maximum margin hyper plane) over training data as in standard SVMs, semi-supervised SVMs (S3VMs) estimate a hyper plane to balance maximum-margin partitioning of labeled data while encouraging a separation through low-density regions of the data [162]. For example, transductive support vector machines (TSVMs) were developed as one of the earliest incarnations of semi-supervised SVMs [163].1 Various optimization techniques have been applied to solve S3VMs[165], resulting in a wide range of methods, such as low density separation [164], semi-definite programming based methods [166][167], and a branch-and-bound based approach [165].

Another family of SSL methods known as graph-based approaches have recently become popular due to their high accuracy and computational efficiency. Graph-based semi-supervised learning(GSSL) treats both labeled and unlabeled samples from a data set as vertices in a graph and builds pair-wise edges between these vertices which are weighted by the affinity between the corresponding samples. The small portion of vertices with labels are then used by SSL methods to perform graph partition or information propagation to predict labels for unlabeled vertices. For instance, the graph min-cuts approach formulates the label prediction as a graph cut problem [168][169][170]. Other GSSL methods, like graph transductive learning, formulate the problem as regularized function estimation over an undirected weighted graph. These methods optimize a trade-off between the accuracy of the classification function on labeled samples and a regularization term that favors a smooth function. The weighted graph and the optimal function ultimately propagate label information from labeled data to unlabeled data to produce transductive predictions. Popular algorithms for GSSL include graph cuts [168][169][170][171][172], Greedy Max-Cut[182], graph random walks[173][174], manifold regularization [175][176][177][178], and graph regularization[179][180]. Comprehensive survey articles have also been disseminated[181].

2. VISUAL DETECTION ALGORITHMS

This section describes our proposed visual detection algorithms, consisting of deep learning features, semi-supervised learning and defect segmentation, and then demonstrates the experimental results.

2.1 Deep learning

In Tunnel images, heterogeneous salient objects of interest (defects) generally show low contrast to cluttered background. From our previous investigation (see Deliverable 3.1) we found that traditional features (e.g., scale-invariant feature transform[184], speeded up robust features[185], histogram of oriented gradients[186], and local binary patterns[187])lack sufficient robustness to address the challenging tunnel defects detection. Therefore, to address such an issue we propose to exploit deep learning features of Convolutional neural networks (CNNs), which have recently, show success in a variety of computer vision tasks.

To train CNN models for feature description, a very large set of labeled data is necessary. Unfortunately, tunnels generally do not have many defects, and data labeled data is difficult and time consuming to obtain due to the fact that it requires a specialist to label the data. Moreover, there is no publicly available dataset for the specific task of tunnel defects detection. To address this issue, we firstly train an initial CNN model based on large natural images dataset, e.g. Image Net[188], and then such a pre-trained model is fine-tuned by using a small set of tunnel images.

2.1.1 The ImageNet dataset

Many of the deep learning models like alexnet, RNN, SPP are originally trained from scratch on a large database called the Image Net due to their large number of parameters to prevent over fitting. This database has nearly 15 million images spread across 22000 category. However for the purpose of training, only a subset of this large database is used i.e. 1.2 million images from 1000 categories for training , another 50k for validation, and another 150k for testing.

2.1.2 Convolutional neural networks

Though Deep learning gained momentum only in 2012 after the seminal paper of Krizhevsky et al. [190], CNN's have been used in the past by Vision researchers for varied tasks. For e.g. in the 1990's YannLeCun and his team mates showed how CNN's could be used for document recognition [189]. However due to its computational complexity Support Vector Machines were instead adopted for classification tasks. It was only with the advent of high performance Graphical Processing Units, that in 2012, Krizhevsky et al. [190] rekindled some interest in CNN by training a large deep model comprising of multiple layers of convolutions and max pooling using the Image Net dataset which significantly improved the performance on the Image Net Large Scale Visual Recognition Challenge(ILSVRC) as compared to other human engineered techniques. We employ a similar CNN architecture of Krizhevsky et al. [190] for our tunnel defect detection.

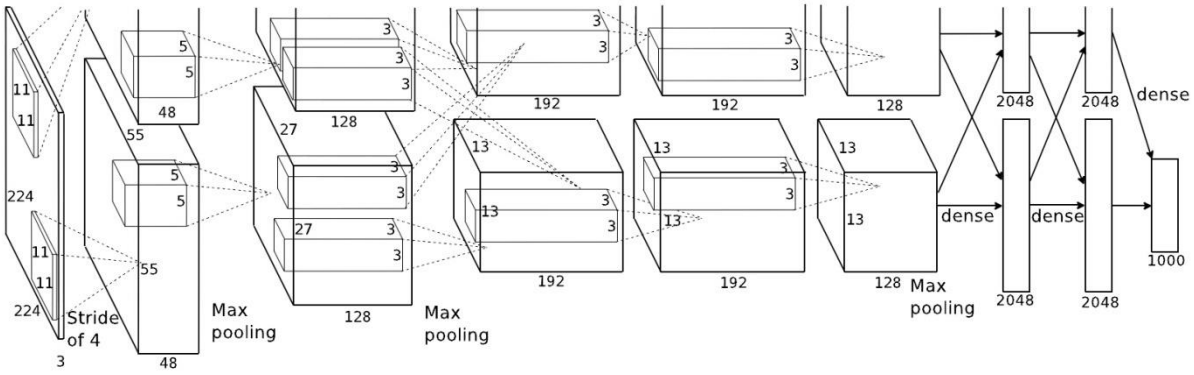


Figure 1: The used CNN architecture from [190], explicitly showing the delineation of responsibilitiesbetween the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-partsat the bottom.

As shown in Figure 1: The used CNN architecture from [190], explicitly showing the delineation of responsibilitiesbetween the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-partsat the bottom., the net is mainly composed of 5 convolutional layers and 3 fully connected layers. The 5 convolutional layers (or conv-layers) are equally shared among the 2 GPU's with the kernels of only the 3rd conv-layer being applied to activations coming from both the GPU's of the previous layer; otherwise for the 2nd, 4th and 5th conv layers the kernels situated on one GPU only scan the activations from the previous layer of that particular GPU. An additional non-linearity (Rectified Linear Units or RELU's) is also applied to the activations of these layers which improves the overall convergence of training. To induce invariance (mostly translational invariance) in the features, max pooling layers are added after the 1st, 2nd and 5th conv layer. As the name suggests, fully connected layers have all the neurons in a particular layer connected through weights to all the neurons in the previous layers.

The first conv-layer has 96 kernels divided equally (48 on each GPU) with a size of 11X11X3 and a stride of 4. Subsequent conv-layers have 256,384,384 and 256 kernels with sizes 5X5X48, 3X3X256 and 3X3X192. Each of the fully connected layer has 4096 neurons respectively with an additional dropout layer which randomly masks (with say probability p) p times the number of neurons in the previous fully connected layer. This is a kind of regularization imposed on the network and have shown to improve the performance by some points.

In general, our CNN training procedure follows that of[190], learning on Image Net ILSVRC using gradient descent with momentum. The hyper-parameters are the same as used by[190].The layers are initialized from a Gaussian distribution with a zero mean and variance equal to 10^{-2}. We also employ similar data augmentation in the form of random crops, horizontal flips, and RGB color jittering.

2.1.3 CNN fine-tuning on tunnel images dataset

To adapt the previously described CNN to the tunnel defects detection, we removed the full-connected layers (layer 6 and layer 7) and replaced the 1000-way classification layer with a randomly initialized (N+1)-way classification layer (where N is the number of defect classes, plus 1 for background). Then each of 256-dimentional feature vectors from layer 5 inputs to (N+1)-way classification layer. We continue stochastic gradient descent training of CNN parameters using tunnel imagery data, where the feature vectors correspond to defects in original input image are set to positive examples and others are set to negative examples.

Apart from performing classification using the outputs layer 5, we also investigated to combine it with previous layers for defect classification, by using the same stochastic gradient descent training method to learn CNN parameters.

2.2 Salient Object Detection

This step is optional. However since the image we have is a high resolution one, we can basically use Salient Object detection technique to detect which part of the image is salient enough and then apply the above mentioned deep learning approach to compute features for only that particular part of the image.

Saliency detection is important to many applications, such as object segmentation, object recognition, content-based image retrieval, and adaptive image/video coding. The original task of saliency detection aims to predict fixation points in an image, where the earliest study on this topic is motivated by the observation from cognitive scientists that the inherent visual attention mechanism enables humans to identify rapidly and effortlessly the visually outstanding(salient) regions/objects in complex scenes.

Following the fixation prediction, saliency detection has been recently extended to mean highlighting the whole salient object in an image. The task of salient object detection in the computer vision literature, is essentially equivalent to solving a binary foreground/background segmentation problem.

A common characteristic of most existing methods for salient object detection is that they generally simplify the original images by partitioning them into blocks or segmenting them into regions through image segmentation algorithms or pixel clustering methods. This is done for efficiency but, most importantly, for computing visual features of wider image support with the hope that these features will thus be more robust. The fixed block partition inevitably merges object pixels into background for those blocks surrounding the object boundaries, while image segmentation methods are typically boundary-preserved and ensure that the pixels within the same region share certain visual characteristics. Thus the state-of-the-art salient object detection models typically compute visual features based on regions for saliency evaluation, where it is important to note that the robustness and richness of these features also determine to a large extent performance of saliency detection.

However, one problem from this is that the features extracted from small regions might be not sufficiently discriminative for detecting salient objects in complex scenes. A suggestion might be to adjust the

segmentation parameters so that an object is composed of very few regions to facilitate the salient object detection task. Unfortunately, natural images may contain a variety of complex scenes, and the state-of-the-art image segmentation methods are still far from separating the whole objects with the well-defined boundaries from background regions. For some images, adjusting the segmentation parameters to decrease the number of regions may result in under-segmentation, where salient object regions are merged into background, and lead to inaccurate salient object detection. As a result, even the state-of-the-art saliency models still have significant difficulties in completely highlighting salient objects in complex scenes. Therefore, given the above discussion, a fundamental question that needs to be addressed is the following: how can we extract features that are both more robust and also richer based on the over-segmented regions for salient object detection?

To that end, we need to have a novel hierarchy-associated feature construction framework for salient object detection, which integrates various elementary features from both the target region and super-regions in a hierarchy. Features computed in this manner are able to represent the context of the entire object/background and are much more discriminative as well as robust for salient object detection. In this context we can also introduce the use of enriched elementary features, which are computed from the outputs of multiple hidden layers of a deep convolutional neural network (CNN) and complement traditionally used features, such as color and texture. By regional contrasts evaluation, these CNN features and other typical elementary features can be effectively incorporated into the proposed feature construction framework to generate hierarchy-associated rich feature(HARF). With such a rich feature representation, we cast saliency detection as a regression problem for which a boosted predictor is trained to estimate regional saliency scores.

Since the end results for defect detection algorithm is a binary map, hence to construct a binary segmentation tree for an input image, we can exploit the gPb (globalized probability of boundary based contour detection) method to generate an ultra metric contour map (UCM), which contains a set of real-valued contours. We normalize the UCM into the range of [0, 1] and generate a segmentation containing approximately 100 regions by thresholding with appropriate contour values. Then the generated regions $R_i(\mathrm{i} = 1,..\mathrm{K})$ are the basis to construct a binary segmentation tree, in which each leaf node represents a region and each non-leaf node represents a super-region. For generating super-regions in the segmentation tree, the merging order is determined by contour values of adjacent regions, i.e.,

$$R_s = argminC(R_n, R_j), \mathrm{R}_n \in \mathbb{N}_j$$

where $\mathbb{N}_j$ denotes the set of neighbors of the target region R_j, and R_n, R_j denotes the contour value between regions R_n and R_j in the UCM. This means that the pair of regions with the lowest contour value are merged first to generate a new super-region R_s. The merging process is performed iteratively to generate the binary segmentation tree until the final two super-regions are merged to compose the entire image.

Based on the resulting binary segmentation tree, we now compute our hierarchy-associated rich features (HARF) that correspond to regions at leaf-nodes using the following method.

$HARF_1$: In this case, basic regional features are combined, from both the local region and more global super-regions to form HARF. Specifically, basic regional features are computed first for both the target region to be represented and for super-regions corresponding to its β ancestor nodes in the segmentation tree. Then the extracted basic regional features from them are stacked into a single feature vector. For abbreviation, the feature extracted in this form is denoted as$HARF_1^{\beta}$ (R) where R represents the target region and β is the number of levels of super-regions in the segmentation hierarchy included to compute HARF. Therefore, assuming that the basic regional feature for each region or super-region is represented by a d-dimensional feature vector, the dimensionality of the generated $HARF_1^{\beta}$ feature is (d+d) X β.

With the HARF representation for each leaf region of the binary segmentation tree, we cast salient object detection as a regression problem that predicts the saliency of a region. For the regression, we used the AdaBoost algorithm in our experiments due to its efficiency in both training and testing. Such a method iteratively assembles weak decision trees to generate a single composite strong learner. In this case, given a set of training regions represented by HARF $\{H_1, \ldots\ldots, H_T\}$ along with their ground-truth labeling, we learn a boosted regressor of the form $w(H_k) = \sum_u \alpha_u T_u(H_k)$

whereα_u and $T_u(H_k)$are the trained coefficient and the weak decision tree, respectively. We use equation of AdaBoost to obtain the predicted score of the boosted regressor for a test region R_k, represented by HARF descriptor H_k. For reasonable saliency precision, we compute the saliency score s_k of R_k by further fitting the output of the boosted regressor into the range of [0, 1] with a sigmoid function, i.e.,

$$s_k = \frac{1}{1 - e^{-w(H_k)}}$$

2.3 Semi-supervised & transfer learning

Traditional classifiers use only labeled data (feature / label pairs) to train. Due to the nature of tunnel defects detection, labeled data is difficult and time consuming to obtain (due to the fact that it requires a specialist to label the data). However, unlabelled data is abundant and much easier to be collected. Therefore, we propose to employ semi-supervised learning to solve these problems by combining the labeled and unlabelled data to build better classifiers. As illustrated in Figure 2: In a binary classification problem, if we assume each class has a Gaussiandistribution, then we can use unlabeled data to help parameter estimation., by using unlabeled data semi-supervised learning technique is able to achieve better classification performance compared to traditional learning methods.

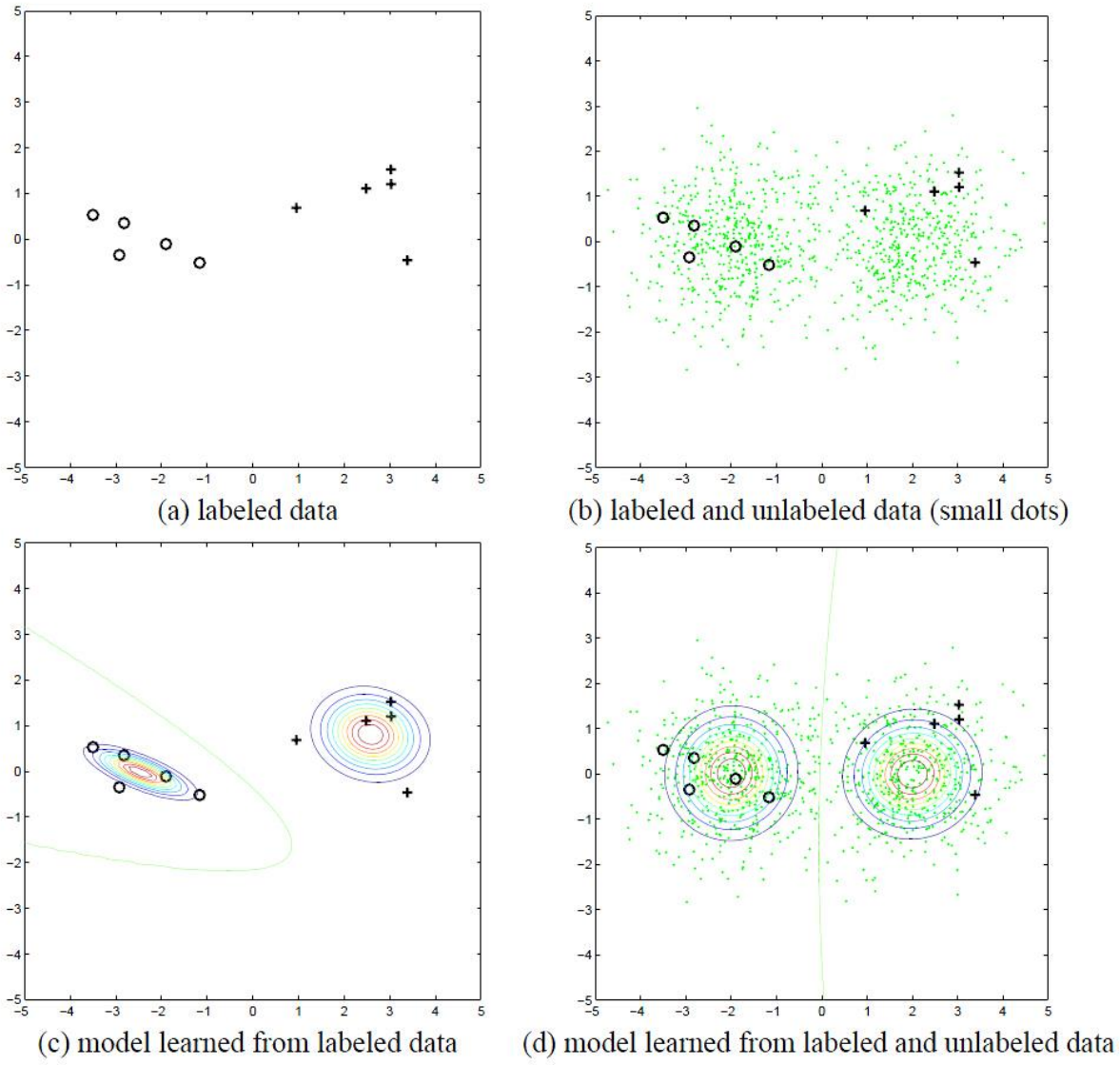


Figure 2: In a binary classification problem, if we assume each class has a Gaussiandistribution, then we can use unlabeled data to help parameter estimation.

