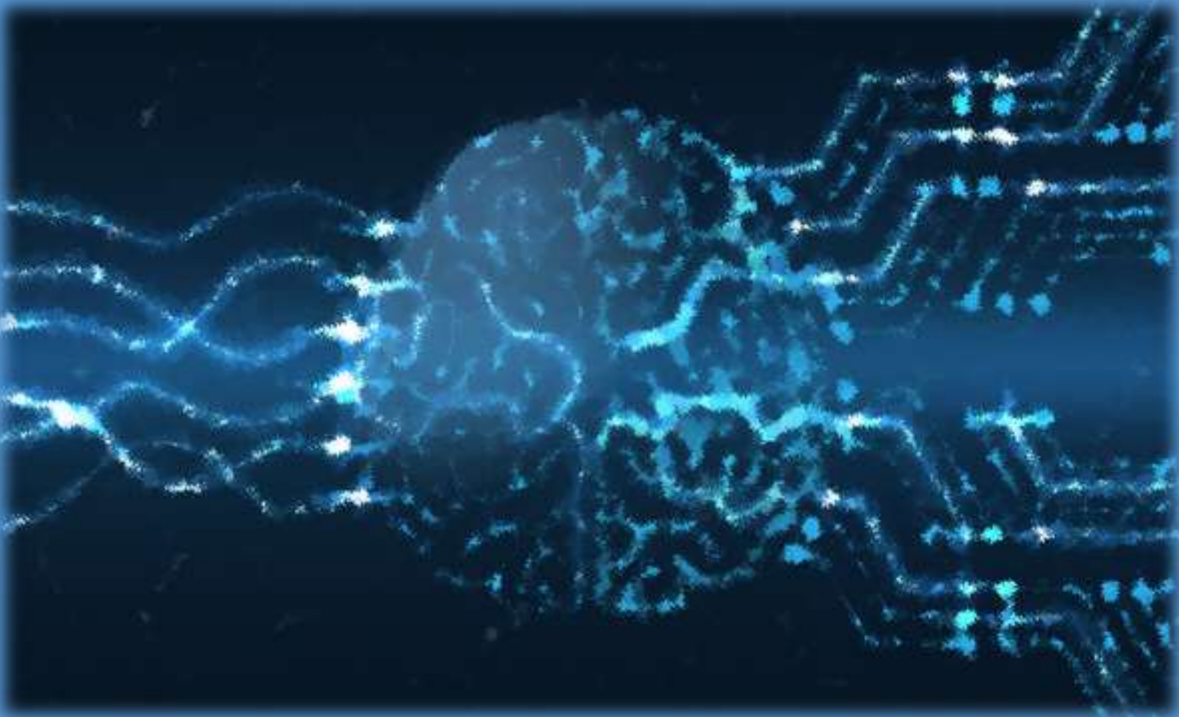
Eftychios Protopapadakis
NATIONAL TECHNICAL UNIVERSITY OF ATHENS
Zografou Campus

Combinatory semi-supervised deep learning models for decision making



This research is co-financed by Greece and the European Union (European Social Fund- ESF) through the Operational Programme «Human Resources Development, Education and Lifelong Learning» in the context of the project "Reinforcement of Postdoctoral Researchers" (MIS-5001552), implemented by the State Scholarships Foundation (IKY).

Ευρωπαϊκή Ένωση
European Social Fund

Operational Programme
Human Resources Development, Education and Lifelong Learning
Co-financed by Greece and the European Union

ΕΣΠΑ 2014-2020
ανάπτυξη - εργασία - αλληλεγγύη

The School of Rural and Surveying Engineering nominated the following person to serve as the Post-Doctoral Committee:

Post-Doctoral Committee

1	**Anastasios**	**Doulamis**	(supervisor)	
	National Technical University of Athens		School of Rural and Surveying Engineering	Assistant Professor
2	**Andreas**	**Georgopoulos**	(advisor)	
	National Technical University of Athens		School of Rural and Surveying Engineering	Professor
3	**Konstantinos**	**Karantzalos**	(advisor)	
	National Technical University of Athens		School of Rural and Surveying Engineering	Associate Professor

Acknowledgments

This research was implemented through IKY scholarships program and co-financed by the European Union (European Social Fund - ESF) and Greek national funds through the action entitled «Reinforcement of Postdoctoral Researchers», in the framework of the Operational Programme «Human Resources Development Program, Education and Lifelong Learning» of the National Strategic Reference Framework (NSRF) 2014 – 2020.

Special Thanks

Many thanks to my family and all those who were like a family to me. The life of a researcher has various unexpected situations, which can become a burden, if you do not have reliable persons by your side.

I would like to express my special appreciation and thanks to my advisor, Dr. Anastasios Doulamis, who always had one or two suggestions for the problem at hand. Thank you for encouraging my research and let me run it at my own pace.

Thankfully, the fellow researchers at the laboratory are capable persons, which allowed for multiple experimental setups in various research fields. The committee members Andreas Georgopoulos and Konstantinos Karantzalos provided focus and direction, emphasizing on rural and surveying applications.

Last but not least, many thanks to Athanasios Voulodimos and Nikolaos Doulamis, for ideas and support.

How to cite

If you would like to cite this work, I recommend using the following bibtex entry:

```
@phdthesis{protopapadakis_combinatory_2019,
	title = {Combinatory semi-supervised deep learning models for decision making},
	school = {National Technical University of Athens},
	author = {Protopapadakis, Eftychios},
	year = {2019}
}
```

Author's info

Eftychios Protopapadakis studied production engineering and management at technical university of Crete. His educational background includes a M.S. degree in management and business administrator and a Ph.D. in decision systems design, both at the same university. He has been working as engineer in European and Interegg projects since 2010.

His research interests focus on machine learning applications in multiple research fields. He has explored the applicability of semi-supervised techniques in maritime surveillance, elder people support, industrial workflow monitoring, and cultural heritage applications. Other investigated areas involve stock market share trends' forecasting and credit risk assessment. Additionally, he has worked on structural assessment in transportation tunnel infrastructures, via deep-learning techniques and robotic platforms, on intangible cultural assets digitization, and intelligent parking management policies. Current research activities involve energy consumption reduction recommendation systems setup.

Eftychios co-authored more than fifty-five publications. His paper on industrial workflow recognition received the best paper award in INFOCOMP 2012. Other awards include university scholarships for excellence, Technical Chamber of Greece award for excellence in studies, state scholarship foundation – SIEMENS doctorate scholarship 2012.

eftprot<youknowwhat>mail<dot>ntua<dot>com
https://gr.linkedin.com/pub/eftychios-protopapadakis/bb/75a/427
https://www.researchgate.net/profile/Eftychios_Protopapadakis

Περίληψη

Η παρούσα έρευνα αναζητά τρόπους ανάπτυξης μοντέλων λήψης αποφάσεων, που αξιοποιούν τα πλεονεκτήματα από δύο περιοχών του ευρύτερου ερευνητικού τομέα της μηχανικής μάθησης: α) την βαθιά μάθηση (deep learning) και β) την μάθηση με μερική επίβλεψη (semi-supervised learning). Οι υφιστάμενες υλοποιήσεις αξιολογούν το κατά πόσο είναι δυνατό να μετριάσουμε τα όποια μειονεκτήματα των τεχνικών βαθιάς μάθησης, με την μικρότερη δυνατή προσπάθεια από την πλευρά του χρήστη, αξιοποιώντας μη τιτλοφορημένα δεδομένα (unlabeled data).