Recent research in Semi-Supervised learning has focused mainly upon graphical methods, where the nodes of the graph are treated as the labeled or unlabelled data and the edges of the graph are the pair wise distance of the incident nodes.

Considering $G = (V, E)$to be a graph with real edge weights given by$\omega: E \to \mathbb{R}$ with the weight $\omega(e)$ of an edge e representing the similarity of the incident nodes. Thus the weighted adjacency matrix W of the graph $G$is defined as -

$$W_{ij} = \begin{cases} \omega(e) & \text{if} e = (i,j) \in E \\ 0 & \text{if} e = (i,j) \in E \end{cases}$$

Since the number of unlabeled samples is huge in tunnel defect detection, learning full-size precision model is inefficient. We build an approximate neighborhood graph using Anchor Graphs (Liu et al., 2010), in which the similarity between a pair of data points is measured with respect to a small number of anchors (typically a few hundred). The resulting graph is built in O(n) time and is sufficiently sparse with performance approaching to the true kNN graph as the number of anchors increases.

Suppose a soft label prediction function $\mathrm{f}: \mathbb{R}^{\mathrm{d}} \rightarrow \mathbb{R}$ defined on the input samples $\mathrm{X} = \{\mathrm{x}_1\}_{\mathrm{i}=1}^{\mathrm{n}}$.Withoutloss of generality, we assume that the first l samples arelabeled and the rest remain unlabeled. To work underlarge scale,the label prediction function can be a weighted average of the labels on a subset of anchor (landmark) samples. As such, if one can infer the labels associated with the much smaller subset, the labels of other unlabeled samples will be easily obtained by a simple linear combination.

The idea is to use a subset $\mathrm{U} = \{\mathrm{u}_\mathrm{k}\}_{\mathrm{k}=1}^{\mathrm{m}} \in \mathbb{R}^{\mathrm{d}}$ in which each u_k acts as an anchor point since we represent f in terms of these points, i.e.,

$$\mathrm{f}(\mathrm{x}_\mathrm{i}) = \sum_{\mathrm{k}}^{\mathrm{m}} Z_{ik} f(\mathrm{u}_\mathrm{k})$$

whereZ_{ik} are sample-adaptive weights, such a label prediction essentially falls into nonparametric regression. Let us define two vectors $\boldsymbol{f} = [f(\mathrm{x}_1), \ldots, f(\mathrm{x}_\mathrm{n})]^T$ and $\boldsymbol{a} = [f(u_1), \ldots, f(u_m)]^T$ and rewrite the previous equation as

$$\boldsymbol{f} = Z\boldsymbol{a}, \quad Z \in \mathbb{R}^{n \times m}, m \ll n$$

In the above formula the computational burden has been largely mitigated by reducing the solution space from large $\boldsymbol{f}$to much smaller$\boldsymbol{a}$.

As in [8], we take these anchor points $\{\mathrm{u}_\mathrm{k}\}$ as k-means clustering centers instead of randomly sampled exemplars because it turns out that k-means clustering centers have a stronger representation power to adequately cover the vast point cloud$\mathrm{X}$.

2.4 Defect segmentation

To further refine defect classification results, we designed an object (defect) segmentation model based on the probabilistic classification outputs from previous classification model.

object segmentation is explicitly formulated as a binary pixel labeling problem, which can be solved under the framework of graph cut. The input image is represented using an undirected graph$\mathrm{G} = (\mathrm{V}, \mathrm{E})$ where V is a set of nodes and E is a set of undirected edges connecting these nodes. Each node in the graph represents each pixel in the image, and there are two additional terminals in the graph, i.e., object terminal$\mathrm{S}$ and and background terminal$\mathrm{T}$. There are two types of edges in the graph. Specifically, edges between neighboring nodes are called n-links where n stands for “neighbor”, and edges connecting nodes to terminals are called t-links where stands for t “terminal”. All graph edges including n-links and t-links are assigned with some

nonnegative costs.Formally, let N denotes the set of all pairs of neighboring pixels in P, which denotesthe set of all pixels in the image. The two sets, V andE, are represented as follows:

$$V = P \cup \{S, T\}$$

$$E = N \cup_{p \in P} \{(p, S), (p, T)\}$$

where all n-links are included in N, (p, S) and (p, T) denote the t-link connecting with S and T, respectively.

A cut is defined as a subset of edges $c \subset E$, and nodes in the graph are separated by this subset of edges. Graph cut seeks to minimize a cost function with the following form to determine the optimal label configuration:

$$E(L) = \sum_{p \in P} R_{L_p}(p) + \gamma \sum_{(p,q) \in N} B(p, q) \cdot \delta_{L_p \neq L_q}$$

where $L = \{L_p\}$ is a binary vector denoting any possible label configuration of all pixels, l_p can be assigned with the label "0" for background or "1" for salient object, and the Kronecker delta $\delta_{L_p \neq L_q}$ is defined as

$$\delta_{L_p \neq L_q} = \begin{cases} 1, & \text{if} L_p \neq L_q \\ 0, & \text{if } L_p = L_q \end{cases}$$

$R_{L_p}(p)$is the data term based on the label L_p, $B(p, q)$ is the smoothness term m for neighboring pixels (p, q), and γ is the weight for balancing the two terms. The data term $R_{L_p}(p)$ penalizes s the inconsistency between a label L_p and the observed data such as saliency value and color feature of a pixel p, and the smoothness term $B(p, q)$ penalizes the label discontinuity of neighboring pixels (p, q) The minimum cut of the graph can be efficiently solved using the max-flow algorithm and the corresponding binary labels of pixels are used to represent the salient object segmentation result.

2.4.1 Data term

The data term measures consistency between the pixel and its label, and is generally defined as the negative log of the likelihood of a foreground/background label being assigned to a pixel. We define the data term from the last layer (classification layer) of the deep learning networks, i.e.,

$$R_{L_p}(p) = -\log(\Phi(L_p))$$

where$\Phi(L_p)$ indicates the probabilistic output of the last layerof the deep learning networks for pixel p.

2.4.2 Smoothness term

The smoothness term is defined within the neighborhood system which consists of all pairs of adjacent pixels. Its goal is to ensure the overall label smoothing by penalizing neighboring pixels assigned with different labels. The smoothness term is defined based on the spatial distance and color contrast between neighboring pixels

$$B(p, q) = \frac{\rho}{\text{dis}(p, q)} [L_p \neq L_q] \exp\{-\beta \|x_p - x_q\|_2^2\}$$

where$\text{dis}(p, q)$ is the spatial Euclidean distance of neighboring pixels, $\|\cdot\|_2$ indicates $l_2 - \text{norm}$. The balance parameter ρ isset to 50 which has been shown to be suitable for most images. The constant β is a contrast-

oriented weight. When β is 0, all neighboring pixels are smoothed with the fixed degree determined byρ. To make the smoothnessadaptive to global contrast of neighboring pixels, β is chosento be

$$\beta = \frac{1}{2 \cdot mean\left(\left(\left\|x_p - x_q\right\|_2^2\right) \cdot \mathrm{dis}(p,q)\right)}$$

2.5 Experiments

In this section, we show experimental results obtained on VSH tunnel dataset. The objective evaluation criterion is presented first, and then we validate the performance of the proposed approach using different configurations.

2.5.1 Evaluation criterion

To objectively evaluate the performance of the proposed tunnel defect detection approach, we adopt F-measure criterion, which is the harmonic mean of precision and recall

$$\mathrm{F-measure} = \frac{(1+\alpha)\mathrm{precision} \cdot \mathrm{recall}}{\alpha \cdot \mathrm{precision} + \mathrm{recall}}$$

whereα is the parameter to balance precision and recall and is set to 1 in our experiments. The precision and recall are computed over the total dataset and are defined as

$$\mathrm{precision} = \frac{1}{\mathrm{T}}\sum_{1}^{\mathrm{T}} \frac{\mathrm{P_t} \cap \mathrm{G_t}}{\mathrm{P_t}}$$

$$\mathrm{recall} = \frac{1}{\mathrm{T}}\sum_{1}^{\mathrm{T}} \frac{\mathrm{P_t} \cap \mathrm{G_t}}{\mathrm{G_t}}$$

where T is the number of test images, $\mathrm{P_t}$ is the set of predicted foreground pixels in test image t and $\mathrm{G_t}$ is the ground-truth.

2.5.2 Performance analysis

We report experimental results on VSH dataset by using the different configurations.

2.5.2.1 Validation of different convolutional layers of CNN

We investigated using different convolutional layers of convolutional neural networks (CNN) of deep learning for tunnel defect detection, including the second layer (conv.2), the third layer (conv. 3), the fourth layer (conv. 4), the fifth layer (conv. 5) and the combination of all of them (conv. 2-5). The first layer is not included, because it is too level feature and has been shown unhelpful for the object defection task.

Figure 3 shows the precision, recall and F-measure by using different CNN convolutional layers. Clearly, the combination of all layers achieved higher performance. The precision improvement is very obvious and results in a higher F-measure.

Figure 4 and Figure 5 show some defect detection results generated by using different CNN convolutional layers. Obviously, the proposed approach by using each CNN layer features performs well for the defect detection. Better results are further obtained by combining all convolutional layers (conv. 2-5), where the defect detection results are more matched with manual labeling (ground-truth).

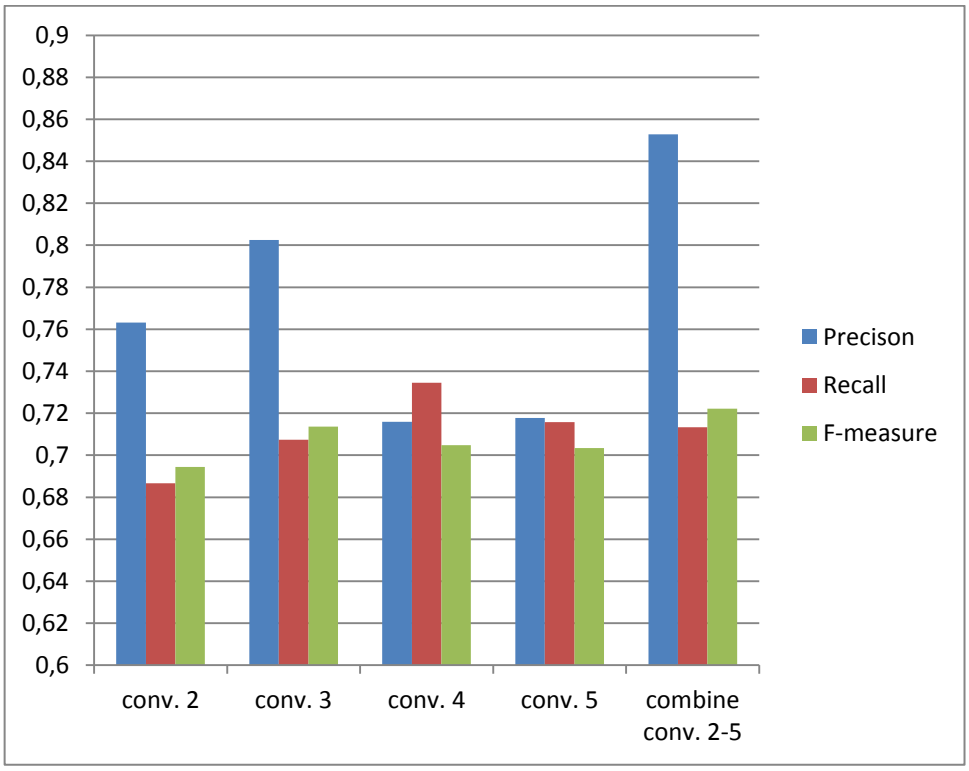


Figure 3: Precision, recall and F-measure by using different convolutional layers of convolutional neural networks

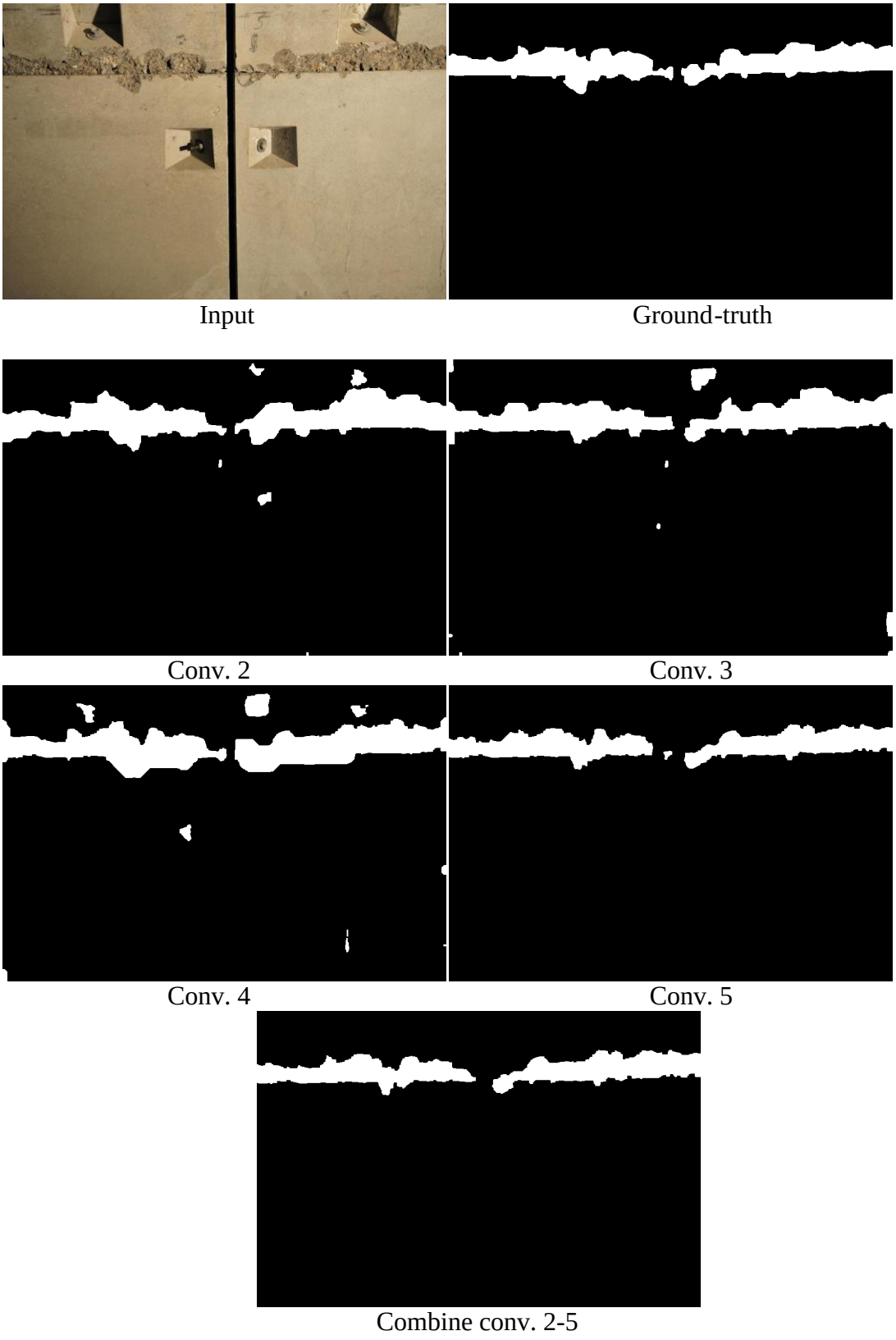


Figure 4: Defect detection results by using different convolutional layers of convolutional neural networks