Η βαθιά μάθηση στηρίζεται στην δημιουργία σύνθετων μοντέλων (πολλαπλά διατεταγμένα επίπεδα υπολογιστικών μονάδων) τα οποία εξάγουν αυτόματα περιγραφικά χαρακτηριστικά (feature values). Στην διεθνή βιβλιογραφία υπάρχει σημαντικός όγκος ερευνητικών εργασιών που επιβεβαιώνουν, πειραματικά, την καταλληλότητα των τεχνικών σε σύνθετα προβλήματα. Εντούτοις, εντοπίζονται δυο βασικά μειονεκτήματα αποτελούν. Αφενός, απαιτείται μεγάλος όγκος δεδομένων εκπαίδευσης και, αφετέρου, ο καθορισμός της τοπολογίας του δικτύου (layers' structure) απαιτεί πολύ χρόνο και υπολογιστικούς πόρους και εξαρτάται από την εμπειρία των ερευνητών πάνω στον τομέα έρευνας.

Η μάθηση με μερική επίβλεψη αξιοποιεί διαθέσιμη πληροφορία, που συναντάται σε μη τιτλοφορημένα δεδομένα (unlabeled data), και την ενσωματώνει ως επιπλέον ρυθμιστικούς όρους (regularizers) στις υφιστάμενες τεχνικές μάθησης (π.χ. SVMs). Οι ρυθμιστικοί όροι βελτιώνουν την απόδοση του μοντέλου, χωρίς να απαιτείται επιπλέον προσπάθεια από τον χρήστη για την δημιουργία μεγάλου όγκου δεδομένων εκπαίδευσης. Το βασικό μειονέκτημα έγκειται στο ότι κάθε τεχνική προϋποθέτει να ικανοποιούνται συγκεκριμένα κριτήρια/ υποθέσεις, σχετικές με τον χώρο των δεδομένων.

Οι υλοποιήσεις που παρουσιάζονται στην έρευνα αφορούν: α) στην επιλογή περιορισμένου αριθμού αντιπροσωπευτικών δεδομένων εκπαίδευσης, β) στην τυπολογική βελτιστοποίηση νευρωνικών δικτύων με χρήση γενετικών αλγορίθμων με συναρτήσεις καταλληλότητας από το πεδίο της μάθησης με μερική επίβλεψη, και γ) στην ενημέρωση των συναπτικών βαρών του δικτύου χρησιμοποιώντας και τα μη τιτλοφορημένα δεδομένα. Οι περιπτώσεις α και γ αντιμετωπίζουν το πρόβλημα του περιορισμένου όγκου δεδομένων εκπαίδευσης ενώ η περίπτωση β αφορά στον καθορισμό της τοπολογίας.

Η τελευταία ενότητα αναλύει πως οι συγκεκριμένες τεχνικές μπορούν να αξιοποιηθούν στο πεδίο ενασχόλησης των αγρονόμων και τοπογράφων μηχανικών. Το συγκεκριμένο παράδειγμα επικεντρώνεται στον εντοπισμό κτισμάτων από αεροφωτογραφίες. Η ιδέα έγκειται στην εκπαίδευση ενός συνθέτου μοντέλου (deep neural network) αξιοποιώντας ένα εξαιρετικά περιορισμένο σύνολο δεδομένων, τα οποία συγκεντρωθήκαν μέσω πληθοπορισμού (crowdsourcing). Τα πειραματικά αποτελέσματα καταδεικνύουν την καταλληλότητα των προτεινόμενων τεχνικών.

Οι ακόλουθες δημοσιεύσεις ήταν αποτέλεσμα της παρούσας έρευνας:

1. A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Computational intelligence and neuroscience*, vol. 2018, 2018.
2. E. Protopapadakis, A. Voulodimos, A. Doulamis, N. Doulamis, and T. Stathaki, "Automatic crack detection for tunnel inspection using deep learning and heuristic image post-processing," *Appl Intell*, Feb. 2019.
3. E. Protopapadakis, D. Niklis, M. Doumpos, A. Doulamis, and C. Zopounidis, "Sample selection algorithms for credit risk modelling through data mining techniques," IJDMMM, vol. 11, no. 2, pp. 103–128, 2019.
4. E. Protopapadakis, A. Voulodimos, N. Doulamis, and A. Doulamis, "Image Clustering and Video Summarization for efficient 3D Modeling and Reconstruction", Book title: Recent Advances in 3D Imaging, Modeling, and Reconstruction, 2019 (under publication).

5. E. Protopapadakis, A. Voulodimos, and N. Doulamis, "An investigation on multi-objective optimization of feedforward neural network topology," in *Information, Intelligence, Systems & Applications (IISA), 2017 8th International Conference on*, 2017, pp. 1–6.
6. E. Protopapadakis, A. Voulodimos, and A. Doulamis, "Data sampling for semi-supervised learning in vision-based concrete defect recognition," in 2017 8th International Conference on Information, Intelligence, Systems Applications (IISA), 2017, pp. 1–6.

Abstract

This postdoctoral research emphasized on the creation of decision support systems, in a way that both the advantages of deep and semi supervised learning are used. All the employed techniques evaluate the possibilities of using deep learning approaches, in a way that the effort from user's side is minimal; this is achieved by utilizing unlabeled data instances, through SSL approaches.

On the one hand, deep learning (DL) core is the creation of complex models, e.g. multiple hierarchical computational layers, capable to extract high level (descriptive) features, depending on the application scenario. Multiple research papers provide experimental results, suggesting the DL techniques superiority over traditional approaches. Nevertheless, two major drawbacks are reported: a) the requirement of many labeled data, which will be used for training and b) hyperparameter setup, e.g. network's topology.

On the other hand, SSL models build on the information provided by unlabeled data entries, which is encoded in the form of regularizes, i.e. additional terms in the loss function of the existing approach, e.g. SVM. That way any effort on labeling the data (i.e. creating data set) is kept low. The main drawback is the assumptions required, for the creation of the regularizing terms.

Taking under consideration all the above, the combination of DL with SSL appears to be an intuitive next-step approach. We could establish models capable to handle complex patterns and at the same time reducing any efforts on the preparation of training sets and enhancing outcomes robustness.

In the following chapters you may find research implementations involving: a) representative sample selection, for the creation of the training data set, b)setting up the topology of neural networks through multi-objective optimization algorithms and semi-supervised fitness functions, and c) fine-tuning of the synaptic weights using SSL inspired performance functions.

The last chapter demonstrates how such combinatory approaches can improve the accuracy and reduce required effort, while developing models for building detection. The case study focuses on the combination of data, e.g. aerial images combined with the corresponding Dense Image Matching point clouds. Data are feed to a semi-supervised trained network scheme. The main contribution lies in the incorporation of all available data, during all steps of the training process, regardless their initial status (i.e. labeled or unlabeled). Annotations became available using crowdsourcing, on a limited amount of data.