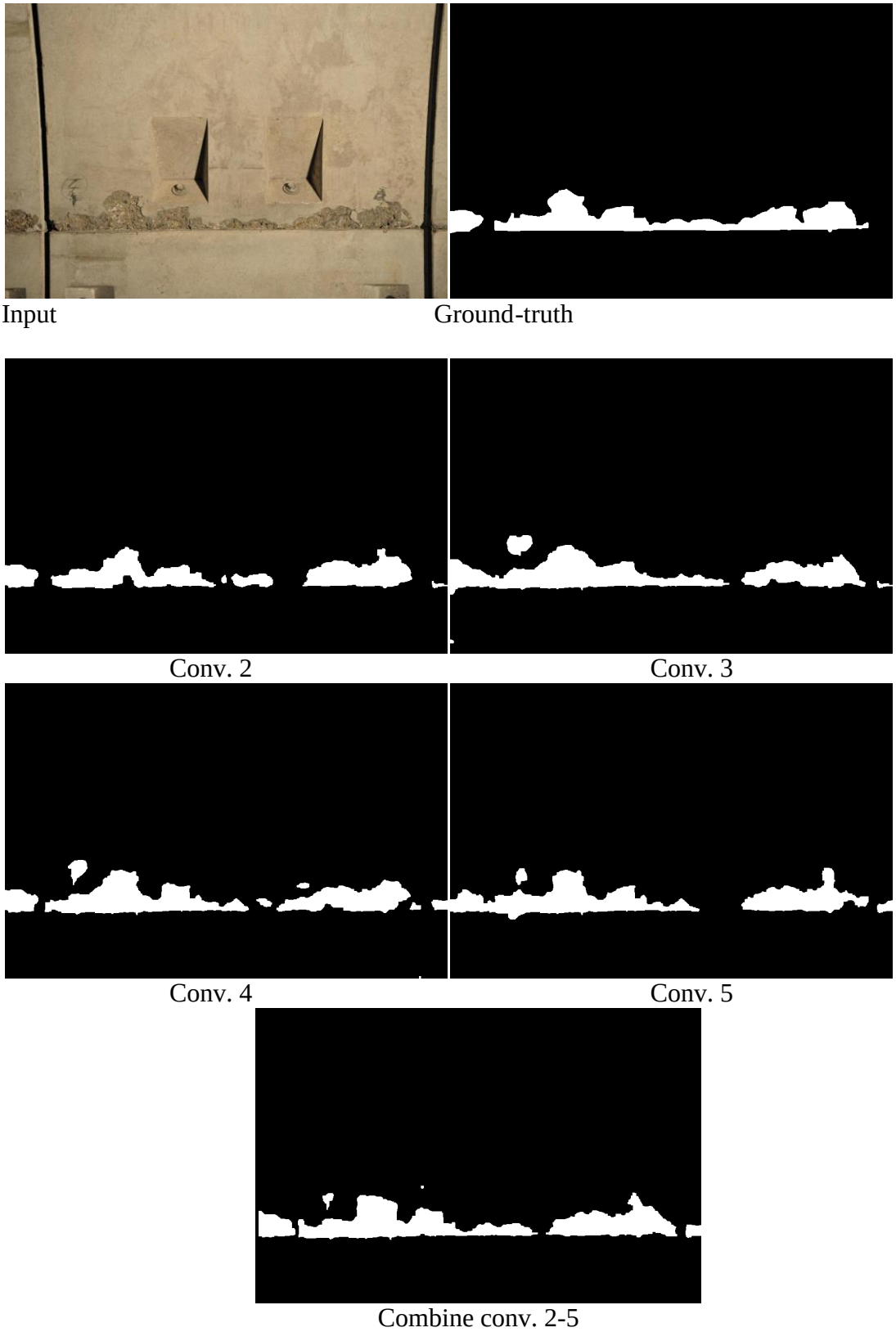


Figure 5: Defect detection results by using different convolutional layers of convolutional neural networks

2.5.2.2 Validation of graph-cut based defect segmentation

To see if the proposed graph-cut (GC) based defect detection is able to improve defect detection performance, we also generate binary defect detection results by thresholding the probabilistic classification map from the final layer of CNN deep learning networks.

Figure 6 presents the precision, recall and F-measure of the proposed approach by using or without using graph-cut for defection segmentation. Thanks to the graph-cut segmentation, the recall is substantially increased while keeping high precision, thus F-measure is increased from 60% to 72%, which is a significant improvement.

Figure 7 and Figure 8 show some defect detection results by using or without using graph-cut for defect segmentation. We can observe that the graph-cut segmentation is able remove noisy prediction from defect classification, while preserving defect contours.

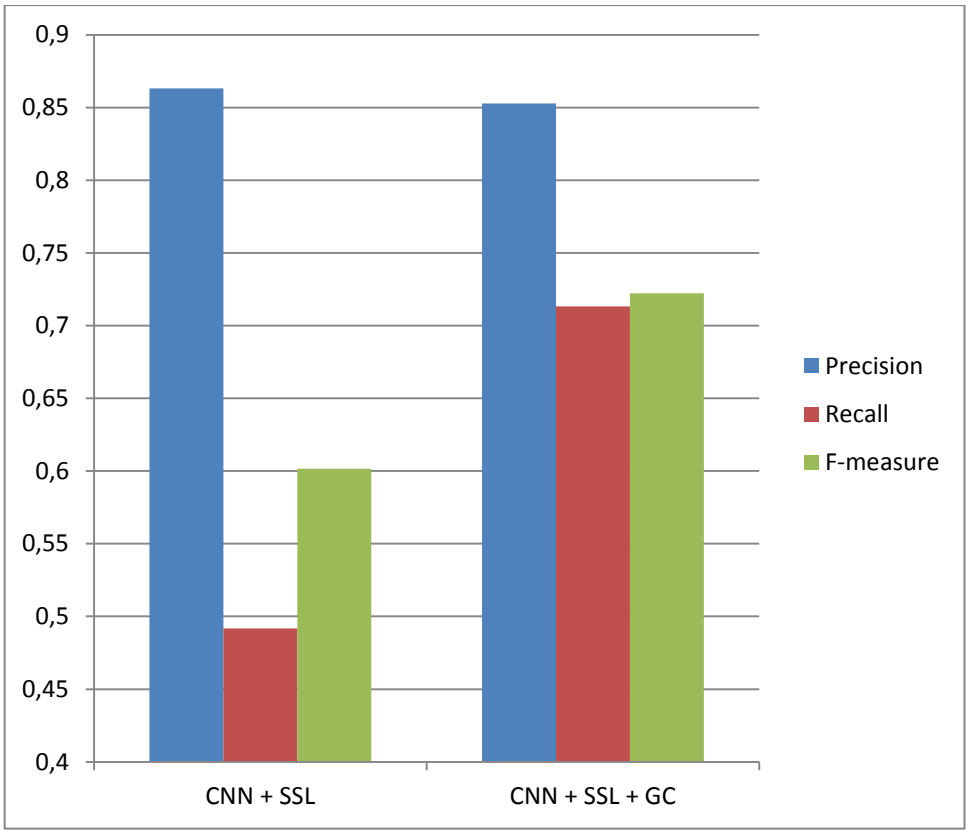


Figure 6: Validation of graph-cut (GC) based defect segmentation

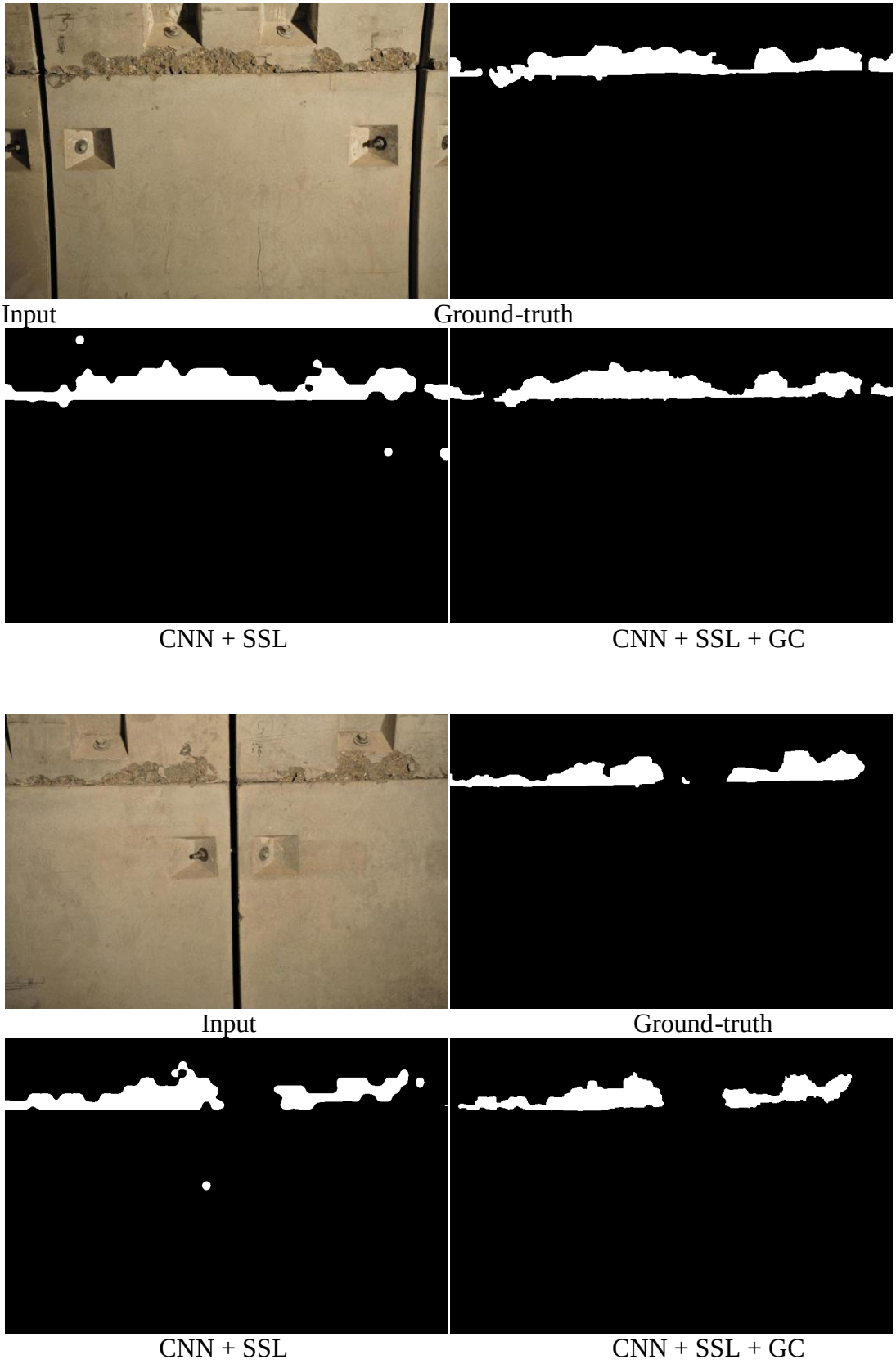


Figure 7: Defect detection results by using or without uisng graph-cut for defect segmentation

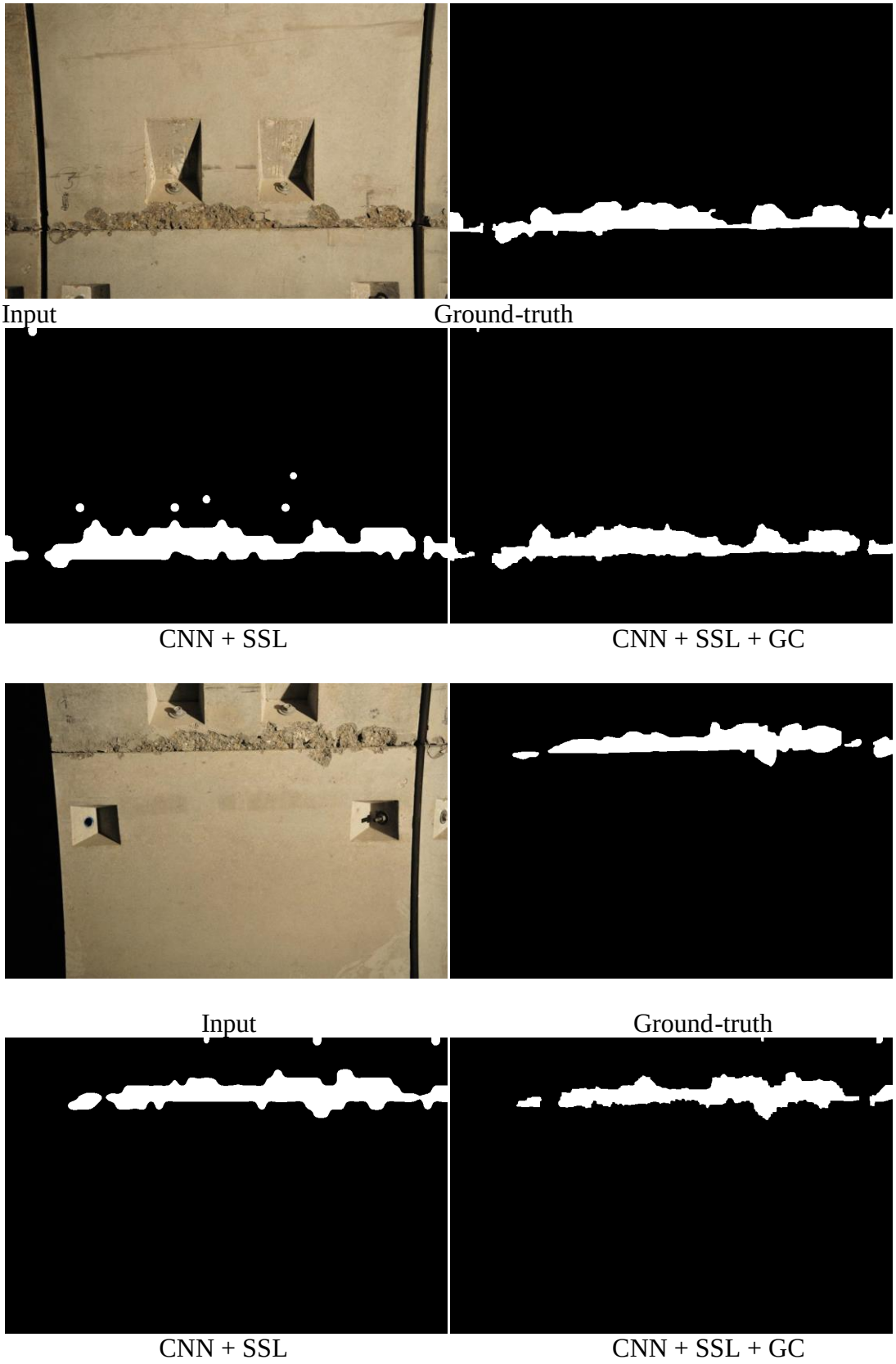


Figure 8: Defect detection results by using and without uisng graph-cut for defect segmentation

2.5.2.3 Validation of semi-supervised learning

To validate the semi-supervised learning (SSL) method, which employs both labled and unlabeled data to train defect detection model, we compare its perfomrance to the traditional learning approach which uses labled data only.

Figure 9 shows the precision, recall and F-measure obtained by the traditional learning approach and by semi-supervised learning. We can see that the semi-supervised learning achieves objviously higher performance. Quantitatively, it obtains 5% improvement in terms of F-measure.

Figure 10 and Figure 11 show some defect detection results generated by using the traditional learning approach and by the semi-supervised learning method, respecatively. We can obsver that the semi-supervised learning method can detect defects more complitely compared to the traditional one.