The following publications were the outcomes of this research:

1. A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Computational intelligence and neuroscience*, vol. 2018, 2018.
2. E. Protopapadakis, A. Voulodimos, A. Doulamis, N. Doulamis, and T. Stathaki, "Automatic crack detection for tunnel inspection using deep learning and heuristic image post-processing," *Appl Intell*, Feb. 2019.
3. E. Protopapadakis, D. Niklis, M. Doumpos, A. Doulamis, and C. Zopounidis, "Sample selection algorithms for credit risk modelling through data mining techniques," IJDMMM, vol. 11, no. 2, pp. 103–128, 2019.

4. E. Protopapadakis, A. Voulodimos, N. Doulamis, and A. Doulamis, "Image Clustering and Video Summarization for efficient 3D Modeling and Reconstruction", Book title: Recent Advances in 3D Imaging, Modeling, and Reconstruction, 2019 (under publication).
5. E. Protopapadakis, A. Voulodimos, and N. Doulamis, "An investigation on multi-objective optimization of feedforward neural network topology," in *Information, Intelligence, Systems & Applications (IISA), 2017 8th International Conference on*, 2017, pp. 1–6.
6. E. Protopapadakis, A. Voulodimos, and A. Doulamis, "Data sampling for semi-supervised learning in vision-based concrete defect recognition," in *2017 8th International Conference on Information, Intelligence, Systems Applications (IISA)*, 2017, pp. 1–6.

Looking back in in 1943, Warren McCulloch and Walter Pitts introduced the early model of an artificial neuron. More complex structures, containing multiple neurons appeared through the years, for various applications. However, despite the promising experimental results, no mathematical theory could describe the bounds (if any) of such techniques. Many years had to pass before a mathematical proof was established, indicating what had already been observed: artificial neural networks can be used as universal approximators. It was 1989; George Cybenko's work on sigmoid functions (Cybenko, 1989), set the basis for the universal approximation theorem (UAT).

UAT states that we can approximate continuous functions on compact subsets, using a finite number of known functions, if three assumptions, related to the function form (bounded, increasing, and continuous), are valid. Then, it was Kurt Hornik (Hornik, 1991), who showed that combining multiple neurons, using sigmoid activation functions, i.e. creating multilayer feedforward architecture, gives the potential of creating universal approximators. That was a milestone in the artificial neural networks (ANNs) research field. Since then, ANNs have been extensively used in a variety of applications.

ANNs are complex topologies containing hidden layers, neurons (i.e. activation functions), synaptic weights, biases, and other parameters. Typically, the contributions of the neurons are regulated through synaptic weights and biases, both of which are randomly initialized and then adjusted by backpropagation algorithm, during a training phase. The importance of the back-propagation algorithm was established at late 80s (Rumelhart et al., 1986). As such, among other things, data availability and training paradigms features' quality are crucial when creating an ANN inference system.

Nowadays data availability is not a problem; it has, mainly, the form of images, video/audio files or texts and it is available online. Therefore, research trends focus on available data utilization, to create appropriate structures and inference mechanisms. In the end, created structures produce an outcome for specific data, at a user's request (e.g. recommendation of a movie, image identification, song recognition, etc.). A system that understands data patters and can support a decision-making process is called Decision Support System (DSS). DSSs exploit a variety of methods from the machine learning field.

Regardless of the application scenario, the main goal of any DSS is always the same: *Provide information to the user, in a meaningful way, supporting the decision-making process*. Available data affects DSS performance (Chen et al., 2012; Demirkan and Delen, 2013); *the more we have the better the performance,* if we have data of good quality. Nevertheless, *data abundance does not imply good features quality, neither explicit information over all of them*. The former case, i.e. feature quality, can be handled by using deep learning (DL) models. The latter case, i.e. using unlabeled data, can be handled using semi-supervised learning (SSL) techniques.

DL approaches allow for an automated feature extraction process. Thus, there is limited need for custom-made feature extraction techniques, employed by an expert. The DL models require large, annotated data sets. The annotation of any large data set can be time consuming and is prone to the human error. Typically, DL models are trained using thousands of samples over log time periods. Further details regarding DL can be found in (Deng and Yu, 2014).

SSL approaches allow the utilization of unlabeled data. An approach that emerged naturally, as the data availability grow bigger over the years. In this case, employed techniques exploit experts' knowledge on a small portion of data, together with a many times greater portion, available online (van Engelen and Hoos, 2020).

Contents

Chapter I: Introduction

Human behavior flows from three main sources: desire, emotion, and knowledge.

Plato, Greek philosopher

1 Introduction

The term deep learning (DL) is used to describe a class of machine learning algorithms, which share some common characteristics; i.e. many layers of nonlinear processing units, for feature extraction and transformation, and unsupervised learning of multiple levels of features or representations of the data. To put it simply, research in this area attempts to make better representations and create models, to learn these representations, from large-scale unlabeled data, e.g. stacked autoencoders (Eftychios Protopapadakis et al., 2017a), or large scale labeled data, e.g. convolutional neural networks (Kaselimi et al., 2019).

Deep learning emerged naturally as the next step in the machine learning field. It was a decade ago (i.e. 2006) when most machine learning and signal processing techniques had exploited shallow-structured architectures. These architectures typically contain at most one or two layers of nonlinear feature transformations. Although such techniques were effective in solving many simple or well-constrained problems, they had limited modeling and representational power, which could cause difficulties when dealing with more complicated real-world applications; i.e. handling natural signals such as human speech, natural sound and language, and natural image and visual scenes.

Evolution is a slow process; it requires small steps and is affected by the status-quo. Although, deep learning approaches appeared in 1965 (Ivakhnenko and Lapa, 1967) , DL based on Artificial Neural Networks (ANNs) were introduced latter in 1980 (Fukushima and Miyake, 1982). From this point onward, DL will describe ANN based approaches (Schmidhuber, 2015), unless explicitly stated differently.

The concept of DL originated from artificial neural network research. Hence, one may occasionally hear the discussion of "new-generation neural networks". Feed-forward neural networks or MLPs with many hidden layers, which are often referred to as deep neural networks (DNNs), are good examples of the models with a deep architecture. Back-propagation (BP), popularized in 1980s, has been a well-known algorithm for learning the parameters of these networks. Unfortunately BP alone does not work well in practice, for learning networks with more than two hidden layers; see a review and analysis in (Glorot and Bengio, 2010).

The pervasive presence of local optima and other optimization challenges, in the non-convex objective function of the deep networks, are the main source of difficulties in the learning. BP is based on local gradient information and starts usually at some random initial points. It often gets trapped in poor local optima when the batch-mode or even stochastic gradient descent BP algorithm is used. The severity increases significantly as the depth of the networks increases. This difficulty is partially responsible for steering away most of the machine learning and signal processing research from neural networks to shallow models that have convex loss functions (e.g., SVMs, CRFs, and MaxEnt models), for which the global optimum can be efficiently obtained at the cost of reduced modeling power, although there had been continuing work on neural networks with limited scale and impact, e.g. (Morin and Bengio, 2005).

Chapter II: The Basics

Learning never exhaust the mind.