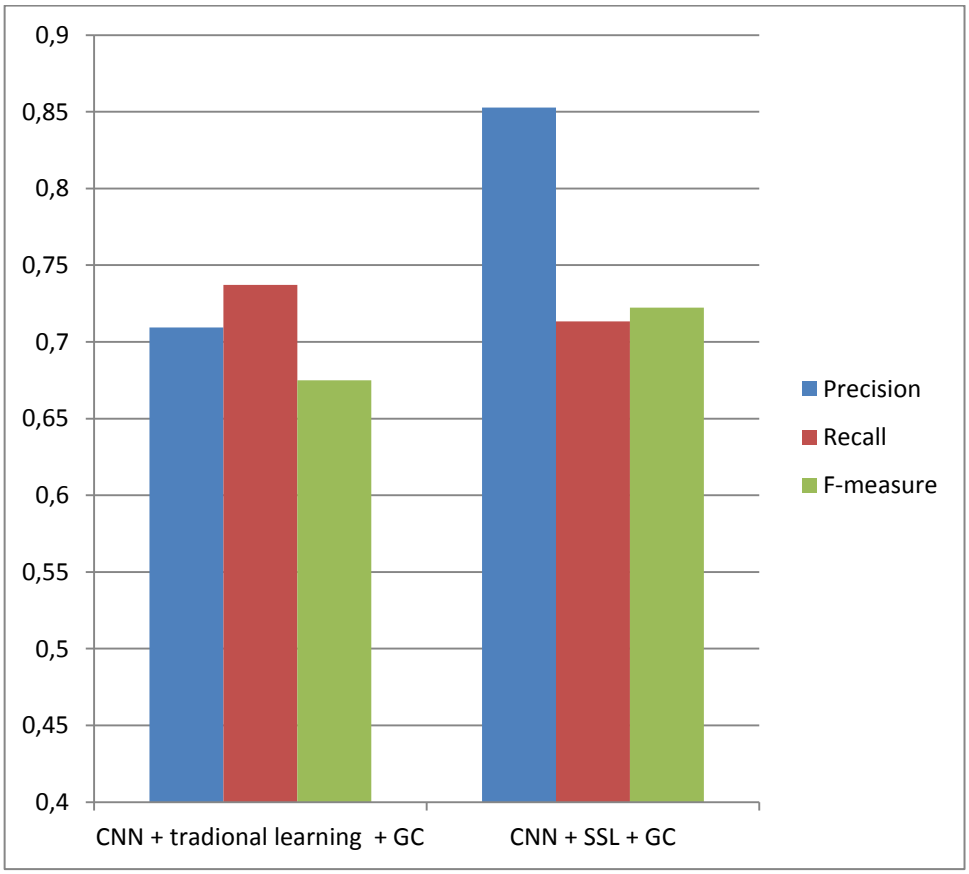


Figure 9: Precision, recall and F-measure by using semi-supervised learning (SSL) and by traditional learning methods, respectively

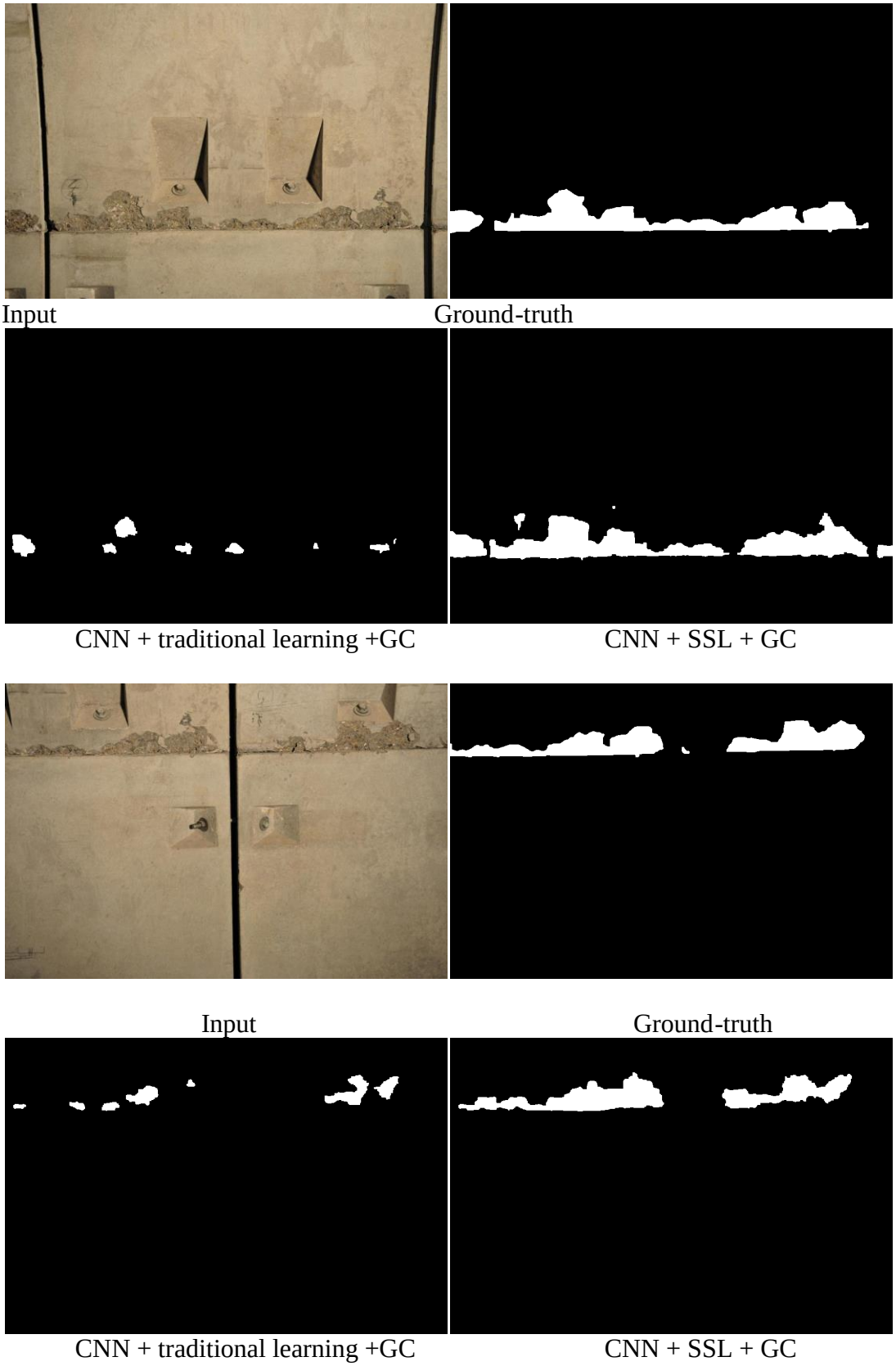


Figure 10: Defect detection results by traditional learning and by semi-supervised learning

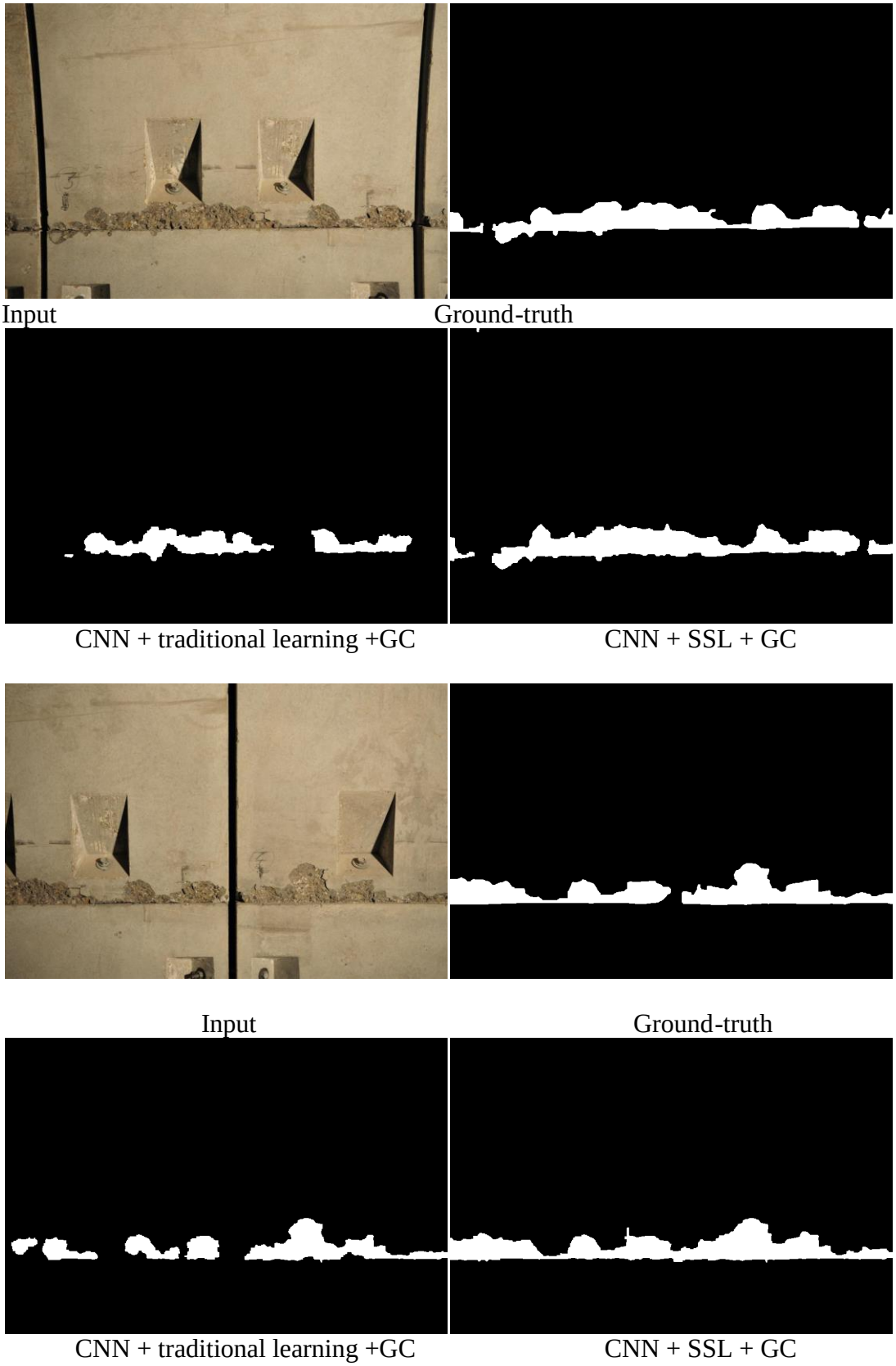


Figure 11: Defect detection results by traditional learning and by semi-supervised learning

2.5.2.4 Testing on non-defect images

To validate the robustness of the proposed defect detection algorithm, we also apply it to those images which do not contain specific (spalling)defects. Figure 12 and Figure 13 show the detection results on these images. Obviously, the proposed algorithm mostly does not generate error prediction when it is applied to relatively simple images (like the first image in Figure 12)or more comlex ones (the second image in Figure 12 and all images in Figure 13). Therefore, the proposed algorithm is robust.

Figure 12: Defect detection results on images that do not contain spalling defect.

Input Ground-truth

Defect detection result

Input Ground-truth

Defect detection result

Figure 13: Defect detection results on images that do not contain spalling defect.

3. CRACK DETECTION

3.1 Visual Inspection of Concrete Surfaces

Visual Inspection (VI) is widely used to determine the safety of concrete structures. VI involves the measurement of defects either manually or in an automatic way. Manual inspection requires significant human effort by involving inspectors walking along the surface of the structure while using only their naked eye. Therefore, a rapid and complete survey cannot be ensured, as it yields subjective results prone to human errors. Automated approaches to the problem of VI consist a challenging alternative to overcome the aforementioned problem.

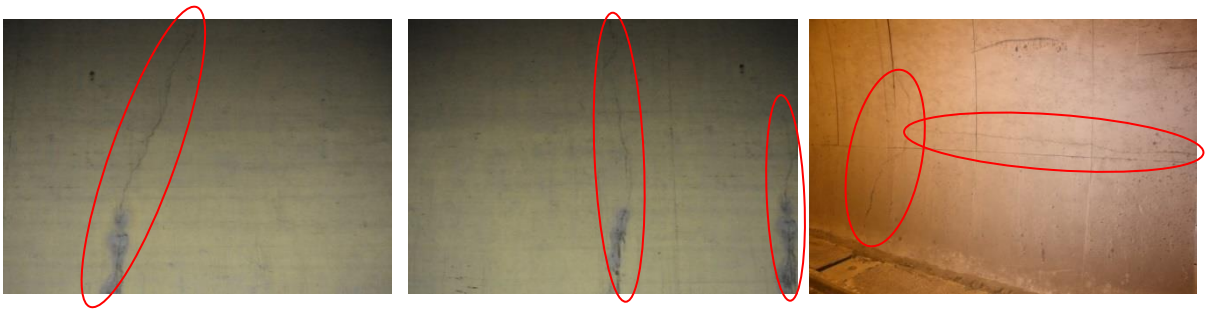
Approaches that utilize automated procedures for VI of concrete infrastructures aim specifically to defects detection by providing objective data to be used for structure evaluation. Towards this direction, such methods exploit image processing and machine learning techniques. Initially, low-level image features are used towards the construction of complex handcrafted features, which in turn are used to train learning models; i.e. the detection methods. Automated approaches have been applied in practical settings including roads, bridges, fatigues, and sewer-pipes[192][193][194][195][196].

Most of the approaches that use automated procedures for VI rely on the construction of handcrafted features, which in turn are used by classifiers in order to evaluate concrete infrastructures. Towards this direction Liu et al. in[197] utilize image intensity features and the application of Support Vector Machine (SVM) algorithm to detect cracks on tunnel surfaces. Image color properties are also investigated in[198]. Different non-RGB color spaces and various machine learning algorithms, such as Gaussian Mixture Model (GMM), Artificial Neural Network (ANN) and SVM, were evaluated in order to find the optimal combination in terms of detection accuracy.

Abdel-Qaderet al. in[195] evaluate the effectiveness of four different edge detection techniques on detecting concrete cracks. Edge detection algorithms are also utilized by Yu et al. in[199]; the Sobel and Laplacian operators are applied on raw image data to extract crack information, which is used to measure cracks through the exploitation of a graph based search algorithm. Mohanty and Wang in[200] are further extending these works by proposing an image mosaic technology for detecting tunnels surface defects which are not identifiable though edge detection.

In the following, we present our approach towards crack detection on tunnels' surface. Our method relies exclusively on image processing techniques by exploiting image intensity and shape characteristics.

Figure 14: Examples of Cracks.



3.2 Crack Detection Approach

In this section, we describe our approach towards the detection of cracks on tunnels' surface. Cracks are the most common defect type. They appear in concrete for many reasons; usually as secondary symptoms of other defects, such as a long rounded crack following the structural failure of a warped slab, caused by volume changes and structural failure. As such, the identification of a crack should be the first step, prior to the extensive analysis in the surrounding area. Additionally, given the magnitude of the investigated area, crack identification should be as fast as possible, once we have positively identify a crack we can proceed with a further advanced techniques.

A crucial step towards cracks detection, is the definition of cracks characteristics. Cracks can be characterized by their shape and intensity, examples of cracks on tunnels' surface are presented in Figure 14: Examples of Cracks.. Cracks is expected, on the one hand to present large length and small width, and on the other, to be "not straight" lines. Furthermore, pixels that belong to cracks are expected to be darker than their neighbouring pixels that do not belong to cracks. In other words, shape properties describe the ratio length to width for the detected edges, while intensity properties describe the spatial relationship of cracks to their surroundings.

Thus, based on cracks characteristics, the crack detection problem can be addressed through two different approaches; i) through approaches that are based on image processing techniques and ii) through approaches that are based on machine learning techniques.

Image processing techniques exploit cracks' characteristics, regarding pixels intensity and their spatial relations, through the utilization of morphological operations, kernel filtering and simple shape analysis based on the size of detected areas and their "sphericity". The utilization of image processing techniques can be though as an unsupervised crack detection approach. It requires no annotated data, while at the same time and all image processing operations can be parallelized in order to achieve better than real time performance (25fps). However, a crack detection system that is based on image processing presents low generalization ability due to the fact that applied operations are fine tuned for a specific dataset.

On the other hand approaches based on machine learning techniques can exploit cracks' intensity and shape characteristics by utilizing low level image features for the detection process (conventional pattern recognition paradigm) or to hierarchically construct high level features, which in turn will be used for detection purposes (deep learning paradigm). Neural Networks, Markov Random Fields and Support Vector Machines are some well-known examples of learning machines. Machine learning approaches present high generalization abilities and although their training may be time consuming, they can achieve real time predictions. However, the utilization of machine learning techniques can be though as a supervised crack detection approach. Machine learning techniques require data annotation, which is a tedious and time consuming task.