Leonardo da Vinci, Italian polymath

2 A brief description of employed techniques

In this section we try to grasp, briefly, the notion behind Deep Neural Networks (DNNs) and Semi-Supervised Learning (SSL). As we will see, the core idea lies in *the need for evolution, of existing techniques, in a cost-effective way*. It is extremely important to handle the data abundance, in a meaningful way, in order to: (a) improve model's performance and (b) minimize experts' interventions. As such, the first steps on the SSL field begun around 1960 when scientists started exploiting new (i.e. unseen so far, unlabeled) data to advance their models' abilities. Since then, and up to date, SSL field continues to involve, motivated by the data abundance of our age.

2.1 Deep learning

In this section, we will try to explain, briefly, the context behind DL approaches. DL could be described as a sub-field within machine learning that is based in algorithms for learning multiple levels of representation to model complex relationships among data. Although there are many definitions or high-level descriptions, the core of DL concepts can be summarized using two words: "learning representations" (Voulodimos et al., 2018).

Learning the appropriate representation implies an automated feature extraction process. If there are many training paradigms, effective representation extraction is plausible. Such an ability was more than enough, to trigger an increase in popularity of DL methods. Nowadays, DL research is further supported by (a) the drastically increased chip processing abilities (e.g., general-purpose graphical processing units or GPGPUs), (b) the significantly increased size of data used for training, and the recent advances in machine learning and (c) signal/information processing research.

These advances have enabled the deep learning methods to effectively exploit complex, compositional nonlinear functions, to learn distributed and hierarchical feature representations, and to make effective use of both labeled and unlabeled data. Common among the various high-level descriptions of deep learning above are two key aspects: (a) models consisting of multiple layers or stages of nonlinear information processing; and (b) methods for supervised or unsupervised learning of feature representation at successively higher, more abstract layers.

2.1.1 Key concepts of deep neural networks

Deep-learning networks are distinguished from the more commonplace single-hidden-layer neural networks by their depth; that is, the number of node layers through which data passes in a multistep process of pattern recognition.

Earlier versions of neural networks such as the first perceptrons were shallow, composed of one input and one output layer, and at most one hidden layer in between. More than three layers (including input and output) qualifies as "deep" learning. So deep is a strictly defined, technical term that means more than one hidden layer.

In deep-learning networks, each layer of nodes trains on a distinct set of features based on the previous layer's output. The further you advance into the neural net, the more complex the features your nodes can recognize, since they aggregate and recombine features from the previous layer. Each of these layers is called autoencoder. Stacking many encoding layers results in a deep learning model called stacked autoencoders.

Autoencoders are a specific type of feedforward neural networks where the input is the same as the output. They compress the input into a lower-dimensional code and then reconstruct the output from this representation. The code is a compact “summary” or “compression” of the input, also called the latent-space representation. An autoencoder consists of 3 components: encoder, code and decoder. The encoder compresses the input and produces the code, the decoder then reconstructs the input only using this code.

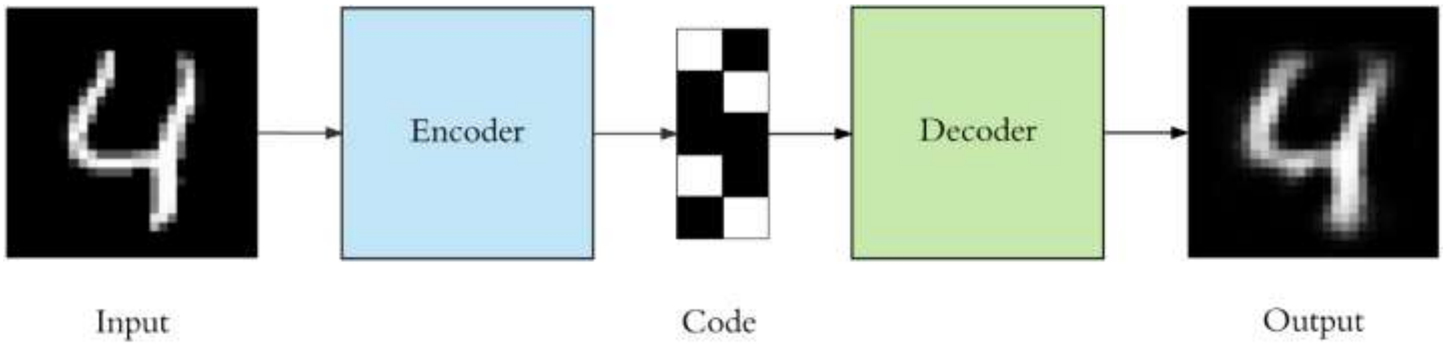


Figure 2.1 Illustrating the function of an autoencoder. In this case, the input image is encoded and then decoded to a same size image, which contains as much information from the original as possible (source: https://towardsdatascience.com/applied-deep-learning).

Autoencoders are, mainly, a dimensionality reduction (or compression) algorithm with a couple of important properties:

1. **Data-specific**: Autoencoders are only able to compress data similar to what they have been trained on. Since they learn features specific for the given training data, we cannot expect an autoencoder trained on handwritten digits to compress landscape photos.
2. **Lossy**: The output of the autoencoder will not be the same as the input; it will be a close, but degraded representation.
3. **Unsupervised**: Autoencoders are considered an unsupervised learning technique since they do not need explicit labels to train on.

Keeping the code layer small (i.e. using less nodes than the size of the input space) forced our autoencoder to learn an intelligent representation of the data. There is another way to force the autoencoder to learn useful features, which is adding random noise to its inputs and making it recover the original noise-free data. This way the autoencoder cannot simply copy the input to its output because the input also contains random noise. We are asking it to subtract the noise and produce the underlying meaningful data. This is called a denoising autoencoder.

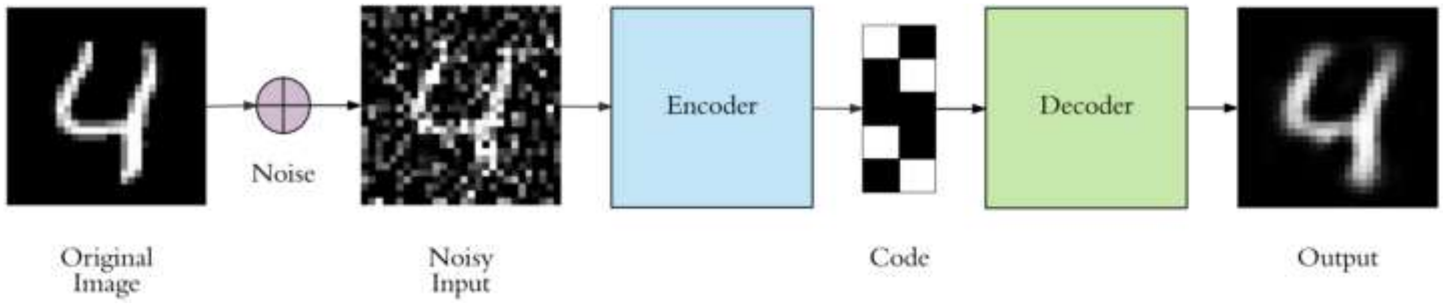


Figure 2.2 Adding white noise prior to the encoder is an alternative approach for the encoder to learn useful features.