The developed crack detection system is based exclusively on image processing techniques. Concretely, our approach is based on the following steps;

1. Line enhancing
2. Noise removal
3. Straight lines removal
4. Shape filtering
5. Morphological reconstruction

Line enhancing exploits intensity characteristics of cracks. The intensity of pixels that are darker than the average intensity of their neighbours is set to zero. The number of neighbours of a pixel located at (x, y) position on image plane is defined by the size of a square window centred at the same position. Although that line enhancement can emphasize region that may contain a crack, it produces "salt and pepper" noise. For this reason, line enhancement is followed by a noise removal step that exploits a median filter. Furthermore, during the noise removal step, the image is converted to binary by using a thresholding operation. Then, areas that are smaller than a pre-specified threshold are removed. This step is based on the detection of connected components on the binary image plane.

The noise removal step is followed by a straight line removal step. Straight lines is something common and usually correspond to man-made crafts (e.g. wiring). Straight lines are located by using the probabilistic Hough transform (faster and more computational effective than conventional Hough transform). Straight line removal is based of shape characteristics of cracks.

Shape filtering using appropriate image moments is another crucial step towards crack detection. Cracks is expected to have the form of curves. Thus, by locating the minimum enclosing circles of connected components we are able to exclude candidate cracks, whose shape is not like a curve. Finally, we perform a classical morphological operation called "opening by reconstruction". Opening by reconstruction starts from a set of starting points (seeds) and then grown in flood-fill fashion ton include complete connected components.

3.3 Crack Detection Results

All algorithms are applied on raw image data and their performance is evaluated in regard to ground truth data.

3.3.1 Available datasets

All the images were collected in the framework of ROBO-SPECT European FP7 project. They originate from the Metsovo motorway tunnel in Greece, which is a 3,5km long twin tunnel. Construction of the span north bore was completed in 1994. In a distance of 20m parallel and north to this bore, runs the ventilation tunnel. Although, the last 526m of the ventilation tunnel have been lined with a thick cast in place unreinforced concrete lining, no waterproofing system has been constructed between the temporary support and the final lining. Due to this fact the main tunnel suffered a significant deformation in a Serpentine under an overburden of almost 200m with high ground water inflow.

Image data were captured at this part of the tunnel, using a hand held DSLR camera. During image acquisition no special setup took place, i.e. images are taken from any angle and distance from the tunnel surface. Regions depicting defects, for each one of the captured images, were manually annotated in order to create the ground truth dataset. All algorithm are applied on the raw image data and their performance is evaluated in regard to the ground truth data.

Although, captured images depict various types of defects, the following system performance evaluation focuses exclusively on the detection of cracks. Depicted cracks have arbitrary length (small, medium and long cracks) and arbitrary orientation. Furthermore, each image may contain zero, one or more cracks that have to be detected.

3.3.2 Performance Evaluation

Line enhancement is performed on 13×13 windows by thresholding the 99% of the mean intensity value. The output of the application of line enhancement step on the raw image data is presented in Figure 15: Line Enhancement Results.. As we can see the line enhancement step emphasize regions that may contain cracks, while at the same time it suppresses smooth areas, that is not probable to contain cracks. The line enhanced image is converted to binary by using a thresholding operation based on the intensity of the pixels.

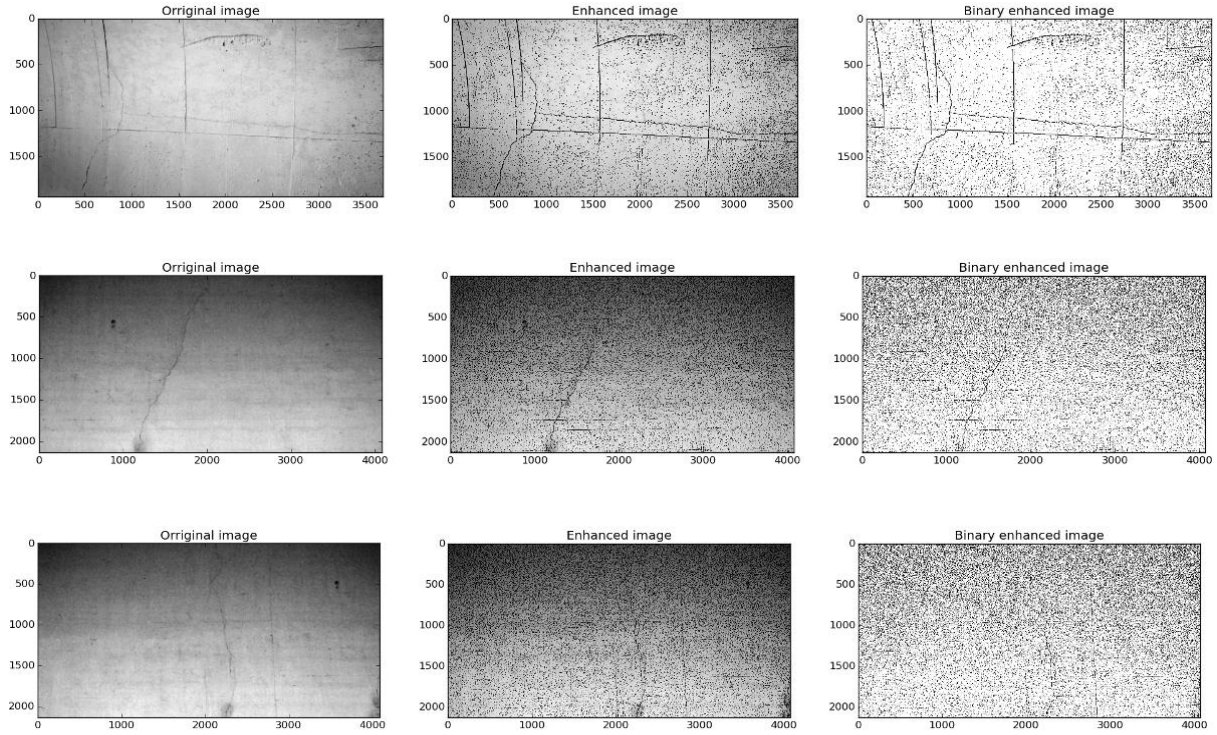
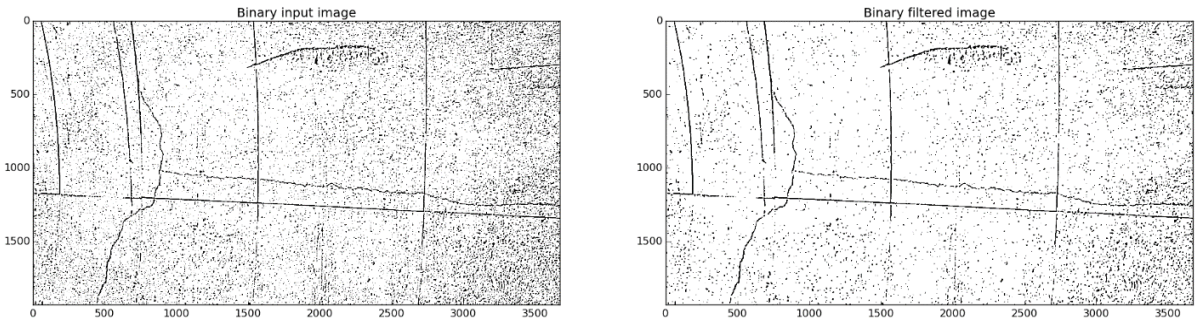


Figure 15: Line Enhancement Results.

Although, that line enhancement operation can emphasize image regions that may contain a crack, it is obvious that it produces “salt and pepper” noise. For this reason this operation is followed by a noise removal step that exploits a median filter. Figure 16: Noise removal results using a median filter.. presents the output of the noise removal step. Depending on the visual content of the image, this step can successfully suppress non crack regions.



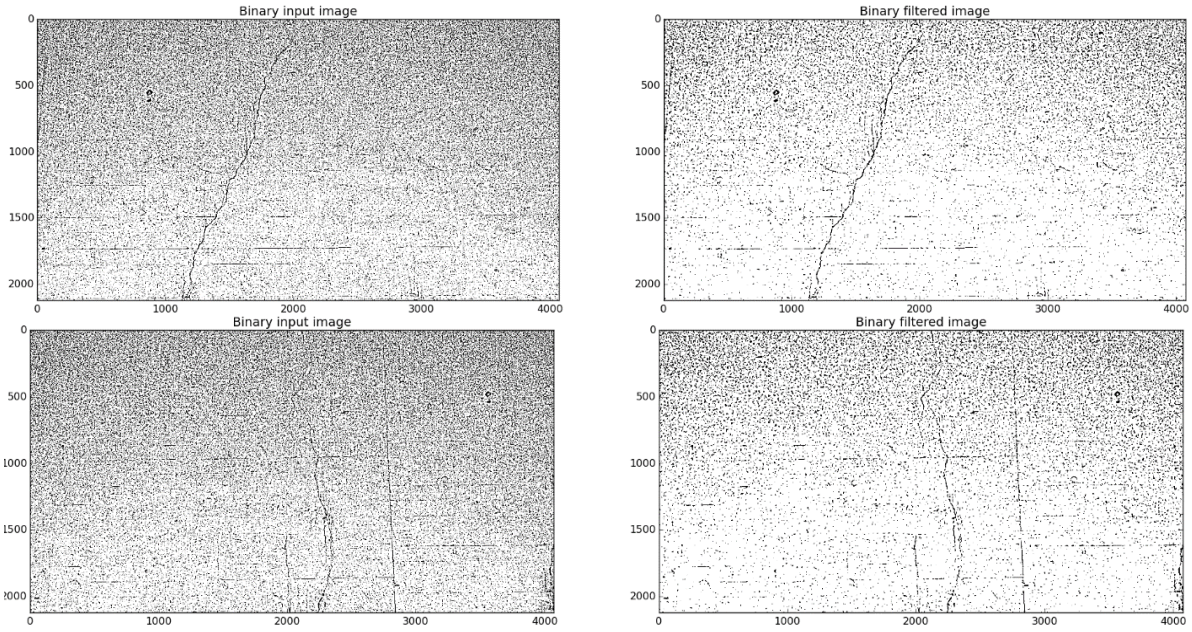
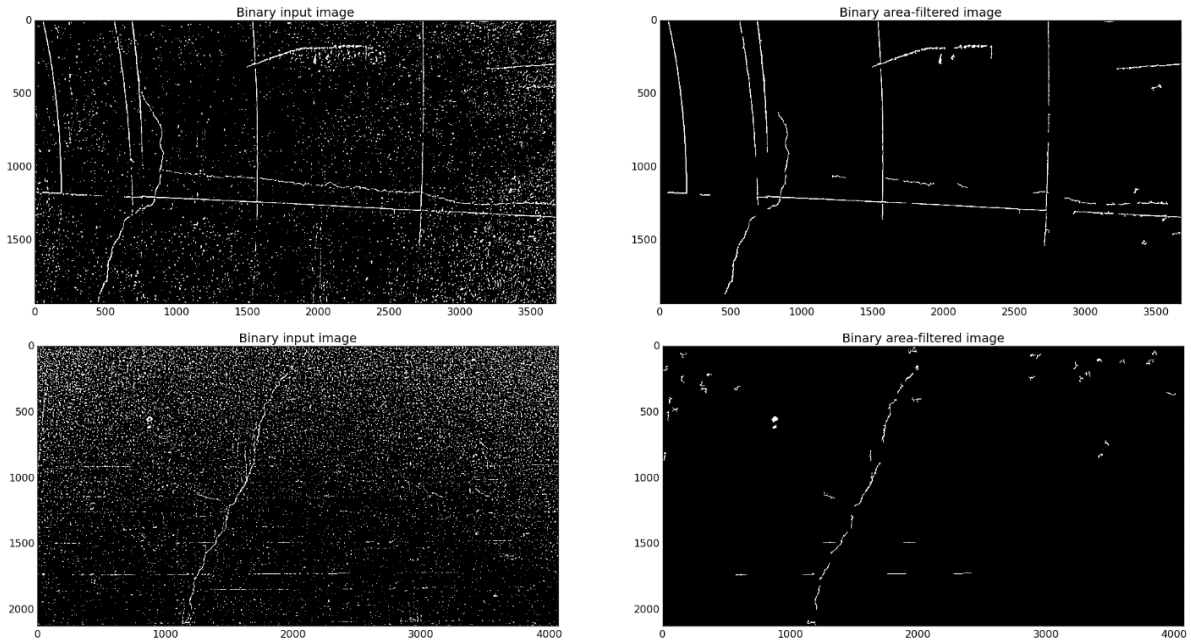


Figure 16: Noise removal results using a median filter.

The filtering using a median filter is followed by a filtering based on the area of candidate cracks. During these filtering operation small sized candidate cracks are removed. For completing this step initially, the connected components are detected and then they are filtered using a pre-defined area threshold. This threshold is depended on the resolution of the raw image. For this dataset we excluded connected components spanning areas less than 550 pixels. The output of the area filtering operation is presented in Figure 17: Area filtering results..



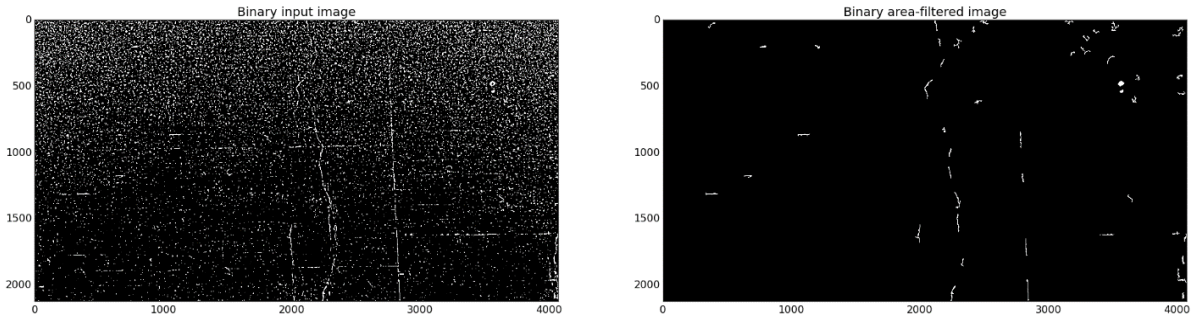


Figure 17: Area filtering results.

Having exclude the vast amount of areas that do not contain a crack, the algorithm proceeds by removing straight lines, which is very probable to depict man-made crafts. For detecting straight lines we used the probabilistic Hough transform, with distance and angle resolution parameters to equal 5 pixels and 0 radians respectively. The output of the straight line removal process is presented in Figure 18: Straight line removal output..

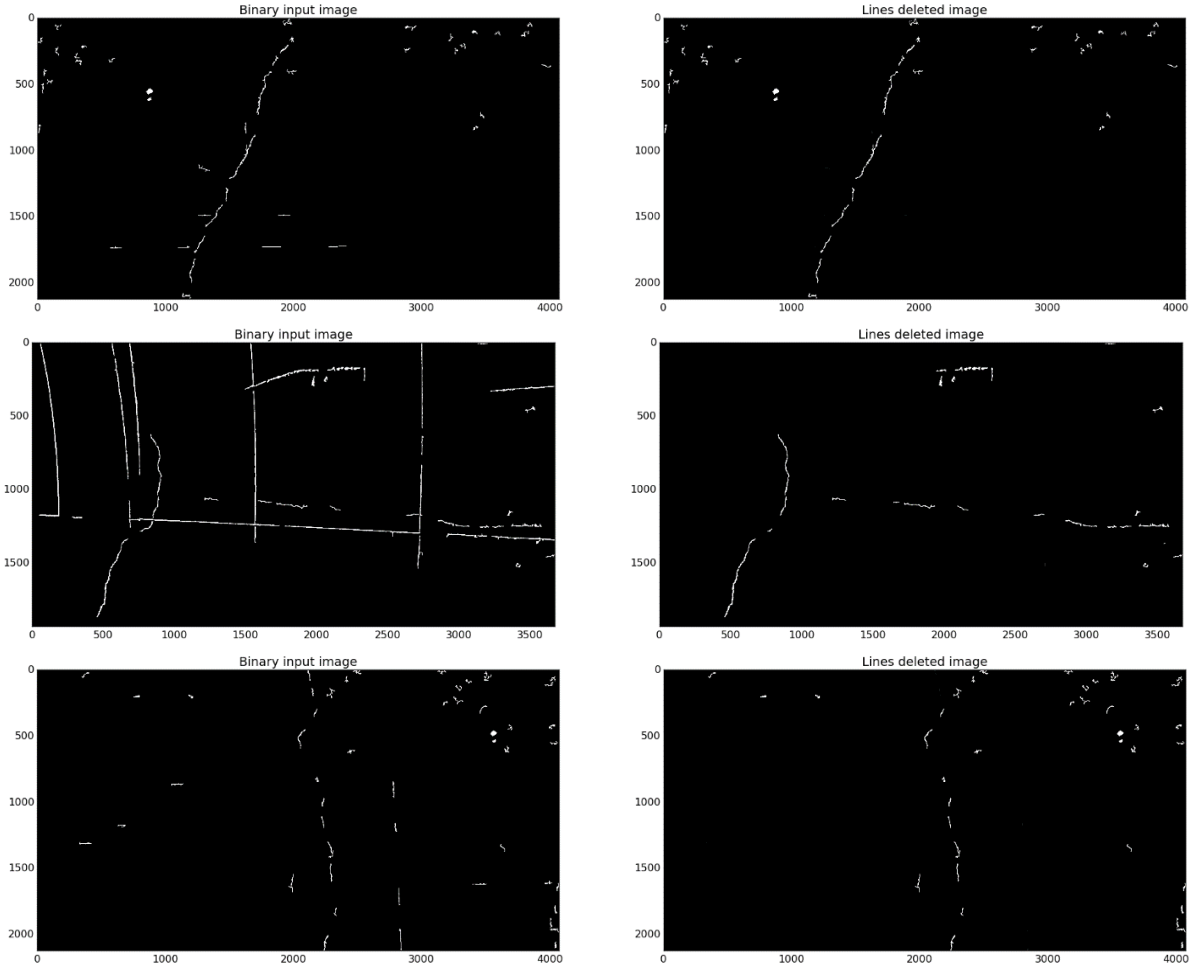


Figure 18: Straight line removal output.