The third approach to force the autoencoder to learn useful features is regularization. The sparsity constraint is applied. The constraint states that only a fraction of the nodes would have nonzero values; these are called active nodes.

2.1.2 Stacked Autoencoders

Autoencoders can be stacked to form a deep network by feeding the latent representation (output code) of the denoising autoencoder, found on the previous layer, as input to the current layer. The unsupervised pre-training of such an architecture is done one layer at a time. Each layer is trained as a denoising autoencoder by minimizing the error in reconstructing its input (i.e. the output code of the previous layer). Once the first k layers are trained, we can train the $k+1$-th layer because we can now compute the code or latent representation from the layer below.

Once all layers are pre-trained, the network goes through a second stage of training, called fine-tuning. Here we consider supervised fine-tuning where we want to minimize prediction error on a supervised task. For this, we first add a logistic regression layer on top of the network (more precisely on the output code of the output layer). We then train the entire network as we would train a multilayer perceptron. At this point, we only consider the encoding parts of each auto-encoder. This stage is supervised, since now we use the target class during training.

2.2 Semi-supervised learning

SSL is the machine learning task of inferring a function from labeled and unlabeled data. Thus, SSL falls between unsupervised learning (Makantasis et al., 2013), and supervised learning (Kosmopoulos et al., 2011). The main idea lies in the usefulness of unlabeled data (Protopapadakis et al., 2015); when used, in conjunction with a small amount of labeled data, can considerably improve the model's performance in terms of accuracy, precision, or other related metrics. Nowadays, SSL impact over deep learning models is still investigated (Oliver et al., 2018). In order to make any use of unlabeled data, we must assume some structure to the underlying distribution of data. Sections 2.2.1 and 2.2.2 provide a short description on the topic (Protopapadakis, 2016).

2.2.1 Assumptions

SSL algorithms make use of at least one of the following assumptions:

1. Smoothness assumption: Points close to each other are more likely to share a label. This is also generally assumed in supervised learning and yields a preference for geometrically simple decision boundaries. In the case of semi-supervised learning, the smoothness assumption additionally yields a preference for decision boundaries in low-density regions. In such regions we expect fewer points close to each other, and those few found should belong in different classes (Luo et al., 2018).
2. Cluster assumption: The data tend to form discrete clusters, and points in the same cluster are more likely to share a label (although data sharing a label may be spread across multiple clusters). This is a special case of the smoothness assumption and gives rise to feature learning with clustering algorithms (von Kügelgen et al., 2019).
3. Manifold assumption: The data lie approximately on a manifold of much lower dimension than the input space. In this case, we can attempt to learn the manifold using both the labeled and unlabeled data to avoid the curse of dimensionality. Then learning can proceed using distances and densities defined on the manifold. Manifolds are usually estimated using graphs (Chen et al., 2017).

2.2.2 Regularization

Regularization refers to any process that introduce additional information to solve an ill-posed problem or to prevent overfitting. It usually has the form of a penalty for complexity, such as restrictions for smoothness or bounds on the vector space norm. This section focuses on gasping the regularization notion for each assumption, described in subsection 2.2.1. There are, also, approaches that utilize multiple regularizers (Berthelot et al., 2019).

Smoothness regularization focus on significant changes in f values, for closely located feature vectors $\boldsymbol{x}$. This regularizer penalize any function that does too many jumps especially in dense areas. A typical smoothness functional is the following (Belkin et al., 2004):

$$S(f) = \sum_{i \sim j} \boldsymbol{W}_{ij} \left(f_i - f_j \right)^2 \tag{2.1}$$

In eq. (2.1) the sum is taken over the adjacent vertices of a given graph. For "good" functions f the functional $S(\cdot)$ takes small values. Typically, data close to each other tend to form clusters. Thus, prior to calculate anything, you should locate the decision boundaries (for the clusters), which are forced away from high density regions.

Cluster based approaches assume that the data clusters should have homogeneous labels each. Unlabeled observations are used to identify these clusters. The cluster assumption can be, also, interpreted as the requirement that the decision boundary has to lie in low density regions. This interpretation has been widely used in learning since it can be used in the design of algorithms, e.g. SVMs (Chapelle and Zien, 2004), which are closely related to the above kernel methods. In such cases, a greater penalization is given to decision boundaries that cross a cluster.

Manifold regularization is a different approach; it involves the creation of an appropriate manifold, capturing the local geometry (Fan et al., 2011), and the use of any (manifold) regularizer over it. Manifold creation is achieved by using the data adjacency graph (Luo et al., 2013). A nearest-neighbor-graph, denoted as $\mathcal{W}$, characterizes the data manifold, which explores the geometric structure of the compact support of the marginal distribution. The Laplacian $\mathcal{L}$ of $\mathcal{W}$ and the prediction $\mathbf{f} = [f(x_1), \dots, f(x_n)]$ are then formulated as a smoothness constraint $\|f\|_I^2 = \mathbf{f}^{\mathrm{T}}$. The manifold regularization framework minimizes the regularized loss:

$$\underset{f \in \mathcal{H}_k}{\operatorname{argmin}} \frac{1}{l} \sum_{i=1}^{l} L(f, x_i, y_i) + \gamma_A \|f\|_k^2 + \gamma_I \|f\|_I^2 \tag{2.2}$$

where L is a predefined loss function, k is the standard scalar valued kernel, i.e., $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, and $\mathcal{H}_k$ is the associated reproducing kernel Hilbert space (RKHS). Here, γ_A and γ_I are trade-off parameters to control the complexities of f in the ambient space and the compact support of the marginal distribution. The representer theorem (Belkin et al., 2006) ensures the solution of eq. (2.2) takes the form $f^*(x) = \sum_{i=1}^{n} a_i k(x, x_i)$, where $a_i \in \mathbb{R}$ is a coefficient. A pair of close samples means that the corresponding conditional distributions are similar, so that the manifold regularization $\|f\|_I^2$ helps the function learning.

While many semi-supervised algorithms have been derived from this perspective and many have enjoyed empirical success, there are few theoretical analyses that characterize the class of problems on which manifold regularization approaches are likely to work (Niyogi, 2013). Even when the data might have a manifold structure, it is not clear whether learning the manifold is necessary for good performance (de Sousa et al., 2015).

2.2.3 The importance of feature selection

Feature extraction is a special form of dimensionality reduction, achieved by transforming the input data into the set of distinctive features. This procedure often involves reducing the amount of resources required to describe a large set of data. When performing analysis of complex data one major problem stems from the number of variables involved. Analysis with many variables, generally, requires a large amount of memory and computational power or a classification algorithm, which overfits the training sample and generalizes poorly to new samples (Protopapadakis et al., 2018b). Feature extraction is a general term for methods constructing combinations of the variables to get around these problems, while still describing the data with sufficient accuracy.

In other words, bad features harm the accuracy of the model. The work of (K. Makantasis et al., 2015) demonstrates that low-level features are not sufficient for complex visual recognition tasks. Yet, the selection of good features does not guarantee a good performance. The work of (Protopapadakis et al., 2019a) on credit risk assessment demonstrates the impact of labeled data selection. Although the features utilized were statistically significant, the random selection of few as representative ones, could jeopardize models' performance. Depending on the core SSL method, a different sampling approach could outperform the rest. It is therefore crucial not only the quality of features, but also the selection of the most descriptive data, to serve as a training set.

2.2.4 Handling imbalanced classes

Imbalanced classes are a common case. In many research fields the total number of a class of data is far less than the total number of another class of data. This problem is extremely common in practice and can be observed in various disciplines, e.g. crack recognition, and structural assessment. There are three major approaches in such cases: data resampling, synthetic data, and customized cost functions. There are, also, combinations of the above.

1. Data sampling. There are two main methods that you can use to even-up the classes: a) add copies of instances from the under-represented class called over-sampling (or more formally sampling with replacement), or b) delete instances from the over-represented class, called under-sampling. These approaches are often easy to implement and fast to run.
2. Synthetic data generation: This approach emphasizes on the minority class(es). The main goal is to generate new data that have similar attribute to the data from minority class. This case requires an experienced user, knowledge of the application scenario and does not guarantee that the non-linear relationships between the attributes will be preserved. There are systematic algorithms that you can use to generate synthetic samples. The most popular of such algorithms is called SMOTE or the Synthetic Minority Over-Sampling Technique. As its name suggests, SMOTE is an oversampling method. It works by creating synthetic samples from the minor class instead of creating copies. The algorithm selects two or more similar instances (using a distance measure) and perturbing an instance one attribute at a time by a random amount within the difference to the neighboring instances.
3. Customized cost functions. In this approach data set remains intact. However, penalized classification imposes an additional cost on the model for making classification mistakes on the minority class during training. These penalties can bias the model to pay more attention to the minority class. Using penalization is desirable if you are locked into a specific algorithm and are unable to resample or you are getting poor results.

2.3 Evaluating model's performance

In this thesis, we focus on classification problems, i.e. a type of predictive models, which generate nominal, binary or discrete outputs. However, all the applied techniques produce soft labels (i.e. continuous values), which are later transformed to specific labels. Given on what type of output we use to measure the performance of the model, different evaluation metrics can be applied.

2.3.1 Classification related scores

Confusion matrix is a breakdown of predictions into a table, showing correct predictions (the diagonal) and the types of incorrect predictions made (what classes incorrect predictions were assigned). Then, based on these values we can calculate various performance scores. *Table 2.1* describe some of these scores.

Table 2.1. Metrics for quantitative performance evaluation.

Metric	*Formulation*	*Description*
Sensitivity (TPR)	TPR = TP / P	Also, known as recall; a measure of a classifiers completeness: the fraction of the positive samples that are relevant to the query that are successfully retrieved.
Specificity (SPC)	SPC = TN / N	The probability that the model correctly identifies a non-relevant class.
Precision (PPV)	PPV = TP / (TP + FP)	A measure of a classifier's exactness. In binary classification known as positive predictive value. How many correct positive predictions we have.
Accuracy (ACC)	ACC = (TP + TN) / (P + N)	Percentage of correct classification for ALL classes.
F1-score (F1)	F=2TP/(2TP+FP+FN)	The weighted harmonic mean of precision and recall.

2.3.2 Regression related scores

Statistical errors calculate the sum of differences between actual (simulated) and forecasted (regressor estimated) values. The statistical measurements used were the mean absolute error (MAE) and the root mean square error (RMSE). MAE measures the average magnitude of the errors in a set of predictions, without considering their direction. It is the average over the test sample of the absolute differences between prediction and actual observation where all individual differences have equal weight. RMSE is a quadratic scoring rule that also measures the average magnitude of the error. It is the square root of the average of squared differences between prediction and actual observation.

Both MAE and RMSE express average model prediction error in units of the variable of interest. Both metrics can range from 0 to ∞ and are indifferent to the direction of errors. They are negatively oriented scores, which means lower values are better. Taking the square root of the average squared errors has some interesting implications for RMSE. Since the errors are squared before they are averaged, the RMSE gives a relatively high weight to large errors. This means the RMSE should be more useful when large errors are particularly undesirable. Another implication of the RMSE formula that is not often discussed has to do with sample size. Using MAE, we can put a lower and upper bound on RMSE, so that:

1. MAE≤RMSE. The RMSE result will always be larger or equal to the MAE. If all the errors have the same magnitude, then RMSE=MAE.
2. RMSE≤MAE $\cdot \sqrt{n}$, where n is the number of test samples. The difference between RMSE and MAE is greatest when all the prediction error comes from a single test sample. The squared error then equals to $\mathrm{MAE}^2 \cdot \mathrm{n}$ for that single test sample and 0 for all other samples. Taking the square root, RMSE = MAE $\cdot \sqrt{n}$.

An additional performance score, i.e. coefficient of determination, R^2, is used. R^2 provides a measure of how well observed outcomes are replicated by the model, based on the proportion of total variation of outcomes explained by the model. Values close to 1, i.e. $R^2 \approx 1$, indicate that the regression predictions approximate extremely well real outputs.

2.4 Applied techniques

In this study, we consider a range of SSL algorithms. These algorithms can be modified in various ways, to support the application case. This section provides further details on them.

2.4.1 Graph-based approaches

Graph-based semi-supervised methods define a graph over the entire data set, $\boldsymbol{X} = \boldsymbol{X}_L \cup \boldsymbol{X}_U$, where, $\boldsymbol{X}_L = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), \dots, (\boldsymbol{x}_l, \boldsymbol{y}_l)\}$, is the labeled data set and $\boldsymbol{X}_U = \{\boldsymbol{x}_{l+1}, \dots, \boldsymbol{x}_{l+u}\}$ the unlabeled data set. Feature vectors, $\boldsymbol{x}_i \in \mathbb{R}^m, i = 1, \dots, l+u$, are available for all the observations and $\boldsymbol{y}_i \in \mathbb{R}^C, i = 1, \dots, l$, are the corresponding classes of the labeled ones, in a vector form; C denotes the available classes.

The nodes represent the labeled and unlabeled examples in the dataset; edges reflect the similarity among examples. In order to quantify the edges (i.e. assign a similarity value), an adjacency matrix $\boldsymbol{\mathcal{A}}$ is calculated, where:

$$\mathcal{A}_{ij} = \begin{cases} 1 \; if \; \boldsymbol{x}_i \; close \; to \; \boldsymbol{x}_j \\ 0 \quad otherwise \end{cases} \tag{2.3}$$

Practically, each label is only connected to its k closest labels, so that $\sum_{j=0}^{n} \mathcal{A}_{ij} = k$. The information of the labeled nodes propagates to the unlabeled nodes via paths defined on existing edges, provided by $\boldsymbol{\mathcal{A}}$.

Graph methods are non-parametric, discriminative, and transductive in nature. Intuitively speaking, in a graph that various data points are connected, the greater the similarity, the greater the probability of having similar labels. Thus, the information (of labels) propagates from the labeled points to the unlabeled ones. These methods usually assume label smoothness over the graph. That is, if two instances are connected by a strong edge, their labels tend to be the same.