The straight line removal is followed by a shape filtering operation based on image moments. Candidate cracks that span area more than 30% of the minimum enclosing circle area are discarded. The output of this operation is presented in Figure 19: Sphericity filtering results.. Finally, the output of shape filtering is used as starting point for applying the opening by reconstruction morphological operation. The output of the opening by reconstruction, which is the final step towards cracks detection, are presented in Figure 20: Final Crack detection results..

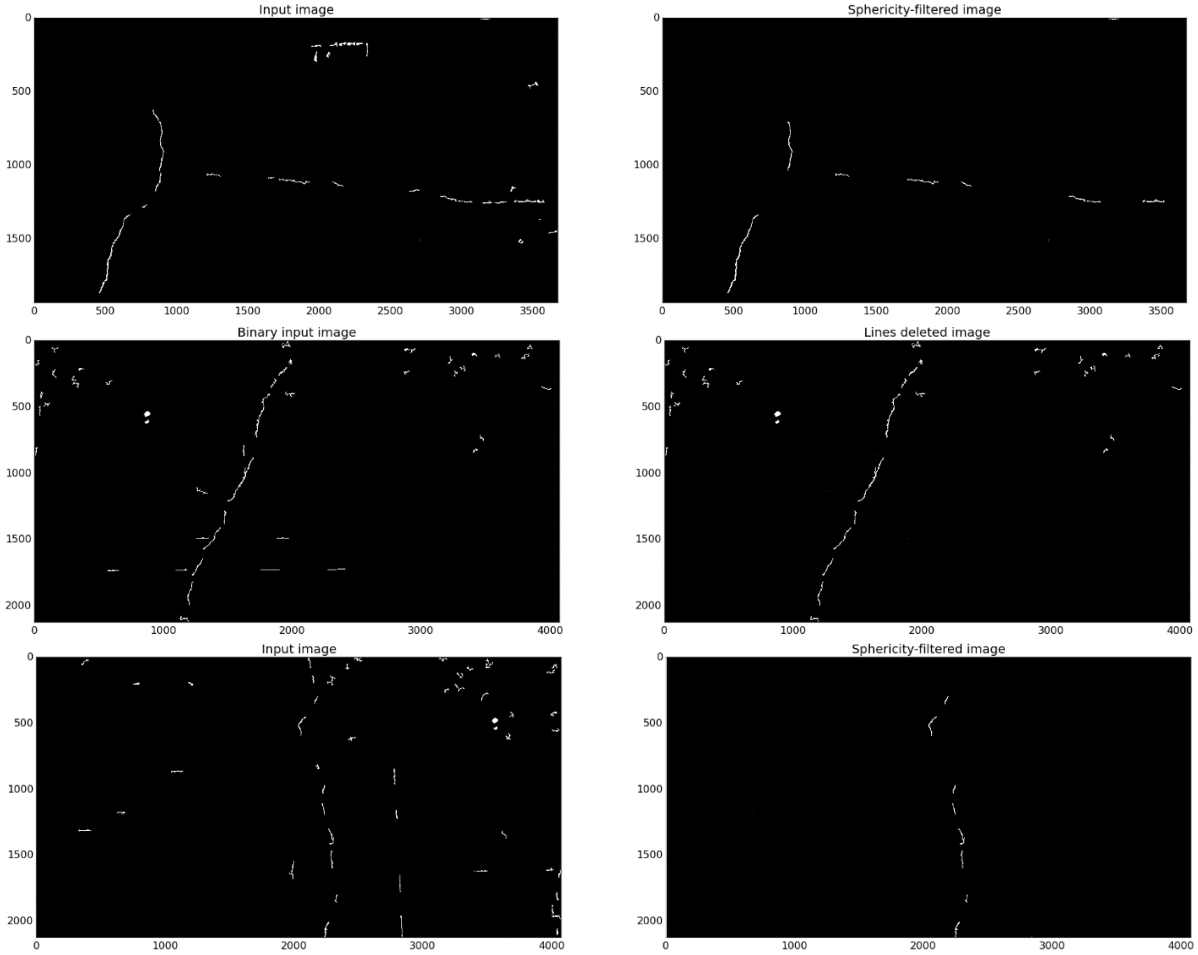
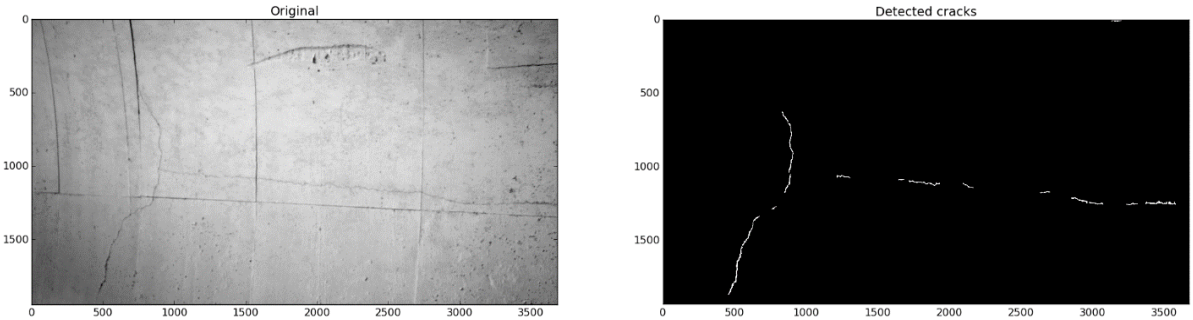


Figure 19: Sphericity filtering results.



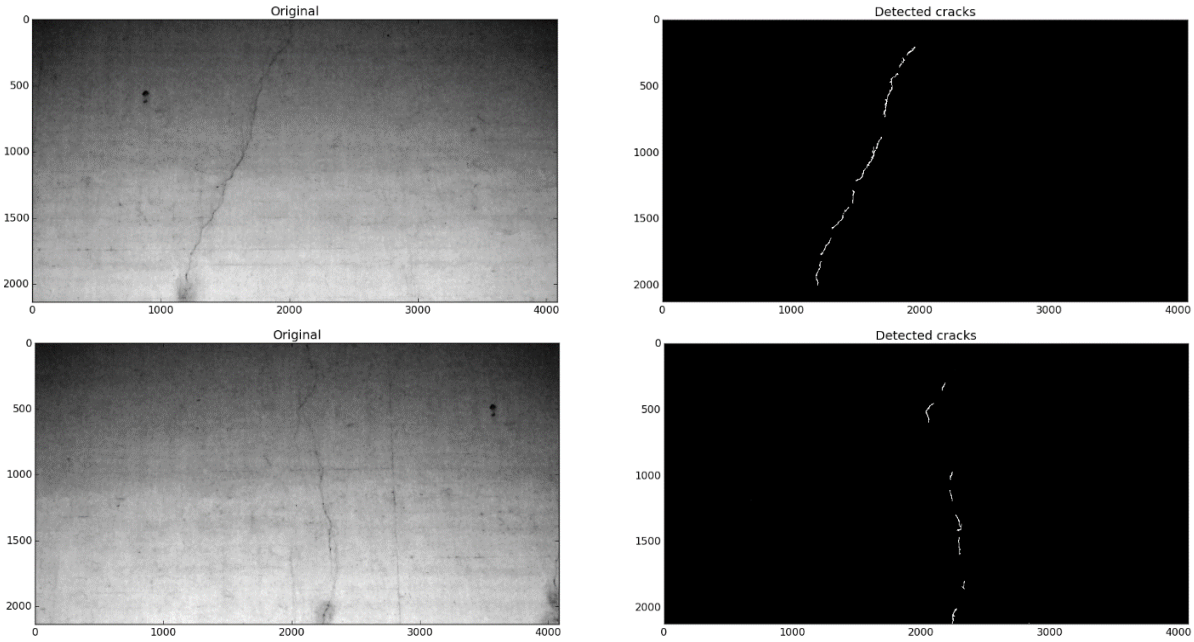


Figure 20: Final Crack detection results.

4. FINAL 3D MEASUREMENTS

The contribution of 3D vision to the overall tunnel inspection system is as follows:i) XYZ coordinates of the crack; ii) a local 3D reconstruction for the structural assessment tool (visual inspection); iii) the estimation of the orientation and position of the crack; iv) estimation of the crack length for choosing whether to measure a crack with ultra-sonic sensors, or not; v) a perpendicular section to the tunnel lining for the estimation of tunnel deformations.

During the Work Package 3 activities on the extraction of 3D information from visual data we have come up with an adaptive algorithm, which is based on a hierarchical matching scheme. The matching strategy is local, hence the requirements of speed and computational efficiency are met, while in the same time the requested accuracy and level of detail are served. Our extensive evaluation of different configurations of matching algorithms (cost functions and optimization methods) has resulted in an extension of the cross-based support regions aggregation method and a novel *Modified Census Transformation* for image matching. These allow forming highly adaptive irregular support regions; it is independent of the cost function; and it degenerates from area to single dimensional aggregation in order to maximize computational efficiency. The novel *Modified Census Transformation* enhances the ability of the original Census Transformation to deal with severe radiometric changes in the stereo pair, something which is common in our real-life datasets.

In Deliverable D3.3, which is dedicated to the contribution of 3D vision to the ROBO-SPECT project, the stereo matching algorithm for dense 3D full reconstruction is presented. In this work several cost functions and aggregation methods have been evaluated and novel aspects have been introduced in the process, in order to improve the results under the specific needs of ROBO-SPECT image data. The evaluation was performed on datasets provided on online evaluation platforms, because for these datasets ground truth are given. These datasets are briefly described and analysed in D3.3. The final results are of course tested and projected on the real images gathered in the ROBO-SPECT tunnels that have been allocated from the consortium partners at the testing sites of the ROBO-SPECT project. The tuning process of the two matching algorithms, which were selected as the final candidates for optimization and evaluation under the ROBO-SPECT data is described. Moreover D3.3 presents elaborate visual and numerical results of each step of the 3D reconstruction process and some relevant comments. The use of the terrestrial 3D laser scanner on the robotic vehicle is also described and the overall system integration with respect to the computer vision module is presented. Finally, the D3.3 report concludes with some remarks on the above developments.

5. USER INTERFACE AND INTEGRATION

User Interface and Integration is in a sense the most important component of the entire project. The ultimate goal of the project is to have a well rounded integrated system which has effective and quick communications among the various sub-systems and at the same time provides the end user with the a handy-to-use interface through which he can monitor the entire working of the system just by sitting in the ground control station.

Defect detection software is built on an already running PC that controls the operation of the crane, as both of them require a Linux operating system. The Crack detection software is running on another PC having a windows Operating system which is used for controlling the Robotic arm situated on the tip of the crane.

In ROBO-SPECT project, we have integrated the cameras used to detect the cracks with the robotic platform. This is performed using the YARP communication protocol between the cameras and the robot.
We have used, in total, three programs. The programs are one sender and two receivers, simple_sender.cpp (denoted as (SiSe), simple_receiver_1.cpp (denoted as SiRe1) and simple_receiver_2.cpp (denoted as SiRe1). All programs were written in C++. Especially for simple_receiver_2.cpp, code written in Python had to be developed. The corresponding YARP executables were created using CMake (an open source software).
The entire set up simulates the image acquisition and processing tasks. In particular, SiSe provides the information of whether a new image is available or not, by sending an appropriate message. The message can contain various information (including image name). That message will be received by the SiRe1 and the image processing shall start. While analyzing the image SiSe is set on standby mode.
SiRe1 is responsible for the crack detection. During image processing, SiRe1 provides information regarding the current detection stage. Once the analyzation of the image is concluded, SiRe1 transmits multiple messages regarding the x, y coordinates of the detected crack medians. These messages will be received by SiRe2 and will be further used for the depth identification of the cracks. Also, Sire2, reenacts the SiSe, simulating the continuation of the image acquisition process
The execution of such simulation requires the execution, in the background, of the YARP server. Additionally, we will need three more consoles (assuming Windows operating system), one for each sender/receiver. The message flow occurs among these consoles.

The images continuously been captured from the defect detection camera's (situated on the crane tip) and stored in a common shared directory access able to both the PC's. The defect detection algorithm runs offline and stores images in the shared directory with the corresponding defect detection results for each image.

The end user sitting at the ground control station which has access to this shared directory, can continuously monitor the progress of the system inside the tunnel. Thus we have a well integrated system which can effectively communicated and pass messages between various subsystems at the same time allowing continuous interaction for the end user.

6. CONCLUSIONS

In this work, we have finalized upon a well experimented pipeline for defect detection in tunnels. For the defect detection part, we have deep learning models for computing robust features in this complex task of detecting defects in tunnels which show much improved performance as compared to simple hand engineered features. We further include semi-supervised techniques to counter the scarcity of labelled dataset and effectively use the enormous unlabelled dataset for training. An optional salient object detection technique has also been suggested which can greatly enhance the results in high resolution images. Finally the computed segmentation results are further enhanced by adding to the existing pipeline graphical models. Experiments with the dataset collected from the VSH tunnel have shown excellent results proving the robustness and feasibility of this approach.It has to be mentioned that the parameters of the techniques applied at each stage of the crack detection pipeline are the same for all images.

In the future we plan to run the defect algorithm on low contrast images collected from live tunnel scenario with actual lighting conditions and camera's from the robot itself and check for the efficacy of the entire pipeline through testing and validation.