2.4.1.1 Harmonic functions

An indicative paradigm of graph-based SSL is the harmonic function approach (Zhu, 2003). This approach estimates a function f on the graph which satisfies two conditions. Firstly, f has the same values as given labels on the labeled data, i.e. $f(\boldsymbol{x}_i) = \boldsymbol{y}_i, i = 1, \dots, l$. Secondly, f satisfies the weighted average property on the unlabeled data:

$$f(\boldsymbol{x}_j) = \frac{\sum_{k=1}^{l+u} w_{jk} f(\boldsymbol{x}_j)}{\sum_{k=1}^{l+u} w_{jk}}, j = l+1, \dots, l+u \tag{2.4}$$

where w_{ij} denotes the edge weight. Those two conditions lead to the following problem:

$$\min_{f:f(\boldsymbol{x})\in\mathbb{R}} \sum_{i,j=1}^{l+u} w_{ij} \left(f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j)\right)^2 \tag{2.5}$$

$$s.t. f(\boldsymbol{x}_i) = \boldsymbol{y}_i, i = 1, \dots, l$$

The problem has an explicit solution, allowing a soft label estimation for all the edges of the graph, i.e. investigated cases.

2.4.1.2 Anchor graph

Anchor graph estimates a labeling prediction function $f: \mathbb{R}^m \to \mathbb{R}$ defined on the samples of $\boldsymbol{X}$, by using a subset $\boldsymbol{\mathcal{U}} = \{\boldsymbol{u}_k\}_k^p \subset \boldsymbol{X}_L$ of the labeled data, the label prediction function can be expressed as a convex combination (Liu et al., 2010):

$$f(\boldsymbol{x}_i) = \sum_{k=1}^{p} Z_{ik} \cdot g(\boldsymbol{u}_k) \tag{2.6}$$

where Z_{ik} denotes sample-adaptive weights, which must satisfy the constraints $\sum_{k=1}^{l} Z_{ik} = 1$ and $Z_{ik} \geq 0$ (convex combination constraints). By defining vectors $\boldsymbol{g}$ and $\boldsymbol{a}$ respectively as $\boldsymbol{g} = [g(\boldsymbol{f}_1), \dots, g(\boldsymbol{f}_n)]^{\boldsymbol{T}}$ and $\boldsymbol{a} = [g(\boldsymbol{x}_1), \dots, g(\boldsymbol{x}_p)]^{\boldsymbol{T}}$ eq. (2.6) can be rewritten as $\boldsymbol{g} = \boldsymbol{Z\alpha}$ where $\boldsymbol{Z} \in \mathbb{R}^{n\times p}$.

The designing of matrix $\boldsymbol{Z}$, which measures the underlying relationship between the samples of $\boldsymbol{X}_U$ and samples $\boldsymbol{X}_L$, is based on weights optimization, i.e. non-parametric regression. Thus, the reconstruction for any data point is a convex combination of its closest representative samples.

Nevertheless, the creation of matrix $\boldsymbol{Z}$ is not sufficient for labeling the entire data set, as it does not assure a smooth function $\boldsymbol{g}$. Despite the small labeled set, there is always the possibility of inconsistencies in segmentation; different companies with almost identical attributes are classified differently. In order to deal with such cases, the following SSL framework is employed:

$$\min_{\boldsymbol{A}=[\boldsymbol{a}_1,\dots,\boldsymbol{a}_c]} \mathcal{Q}(\boldsymbol{A}) = \frac{1}{2}\|\boldsymbol{ZA} - \boldsymbol{Y}\|_F^2 + \frac{\gamma}{2}\operatorname{trace}(\boldsymbol{A}^{\boldsymbol{T}}\hat{\boldsymbol{L}}\boldsymbol{A}) \tag{2.7}$$

where $\hat{\boldsymbol{L}} = \boldsymbol{Z}^T\boldsymbol{LZ}$ is a memory-wise and computationally tractable alternative of the Laplacian matrix $\boldsymbol{L}$. The matrix $\boldsymbol{A} = [\boldsymbol{a}_1, \dots, \boldsymbol{a}_c] \in \mathbb{R}^{p\times c}$ is the soft label matrix for the representative samples, in which each column vector accounts for a class. The matrix $\boldsymbol{Y} = [\boldsymbol{y}_1, \dots, \boldsymbol{y}_c] \in \mathbb{R}^{n\times c}$ is a class indicator matrix on ambiguously labeled samples with $Y_{ij} = 1$ if the label l_i of sample i is equal to j and $Y_{ij} = 0$ otherwise.

The Laplacian matrix $\boldsymbol{L}$, is calculated as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$, where $\boldsymbol{D} \in \mathbb{R}^{n\times n}$ is a diagonal degree matrix and $\boldsymbol{W}$ is approximated as $\boldsymbol{W} = \boldsymbol{Z\Lambda}^{-1}\boldsymbol{Z}^T$. Matrix $\boldsymbol{\Lambda} \in \mathbb{R}^{p\times p}$ is defined as: $\boldsymbol{\Lambda} = \sum_{i=1}^{n} Z_{ik}$. The solution of the eq. (2.7) has the form of:

$$\boldsymbol{A}^* = (\boldsymbol{Z}^T\boldsymbol{Z} + \gamma\hat{\boldsymbol{L}})\boldsymbol{Z}^T\boldsymbol{Y} \tag{2.8}$$

Each sample label is, then, given by:

$$\hat{l}_i = \arg\max_{j\in\{1,\dots,c\}} \frac{\boldsymbol{Z}_i a_j}{\lambda_j} \tag{2.9}$$

where $\boldsymbol{Z}_i \in \mathbb{R}^{1\times p}$ denotes the i-th row of $\boldsymbol{Z}$, and factor $\lambda_j = \mathbf{1}^T\, \boldsymbol{Z}\, \boldsymbol{\alpha}_j$ balances skewed class distributions.

2.4.2 Low density separation

The low-density separation (LDS) assumption pushes the decision boundary in regions with few data points (labeled or unlabeled). The most common approach to achieving this goal is to use a maximum margin algorithm such as support vector machines. The method of maximizing the margin for unlabeled as well as labeled points is called the transductive SVM (TSVM). However, the corresponding problem is non-convex and thus difficult to solve (Singla et al., 2014).