7. REFERENCES

[1] Deng, L.: An overview of deep-structured learning for information processing, in Proc. Asian-Pacific Signal & Information Processing
[2] Annu. Summit and Conf. (APSIPA-ASC), October 2011.
[3] Deng, L.: Expanding the scope of signal processing. IEEE Signal Process. Mag., 25 (3) (2008), 2–4. [3] Hinton, G.; Osindero, S.; Teh, Y.: A fast learning algorithm for deep belief nets. Neural Comput., 18 (2006), 1527–1554.
[4] Bengio, Y.: Learning deep architectures for AI. Found. TrendsMach. Learn., 2 (1) (2009), 1–127.
[5] Bengio, Y.; Courville, A.; Vincent, P.: Representation learning: a review and new perspectives, IEEETrans.PatternAnal.Mach. Intell.,35 (2013), 1798–1828.
[6] Hinton, G. et al.: Deep neural networks for acoustic modeling in speech recognition. IEEE Signal Process.Mag., 29 (6) (2012), 82–97.
[7] Yu, D.; Deng, L.: Deep learning and its applications to signal and information processing. IEEE Signal Process. Mag., 28 (2011), 145–154.
[8] Arel, I.; Rose, C.; Karnowski, T.: Deep machine learning – a new frontier in artificial intelligence, in IEEE Computational IntelligenceMag., 5 (2010), 13–18.
[9] Markoff, J.: Scientists SeePromise inDeep-LearningPrograms. New York Times, November 24, 2012.
[10] Cho, Y.; Saul, L.: Kernel methods for deep learning. NIPS, 2009, 342–350.
[11] Deng, L.; Tur, G., He, X.; Hakkani-Tur, D.: Use of kernel deep convex networks and end-to-end learning for spoken language understanding, in Proc. IEEEWorkshop on Spoken Language Technologies, December 2012.
[12] Vinyals, O.; Jia, Y.; Deng, L.; Darrell, T.: Learning with recursive perceptual representations, in Proc. NIPS, 2012.
[13] Baker, J. et al.: Research developments and directions in speech recognition and understanding. IEEE Signal Process. Mag., 26 (3)(2009), 75–80.
[14] Baker, J. et al.: Updated MINS report on speech recognition and understanding. IEEE Signal. Process. Mag., 26 (4) (2009), 78–85.
[15] Deng, L.: Computational models for speech production, in Computational Models of Speech Pattern Processing, 199–213, Springer-Verlag, 1999, Berlin, Heidelberg.
[16] Deng, L.: Switching dynamic system models for speech articulation and acoustics, in Mathematical Foundations of Speech andLanguage Processing, 115–134, Springer, NewYork, 2003.
[17] George, D.: How the Brain Might Work: A Hierarchical and Temporal Model for Learning and Recognition. Ph.D. thesis, Stanford University, 2008.
[18] Bouvrie, J.: Hierarchical Learning: Theory with Applications in Speech and Vision. Ph.D. thesis, MIT, 2009.
[19] Poggio, T.: How the brain might work: the role of information and learning in understanding and replicating intelligence, in Information: Science and Technology for the New Century (G. Jacovitt, A. Pettorossi, R. Consolo, V. Senni, eds), 45–61, Lateran UniversityPress, 2007, Amsterdam, Netherlands.
[20] Glorot, X.; Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks, in Proc. AISTAT, 2010.
[21] Hinton, G.; Salakhutdinov, R.: Reducing the dimensionality of data with neural networks. Science, 313 (5786) (2006), 504–507.
[22] Dahl, G.; Yu, D.; Deng, L.; Acero, A.: Context-dependent DBNHMMs in large vocabulary continuous speech recognition, in Proc.ICASSP, 2011.
[23] Mohamed, A.; Yu, D.; Deng, L.: Investigation of full-sequence training of deep belief networks for speech recognition, in Proc. Interspeech, September 2010.
[24] Mohamed, A.; Dahl, G.; Hinton, G.: Acoustic modeling using deep belief networks. IEEE Trans. Audio Speech Lang. Process., 20 (1)(2012), 14–22.
[25] Dahl, G.; Yu, D.; Deng, L.; Acero, A.: Context-dependent DBNHMMs in large vocabulary continuous speech recognition. IEEETrans. Audio Speech, Lang. Process., 20 (1) (2012), 30–42.
[26] Mohamed, A.;Hinton, G.; Penn, G.: Understanding how deep belief networks perform acoustic modelling, in Proc. ICASSP, 2012.

[27] Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; Manzagol, P.: Stacked denoising autoencoders: leaning useful representations in adeep network with a local denoising criterion. J. Mach. Learn. Res., 11 (2010), 3371–3408.
[28] Rifai, S.; Vincent, P.; Muller, X.; Glorot, X.; Bengio, Y.: Contractive autoencoders: explicit invariance during feature extraction, in Proc.ICML, 2011, 833–840.
[29] Ranzato, M.; Boureau, Y.; LeCun, Y.: Sparse feature learning for deep belief networks, in Proc. NIPS, 2007.
[30] Deng, L.; Seltzer, M.; Yu, D.; Acero, A.; Mohamed, A.; Hinton, G.: Binary coding of speech spectrograms using a deep auto-encoder,in Proc. Interspeech, 2010.
[31] Bengio, Y.; De Mori, R.; Flammia, G.; Kompe, F.: Global optimization of a neural network – Hidden Markov model hybrid, in Proc.Proc. Eurospeech, 1991.
[32] Bourlard, H.; Morgan, N.: Connectionist Speech Recognition: A Hybrid Approach, Kluwer, Norwell, MA, 1993.
[33] Morgan, N.: Deep and wide: multiple layers in automatic speech recognition. IEEE Trans.Audio Speech, Lang. Process., 20 (1) (2012),7–13.
[34] Deng, L.; Li, X.:Machine learning paradigms in speech recognition: an overview. IEEE Trans. Audio Speech, Lang., 21 (2013), 1060–1089.
[35] LeCun,Y.; Chopra, S.; Ranzato,M.;Huang, F.: Energy-basedmodels in document recognition and computer vision, in Proc. Int. Conf.Document Analysis and Recognition, (ICDAR), 2007.
[36] Ranzato, M.; Poultney, C.;Chopra, S.; LeCun,Y.: Efficient learning of sparse representations with an energy-based model, in Proc. NIPS,2006.
[37] Ngiam, J.; Khosla,A.;Kim,M.; Nam, J.; Lee,H.; Ng,A.:Multimodal deep learning, in Proc. ICML, 2011.
[38] Ngiam, J.; Chen, Z.; Koh, P.; Ng, A.: Learning deep energy models, in Proc. ICML, 2011.
[39] Hinton, G.; Krizhevsky, A.; Wang, S.: Transforming auto-encoders, Proc. Int. Conf. Artificial Neural Networks, 2011.
[40] Salakhutdinov, R.; Hinton, G.: Deep Boltzmann machines, in Proc. AISTATS, 2009.
[41] Salakhutdinov, R.; Hinton, G.: A better way to pretrain deep Boltzmann machines, in Proc. NIPS, 2012.
[42] Srivastava, N.; Salakhutdinov, R.: Multimodal learning with deep Boltzmann machines, in Proc. NIPS, 2012.
[43] Dahl, G.; Ranzato, M.; Mohamed, A.; Hinton, G.: Phone recognition with themean-covariance restricted Boltzmannmachine. Proc.NIPS, 23 (2010), 469–477.
[44] Poon, H.; Domingos, P.: Sum-product networks: a new deep architecture, in Proc. Twenty-Seventh Conf. Uncertainty in Artificial Intelligence, Barcelona, Spain, 2011.
[45] Gens, R.; Domingo, P.: Discriminative learning of sum-product networks. Proc. NIPS, 2012.
[46] Sutskever, I.;Martens, J.; Hinton, G.: Generating text with recurrent neural networks, in Proc. ICML, 2011.
[47] Martens, J.: Deep learning with Hessian-free optimization, in Proc. ICML, 2010.
[48] Martens, J.; Sutskever, I.: Learning recurrent neural networks with Hessian-free optimization, in Proc. ICML, 2011.
[49] Bengio, Y.; Boulanger, N.; Pascanu, R.: Advances in optimizing recurrent networks, in Proc. ICASSP, 2013.
[50] Sutskever, I.: Training Recurrent Neural Networks. Ph.D. thesis, University of Toronto, 2013.
[51] Mikolov, T.; Karafiat, M.; Burget, L.; Cernocky, J.; Khudanpur, S.: Recurrent neural network based language model, in Proc. ICASSP,2010, 1045–1048.
[52] Mesnil, G.; He, X.; Deng, L.; Bengio, Y.: Investigation of recurrentneural-network architectures and learning methods for spoken language understanding, in Proc. Interspeech, 2013.
[53] Deng, L.: DYNAMIC SPEECHMODELS – Theory, Algorithm, and Application, Morgan & Claypool, December 2006.
[54] Deng, L.: A generalized hidden Markov model with stateconditioned trend functions of time for the speech signal. Signal Process., 27 (1) (1992), 65–78.
[55] Deng, L.: A stochastic model of speech incorporating hierarchical nonstationarity. IEEE Trans. Speech Audio Process., 1 (4) (1993), 471–475.
[56] Deng, L.; Aksmanovic, M.; Sun, D.; Wu, J.: Speech recognition using hiddenMarkov models with polynomial regression functionsas nonstationary states. IEEE Trans. Speech Audio Process., 2 (4) (1994), 507–520.
[57] Ostendorf, M.; Digalakis, V.; Kimball, O.: From HMM's to segment models: a unified view of stochastic modeling forspeech recognition. IEEE Trans. Speech Audio Process., 4 (5) (1996), 360–378.
[58] Deng, L.; Sameti,H.: Transitional speech units and their representation by regressiveMarkov states: applications to speech recognition.IEEE Trans. Speech Audio Process., 4 (4) (1996), 301–306.

[59] Deng, L.; Aksmanovic, M.: Speaker-independent phonetic classification using hidden Markov models with state-conditionedmixtures of trend functions. IEEE Trans. Speech Audio Process., 5 (1997), 319–324.
[60] Yu, D.; Deng, L.: Solving nonlinear estimation problems using Splines. IEEE Signal Process. Mag., 26 (4) (2009), 86–90.
[61] Yu, D., Deng, L.; Gong, Y.; Acero, A.: A novel framework and training algorithm for variable-parameter hidden Markov models.IEEE Trans. Audio Speech Lang. Process., 17 (7) (2009), 1348– 1360.
[62] Zen, H.; Nankaku, Y.; Tokuda, K.: Continuous stochastic feature mapping based on trajectory HMMs. IEEE Trans. Audio Speech,Lang. Process., 19 (2) (2011), 417–430.
[63] Zen, H.; Gales, M. J. F.; Nankaku, Y.; Tokuda, K.: Product of experts for statistical parametric speech synthesis. IEEE Trans.Audio Speech, Lang. Process., 20 (3) (2012), 794–805.
[64] Ling, Z.; Richmond,K.; Yamagishi, J.:Articulatory control ofHMMbased parametric speech synthesis using feature-space-switchedmultiple regression. IEEE Trans. Audio Speech Lang. Process., 21 (2013), 207–219.
[65] Ling, Z.; Deng, L.; Yu, D.: Modeling spectral envelopes using restricted Boltzmann machines for statistical parametric speechsynthesis, in ICASSP, 2013, 7825–7829.
[66] Shannon, M.; Zen, H.; Byrne, W.: Autoregressive models for statistical parametric speech synthesis. IEEE Trans. Audio Speech Lang.Process., 21 (3) (2013), 587–597.
[67] Deng, L.; Ramsay, G.; Sun, D.: Production models as a structural basis for automatic speech recognition. Speech Commun., 33 (2–3)(1997), 93–111.
[68] Bridle, J. et al.: An investigation of segmental hidden dynamic models of speech coarticulation for automatic speech recognition. FinalReport for 1998 Workshop on Language Engineering, CLSP, Johns Hopkins, 1998.
[69] Picone, P. et al.: Initial evaluation of hidden dynamic models on conversational speech, in Proc. ICASSP, 1999.
[70] Minami, Y.; McDermott, E.; Nakamura, A.; Katagiri, S.: A recognition method with parametric trajectory synthesized using directrelations between static and dynamic feature vector time series, in Proc. ICASSP, 2002, 957–960.
[71] Deng, L.; Huang, X.D.: Challenges in adopting speech recognition. Commun. ACM, 47 (1) (2004), 11–13.
[72] Ma, J.; Deng, L.: Efficient decoding strategies for conversational speech recognition using a constrained nonlinear statespacemodel. IEEE Trans. Speech Audio Process., 11 (6) (2003), 590–602.
[73] Ma, J.; Deng, L.: Target-directed mixture dynamic models for spontaneous speech recognition. IEEE Trans. Speech Audio Process., 12(1) (2004), 47–58.
[74] Deng, L.; Yu,D.;Acero, A.: Structured speechmodeling. IEEE Trans. Audio Speech Lang. Process., 14 (5) (2006), 1492–1504.
[75] Deng, L.; Yu, D.; Acero, A.: A bidirectional target filtering model of speech coarticulation: two-stage implementation for phoneticrecognition. IEEE Trans. Audio Speech Process., 14 (1) (2006a), 256–265.
[76] Deng, L.; Yu, D.: Use of differential cepstra as acoustic features in hidden trajectory modeling for phonetic recognition, in Proc ICASSP, April 2007.
[77] Bilmes, J.; Bartels, C.: Graphical model architectures for speech recognition. IEEE Signal Process. Mag., 22 (2005), 89–100.
[78] Bilmes, J.: Dynamic graphical models. IEEE Signal Process.Mag., 33 (2010), 29–42.
[79] Rennie, S.;Hershey,H.; Olsen, P.: Single-channelmultitalker speech recognition – graphical modeling approaches. IEEE Signal Process.Mag., 33 (2010), 66–80.
[80] Wohlmayr, M.; Stark, M.; Pernkopf, F.: A probabilistic interaction model for multipitch tracking with factorial hiddenMarkov model.IEEE Trans. Audio Speech, Lang. Process., 19 (4) (2011).
[81] Stoyanov, V.; Ropson, A.; Eisner, J.: Empirical risk minimization of graphicalmodel parameters given approximate inference, decoding,and model structure, in Proc. AISTAT, 2011.
[82] Kurzweil, R.:How to Create aMind. Viking Books, December, 2012.
[83] Fine, S.; Singer, Y.; Tishby, N.: The Hierarchical Hidden Markov Model: analysis and applications. Mach. Learn., 32 (1998), 41–62.
[84] Oliver, N.; Garg, A.; Horvitz, E.: Layered representations for learning and inferring office activity from multiple sensory channels.Comput. Vis. Image Understand., 96 (2004), 163–180.
[85] Taylor, G.; Hinton, G.E.; Roweis, S.:Modeling human motion using binary latent variables, in Proc. NIPS, 2007.

[86] Socher, R.; Lin, C.; Ng, A.; Manning, C.: Learning continuous phrase representations and syntactic parsing with recursive neuralnetworks, in Proc. ICML, 2011.
[87] Juang, B.-H., Chou, W.; Lee, C.-H.: Minimum classification error rate methods for speech recognition. IEEE Trans. Speech AudioProcess., 5 (1997), 257–265.
[88] Chengalvarayan, R.; Deng, L.: Speech trajectory discrimination using the minimumclassification error learning. IEEETrans. SpeechAudio Process., 6 (6) (1998), 505–515.
[89] Povey,D.;Woodland, P.:Minimumphone error and i-smoothing for improved discriminative training, in Proc. ICASSP, 2002, 105–108.
[90] He, X.; Deng, L.; Chou, W.: Discriminative learning in sequential pattern recognition – a unifying review for optimization-orientedspeech recognition. IEEE Signal Process. Mag., 25 (2008), 14–36.
[91] Jiang, H.; Li, X.: Parameter estimation of statistical models using convex optimization: an advanced method of discriminative trainingfor speech and language processing. IEEE Signal Process. Mag., 27 (3) (2010), 115–127.
[92] Yu, D.; Deng, L.; He, X.; Acero, X.: Large-margin minimum classification error training for large-scale speech recognition tasks, inProc. ICASSP, 2007.
[93] Xiao, L.; Deng, L.: A geometric perspective of large-margin training of Gaussian models. IEEE Signal Process. Mag., 27 (6) (2010), 118–123.
[94] Gibson, M.; Hain, T.: Error approximation and minimum phone error acoustic model estimation. IEEE Trans. Audio Speech, Lang.Process., 18 (6) (2010), 1269–1279.
[95] Yang, D.; Furui, S.: Combining a two-step CRF model and a joint source channel model for machine transliteration, in Proc. ACL,Uppsala, Sweden, 2010, 275–280.
[96] Yu,D.;Wang, S.;Deng, L.: Sequential labeling using deep-structured conditional randomfields. J. Sel. Top. Signal Process., 4 (2010), 965–973.
[97] Hifny, Y.; Renals, S.: Speech recognition using augmented conditional random fields. IEEE Trans. Audio Speech Lang. Process., 17(2) (2009), 354–365.
[98] Heintz, I.; Fosler-Lussier, E.; Brew, C.: Discriminative input stream combination for conditional random field phone recognition. IEEETrans. Audio Speech Lang. Process., 17 (8) (2009), 1533–1546.
[99] Zweig, G.; Nguyen, P.: A segmental CRF approach to large vocabulary continuous speech recognition, in Proc. ASRU, 2009.
[100] Peng, J.; Bo, L.; Xu, J.: Conditional neural fields, in Proc. NIPS, 2009.
[101] Heigold, G.; Ney, H.; Lehnen, P.; Gass, T.; Schluter, R.: Equivalence of generative and log-liner models. IEEE Trans. Audio Speech Lang. Process., 19 (5) (2011), 1138–1148.
[102] Yu, D.; Deng, L.: Deep-structured hidden conditional randomfields for phonetic recognition, in Proc. Interspeech, September. 2010.
[103] Yu, D.; Wang, S.; Karam, Z.; Deng, L.: Language recognition using deep-structured conditional random fields, in Proc. ICASSP, 2010,5030–5033.
[104] Pinto, J.; Garimella, S.; Magimai-Doss, M.; Hermansky, H.; Bourlard, H.: Analysis of MLP-based hierarchical phone posteriorprobability estimators. IEEE Trans. Audio Speech Lang. Process., 19 (2) (2011), 225–241.
[105] Ketabdar, H.; Bourlard, H.: Enhanced phone posteriors for improving speech recognition systems. IEEE Trans. Audio Speech Lang.Process., 18 (6) (2010), 1094–1106.
[106] Morgan, N. et al.: Pushing the envelope – aside [speech recognition]. IEEE Signal Process. Mag., 22 (5) (2005), 81–88.
[107] Deng, L.; Yu, D.: Deep Convex Network: a scalable architecture for speech pattern classification, in Proc. Interspeech, 2011.
[108] Deng, L.; Yu, D.; Platt, J.: Scalable stacking and learning for building deep architectures, in Proc. ICASSP, 2012.
[109] Tur, G.; Deng, L.; Hakkani-Tür, D.; He, X.: Towards deep understanding: deep convex networks for semantic utterance classification,in Proc. ICASSP, 2012.
[110] Lena, P.;Nagata, K.; Baldi, P.:Deep spatiotemporal architectures and learning for protein structure prediction, in Proc. NIPS, 2012.
[111] Hutchinson, B.; Deng, L.; Yu, D.: A deep architecture with bilinear modeling of hidden representations: applications to phoneticrecognition, in Proc. ICASSP, 2012.

[112] Hutchinson, B.; Deng, L.; Yu, D.: Tensor deep stacking networks, IEEE Trans. Pattern Anal. Mach. Intell., 35 (2013), 1944–1957.
[113] Deng, L.; Hassanein, K.; Elmasry, M.: Analysis of correlation structure for a neural predictive model with application to speech recognition. Neural Netw., 7 (2) (1994a), 331–339.
[114] Robinson, A.: An application of recurrent nets to phone probability estimation. IEEE Trans. Neural Netw., 5 (1994), 298–305.
[115] Graves, A.; Fernandez, S.; Gomez, F.; Schmidhuber, J.: Connectionist temporal classification: labeling unsegmented sequence data withrecurrent neural networks, in Proc. ICML, 2006.
[116] Graves, A.;Mahamed, A.; Hinton, G.: Speech recognition with deep recurrent neural networks, in Proc. ICASSP, 2013.
[117] Graves, A.: Sequence transduction with recurrent neural networks, in Representation LearningWorksop, ICML, 2012.
[118] LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE, 86 (1998), 2278–2324.
[119] Ciresan, D.; Giusti, A.; Gambardella, L.; Schidhuber, J.: Deep neural networks segment neuronal membranes in electron microscopyimages, in Proc. NIPS, 2012.
[120] Dean, J. et al.: Large scale distributed deep networks, in Proc. NIPS, 2012.
[121] Krizhevsky, A.; Sutskever, I.; Hinton, G.: ImageNet classification with deep convolutional neural Networks, in Proc. NIPS, 2012.
[122] Abdel-Hamid, O.;Mohamed, A.; Jiang, H.; Penn, G.: Applying convolutional neural networks concepts to hybridNN-HMMmodel forspeech recognition. in ICASSP, 2012.
[123] Abdel-Hamid, O.; Deng, L.; Yu, D.: Exploring convolutional neural network structures and optimization for speech recognition. in Proc. Interspeech, 2013.
[124] Abdel-Hamid,O.;Deng, L.; Yu,D.; Jiang, H.:Deep segmental neural networks for speech recognition, in Proc. Interspeech, 2013a.
[125] Sainath, T.;Mohamed, A.; Kingsbury, B.; Ramabhadran, B.: Convolutional neural networks for LVCSR, in Proc. ICASSP, 2013.
[126] Deng, L.;Abdel-Hamid,O.; Yu,D.:Adeep convolutional neural network using heterogeneous pooling for trading acoustic invariancewith phonetic confusion, in Proc. ICASSP, 2013.
[127] Lang, K.;Waibel, A.;Hinton,G.: A time-delay neural network architecture for isolated word recognition. Neural Netw., 3 (1) (1990),23–43.
[128] Hawkins, J.; Blakeslee, S.: On Intelligence: How a New Understanding of the Brain will lead to the Creation of Truly IntelligentMachines, Times Books, New York, 2004.
[129] Waibel, A.; Hanazawa, T.; Hinton, G.; Shikano, K.; Lang, K.: Phoneme recognition using time-delay neural networks. IEEETrans. ASSP, 37 (3) (1989), 328–339.
[130] Hawkins, G.; Ahmad, S.; Dubinsky, D.: Hierarchical Temporal Memory including HTM Cortical Learning Algorithms. NumentaTechnical Report, December 10, 2010.
[131] Lee, C.-H.: From knowledge-ignorant to knowledge-rich modeling: a new speech research paradigm for next-generation automaticspeech recognition, in Proc. ICSLP, 2004, 109–111.
[132] Yu,D.; Siniscalchi, S.;Deng, L.;Lee,C.: Boosting attribute andphone estimation accuracies with deep neural networks for detectionbasedspeech recognition, in Proc. ICASSP, 2012.
[133] Siniscalchi, M.; Yu, D.; Deng, L.; Lee, C.-H.: Exploiting deep neural networks for detection-based speech recognition.Neurocomputing,106 (2013), 148–157.
[134] Siniscalchi, M.; Svendsen, T.; Lee, C.-H.: A bottom-up modular search approach to large vocabulary continuous speech recognition.IEEE Trans. Audio Speech, Lang. Process., 21 (2013), 786–797.
[135] Yu, D.; Seide, F.; Li, G.; Deng, L.: Exploiting sparseness in deep neural networks for large vocabulary speech recognition, in Proc.ICASSP, 2012.
[136] Deng, L.; Sun, D.: A statistical approach to automatic speech recognition using the atomic speech units constructed from overlappingarticulatory features. J. Acoust. Soc. Am., 85 (5) (1994), 2702–2719.
[137] Sun, J.; Deng, L.: An overlapping-feature based phonological model incorporating linguistic constraints: applications to speech recognition. J. Acoust. Soc. Am., 111 (2) (2002), 1086–1101.
[138] Sainath, T.; Kingsbury, B.; Ramabhadran, B.: Improving training time of deep belief networks through hybrid pre-training and largerbatch sizes, inProc.NIPSWorkshop on Log-linearModels, December 2012.

[139] Erhan, D.; Bengio, Y.; Courvelle, A.; Manzagol, P.; Vencent, P.; Bengio, S.:Why does unsupervised pre-training help deep learning?J. Mach. Learn. Res., 11, (2010), 625–660.
[140] Kingsbury, B.: Lattice-based optimization of sequence classificationcriteria for neural-network acousticmodeling, in Proc. ICASSP, 2009.
[141] Kingsbury, B.; Sainath, T.; Soltau, H.: Scalable minimum Bayes risk training of deep neural network acoustic models using distributedHessian-free optimization, in Proc. Interspeech, 2012.
[142] Larochelle, H.; Bengio, Y.: Classification using discriminative restricted Boltzmann machines, in Proc. ICML, 2008.
[143] Lee, H.; Grosse, R.; Ranganath, R.; and Ng, A.: Unsupervised learning of hierarchical representations with convolutional deep beliefnetworks, Communications of the ACM, Vol. 54, No. 10, October, 2011, pp. 95–103.
[144] Lee, H.; Grosse, R.; Ranganath, R.; Ng, A.: Convolutional Deep Belief Networks for Scalable Unsupervised Learning ofHierarchicalRepresentations, Proc. ICML, 2009.
[145] Lee, H.; Largman, Y.; Pham, P.; Ng, A.: Unsupervised feature learning for audio classification using convolutional deep belief networks,Proc. NIPS, 2010.
[146] Ranzato, M.; Susskind, J.; Mnih, V.; Hinton, G.: On deep generative models with applications to recognition, in Proc. CVPR, 2011.
[147] Ney,H.: Speech translation: coupling of recognition and translation, in Proc. ICASSP, 1999.
[148] He, X.; Deng, L.: Speech recognition, machine translation, and speech translation – a unifying discriminative framework. IEEESignal Process. Mag., 28 (2011), 126–133.
[149] Yamin, S.;Deng, L.;Wang, Y.; Acero, A.: An integrative and discriminative technique for spoken utterance classification. IEEE Trans.Audio Speech Lang. Process., 16 (2008), 1207–1214.
[150] He, X.; Deng, L.: Optimization in speech-centric information processing: criteria and techniques, in Proc. ICASSP, 2012.
[151] He, X.; Deng, L.: Speech-centric information processing: an optimization-oriented approach, in Proc. IEEE, 2013.
[152] Deng, L.; He, X.; Gao, J.: Deep stacking networks for information retrieval, in Proc. ICASSP, 2013a.
[153] He, X.; Deng, L.; Tur, G.; Hakkani-Tur, D.: Multi-style adaptive training for robust cross-lingual spoken language understanding, inProc. ICASSP, 2013.

[154] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In Proceedings of the Eleventh Annual Conference on Computational Learning Theory, pages 92–100, Madison, Wisconsin, United States, 1998. ACM.
[155] T. Joachims. Transductive inference for text classifcation using support vector machines. In Proceedings of International Conference on Machine Learning, pages 200–209, 1999.
[156] O. Chapelle, V. Sindhwani, and S. S. Keerthi. Optimization techniques for semi-supervised support vector machines. The Journal of Machine Learning Research, 9:203–233, 2008.
[157] N. V. Chawla and G. Karakoulas. Learning from labeled and unlabeled data: An empirical study across techniques and domains. Journal of Artificial Intelligence Research, 23(1):331–366, 2005.
[158] S. Goldman and Y. Zhou. Enhancing supervised learning with unlabeled data. In Proceedings of the 17th International Conference on Machine Learning, pages 327–334, 2000.
[159] M.-F. Balcan, A. Blum, and K. Yang. Co-training and expansion: towards bridging theory and practice. In Advances in Neural Information Processing Systems, volume 17, pages 89–96. 2005.
[160] W. Wang and Z.-H. Zhou. A new analysis of co-training. In Proceedings of the 27th International Conference on Machine Learning, pages 1135–1142, Haifa, Israel, June 2010.
[161] M.-A. Krogel and T. Scheffer. Multi-relational learning, text mining, and semi-supervised learning for functional genomics. Machine Learning, 57(1):61–81, 2004.
[162] V. Vapnik. Statistical Learning Theory. Wiley, New York, 1998.
[163] T. Joachims. Transductive inference for text classifcation using support vector machines. In Proceedings of International Conference on Machine Learning, pages 200–209, 1999.
[164] O. Chapelle and A. Zien. Semi-supervised classification by low density separation. In Proceedings of International Conference on Artificial Intelligence and Statistics, Barbados, January 2005.

[165] O. Chapelle, V. Sindhwani, and S. S. Keerthi. Branch and bound for semi-supervised support vector machines. In Advances in Neural Information Processing Systems, volume 19, pages 217–224. Cambridge, MA, 2007

[166] T. D. Bie and N. Cristianini. Convex methods for transduction. In Advances in Neural Information Processing Systems, volume 16. 2004.

[167] Z. Xu, R. Jin, J. Zhu, I. King, and M. Lyu. Efficient convex relaxation for transductive support vector machine. volume 21, pages 1641–1648. 2008.

[168] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In Proceedings of International Conference on Machine Learning, pages 19–26, San Francisco, CA, USA, 2001.

[169] A. Blum, J. Lafferty, M. R. Rwebangira, and R. Reddy. Semi-supervised learning using randomized mincuts. In Proceedings of the Twenty-first International Conference onMachine Learning, pages 13–20, Banff, Alberta, Canada, 2004.

[170] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In Proceedings of International Conference on Machine Learning, pages 19–26, San Francisco, CA, USA, 2001.

[171] T. Joachims. Transductive learning via spectral graph partitioning. In Proceedings of InternationalConference on Machine Learning, pages 290–297, 2003.

[172] B. Kveton, M. Valko, A. Rahimi, and L. Huang. Semi-supervised learning with max-margin graph cuts. In Proceedings of the 13th International Conference on Artificial Intelligence and Statistics, pages 421–428, 2010.

[173] A. Azran. The rendezvous algorithm: Multiclass semi-supervised learning with markov random walks. In Proceedings of the International Conference on Machine Learning, pages 49–56, Corvalis, Oregon, 2007. ACM.

[174] M. Szummer and T. Jaakkola. Partially labeled classification with Markov random walks. In Advances in Neural Information Processing Systems, volume 14, pages 945–952. 2002.

[175] M. Belkin, P. Niyogi, and V. Sindhwani. On manifold regularization. In Proceedings of International Conference on Artificial Intelligence and Statistics, Barbados, January 2005.

[176] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: a Geometric framework for learning from labeled and unlabeled examples. Journal of Machine Learning Research, 7:2399–2434, 2006.

[177] V. Sindhwani, P. Niyogi, and M. Belkin. Beyond the point cloud: from transductive to semisupervised learning. In Proceedings of the 22nd International Conference on Machine Learning, pages 824–831, Bonn, Germany, 2005.

[178] V. Sindhwani, J. Hu, and A. Mojsilovic. Regularized co-clustering with dual supervision. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, Advances in Neural Information Processing Systems, pages 976–983. MIT Press, 2008.

[179] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Sch¨olkopf. Learning with local and global consistency. In S. Thrun, L. Saul, and B. Sch¨olkopf, editors, Advances in Neural Information Processing Systems, volume 16, pages 321–328. 2004.

[180] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In Proceedings of International Conference on Machine Learning, pages 912–919, 2003.

[181] X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences,University of Wisconsin-Madison, 2007.

[182] J. Wang, T. Jebara and S. Chang. Semi-supervised learning using greedy max-cut. The Journal of Machine Learning Research, 14(1), 771-800.

[183] W. Liu, J. He, and S. Chang. Large graph construction for scalable semi-supervised learning. In Proceedings of the 27th international conference on machine learning (ICML-10) (pp. 679-686).

[184] D.G. Lowe,"Object recognition from local scale-invariant features." In ICCV, vol. 2, pp. 1150-1157, 1999.

[185] H. Bay, T. Tuytelaars, and L. Van Gool, “Surf: Speeded up robust features”. In ECCV 2006 (pp. 404-417).

[186] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection”. In CVPR 2005.

[187] T. Ojala,M. Pietikainen and T. Maenpaa, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns”. IEEE TPAMI,, 24(7), 971-987.

[188] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” inProc. CVPR, 2009.

[189] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradientbased learning applied to document recognition. Proc. of theIEEE, 1998. 1

[190] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In NIPS, 2012. 1, 3, 4, 7

[191] W. Liu, J. He, and S. Chang. Large graph construction for scalable semi-supervised learning. In Proceedings of the 27th international conference on machine learning (ICML-10) (pp. 679-686).

[192] J. Pynn, A. Wright, and R. Lodge, "Automatic identification of cracks in road surfaces," in Image Processing and Its Applications, 1999. Seventh International Conference on (Conf. Publ. No. 465), vol. 2, 1999, pp.671–675 vol.2.

[193] Y.-S. Kim and C. T. Haas, "A model for automation of infrastructure maintenance using representational forms," Automation in Construction, vol. 10, no. 1, pp. 57–68, Nov. 2000.

[194] P.-C. Tung, Y.-R. Hwang, and M.-C. Wu, "The development of a mobile manipulator imaging system for bridge crack inspection," Automation in Construction, vol. 11, no. 6, pp. 717–729, Oct. 2002.

[195] I. Abdel-Qader, O. Abudayyeh, and M. E. Kelly, "Analysis of edge detection techniques for crack identification in bridges," Journal of Computing in Civil Engineering, vol. 17, no. 4, pp. 255–263, 2003.

[196] S. K. Sinha and P. W. Fieguth, "Automated detection of cracks in buried concrete pipe images," Automation in Construction, vol. 15, no. 1, pp. 58–72, 2006.

[197] Z. Liu, S. A. Suandi, T. Ohashi, and T. Ejima, "Tunnel crack detection and classification system based on image processing," in Electronic Imaging 2002. International Society for Optics and Photonics, 2002, pp. 145–152.

[198] H. Son, C. Kim, and C. Kim, "Automated color model based concrete detection in construction-site images by using machine learning algorithms," Journal of Computing in Civil Engineering, vol. 26, no. 3, pp. 421–433, 2012.

[199] S. Yu, J. Jang, and C. Han, "Auto inspection system using a mobile robot for detecting concrete cracks in a tunnel," Journal of Computation in Civil Engineering, vol. 17, no. 4, pp. 255–263, 2006.

[200] A. Mohanty and T. T. Wang, "Image mosaicking of a section of a tunnel lining and the detection of cracks through the frequency histogram of connected elements concept," vol. 8335, 2012, pp. 83 351P–83 351P–9.