Low density separation (LDS) is a combination of TSVMs (Bruzzone et al., 2006), trained using gradient descend, and traditional SVMs using an appropriate kernel defined over a graph using SSL assumptions (Chapelle and Zien, 2004). Like the SVM approach, the TSVM maximizes the class separating margin. The problem can be stated in the following form, which allows for a standard gradient-based approach:

$$\min_{\boldsymbol{w},b}\left[\frac{1}{2}\boldsymbol{w}^2 + C\sum_{i=1}^{l} L^2\big(y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b)\big) + C^*\sum_{j=l+1}^{l+u} L^*\big(\left|\boldsymbol{w}^T\boldsymbol{x}_j + b\right|\big)\right] \tag{2.10}$$

where $\boldsymbol{w} \in \mathbb{R}^n$is the parameter vector that specifies the orientation and scale of the decision boundary and $b \in \mathbb{R}$ is an offset parameter. Eq. (2.10) exploits both labeled, $\boldsymbol{X}_L$, and unlabeled $\boldsymbol{X}_U$ data. Finally, let us denote as $L(t) = \max(0,1-t)$ and $L^*(t) = \exp(-3t^2)$. Such a formulation allows the use of a non-linear kernel, calculated over a fully connected matrix, $\boldsymbol{W}$, which is formed as $w_{ij} = \exp\big(\rho - dist(i,j)\big) - 1$. Dijkstra's algorithm is employed to compute the shortest path lengths, $d_{SP}(i,j)$ for all pairs of points. The matrix $\boldsymbol{\mathcal{D}}$ of squared ρ -path distances is calculated as:

$$\mathcal{D}_{ij} = \left(\frac{1}{\rho}\log\big(1 + d_{SP}(i,j)\big)\right)^2 \tag{2.11}$$

The final step towards the kernel's creation, involves multidimensional scaling (Cox and Cox, 2008), or MDS, to find a Euclidean embedding of $\boldsymbol{\mathcal{D}}^\rho$(in order to obtain a positive definite kernel). The embedding found by the classical MDS are the eigenvectors corresponding to the positive eigenvalues $\boldsymbol{U\Lambda U}^T = -\boldsymbol{H\mathcal{D}}^\rho\boldsymbol{H}$, where $H_{ij} = \delta_{ij} - \frac{1}{l+u}$. The final representation of $\boldsymbol{x}_i$ is $x_{ik} = U_{ik}\sqrt{\lambda_k}$, $1 \le k \le p$.

2.4.3 Maximal performance gain

Assume a set of b semi-supervised regressors (SSRs) applied over the same unlabeled data set $\boldsymbol{X}_U$. The outcome would be b predictions, i.e. $\{\boldsymbol{f}_1, \dots, \boldsymbol{f}_b\}$, where $\boldsymbol{f}_i = \{f(\boldsymbol{x}_{l+1}), \dots, f(\boldsymbol{x}_{l+\mathrm{u}})\}$, $i = 1, \dots, b$. Let as, also, denote as $\boldsymbol{f}_0$ the predictions over $\boldsymbol{X}_U$, of a traditional supervised approach. If we know the weight of the individual regressors, $\boldsymbol{\alpha} = [a_o, \dots, a_b]$, $a_i \ge 0$, the problem formulation has the form:

$$\max_{\mathbf{f}} \sum_{i=0}^{b} a_i(\|\,\boldsymbol{f}_0 - \boldsymbol{f}_i\|^2 - \|\boldsymbol{f} - \boldsymbol{f}_i\|^2) \tag{2.12}$$

However, since weights are not known a priori, certain assumptions have to be made. At first, we state that $\boldsymbol{a}$ is from a convex linear set $\mathcal{M} = \{a|\boldsymbol{A}^T\boldsymbol{a} \le \boldsymbol{b}, \boldsymbol{a} \ge 0\}$, where $\boldsymbol{A}$ and $\boldsymbol{b}$ are task-dependent coefficients. Without further knowledge to determine the weights of individual regressors, one aim to optimize the worst-case performance gain:

$$\max_{\mathbf{f}} \min_{\boldsymbol{a}\in\mathcal{M}} \sum_{i=0}^{b} a_i(\|\,\boldsymbol{f}_0 - \boldsymbol{f}_i\|^2 - \|\boldsymbol{f} - \boldsymbol{f}_i\|^2) \tag{2.13}$$

The equation above is concave to f and convex to a and thus it is recognized as saddle-point convex-concave optimization (Nesterov, 2013). As described in recent work (Li et al., 2017) equation (2.13) can be formulated as a geometric projection problem, handling that way the computational load. Specifically, by setting the derivative of equation (2.13) to zero, we get a close form solution w.r.t. $\boldsymbol{f}$ as $\boldsymbol{f} = \sum_{i=1}^{b} a_i \boldsymbol{f}_i$, which we can substitute in equation (2.13), obtaining an expression that is only related to $\boldsymbol{\alpha}$: $\min_{a \in \mathcal{M}} \left\| \sum_{i=1}^{b} a_i \boldsymbol{f}_i - \boldsymbol{f}_0 \right\|$. As such, we have we have a convex quadratic problem.

2.4.4 Squared-loss mutual information

An alternative approach to LDS is the information maximization principle (IMP). In IMP a probabilistic classifier is trained in an unsupervised manner, so that a given information measure between data and cluster assignments is maximized. The proposed squared-loss mutual information regularization (SMIR) model (Niu et al., 2013) is convex (under mild conditions) and results in globally optimal solution.

Let $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y} = \{1, \dots, c\}$, where $d, c \in Z^+$. Any pair $(\boldsymbol{x}, y) \in \mathcal{X} \times \mathcal{Y}$ has an underlying $p(\boldsymbol{x}|y)$ and $p(\boldsymbol{x}) > 0$. If we can estimate $p(y|\boldsymbol{x})$, any $x \in \mathcal{X}$ can be mapped to a $\hat{y}$ as: $\hat{y} = \arg\max_{y \in \mathcal{Y}} p(y|\boldsymbol{x})$. The described SMIR approach approximates the class-posterior probability $p(y|\boldsymbol{x})$. Assuming a uniform class-prior probability $p(y) = 1/c$ the SMI has the form:

$$SMI = \frac{c}{2} \int_{\mathcal{X}} \sum_{y \in \mathcal{Y}} \left(p(y|\boldsymbol{x})\right)^2 p(x) dx - \frac{1}{2} \tag{2.14}$$

Then, by adopting an empirical kernel map:

$$\Phi_n : \mathcal{X} \to \mathbb{R}^n, x \to \left(k(\boldsymbol{x}, \boldsymbol{x}_1), \dots, k(\boldsymbol{x}, \boldsymbol{x}_n)\right)^T \tag{2.15}$$

the class-posterior probability $p(y|\boldsymbol{x})$ can be approximated by:

$$q(y|\boldsymbol{x}; \boldsymbol{a}) = \langle \boldsymbol{K}^{-1/2} \boldsymbol{\Phi}_n(\boldsymbol{x}), \boldsymbol{D}^{-1/2} \boldsymbol{a}_y \rangle \tag{2.16}$$

Where $\boldsymbol{a} = \{\boldsymbol{a}_1, \dots, \boldsymbol{a}_c\}$ are model parameters, $\boldsymbol{K} \in \mathbb{R}^{n \times n}$ is the kernel matrix, $\langle \cdot \rangle$ is the inner product, and $\boldsymbol{D}$ is a degree matrix. Equation (2.14) can be written as:

$$\widehat{SMI} = \frac{c}{2n} tr\left(\boldsymbol{A}^T \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{K} \boldsymbol{D}^{-\frac{1}{2}} \boldsymbol{A}\right) - \frac{1}{2} \tag{2.17}$$

where $\boldsymbol{A} \in \mathbb{R}^{n \times c}$ is the matrix representation of model's parameters. Equation (2.15) regularizes a loss function $\Delta(p, q)$ that is convex w.r.t. q. There are three objectives: (i) minimize $\Delta(p, q)$, (ii) maximize $\widehat{SMI}$ and (iii) regularize $\boldsymbol{\alpha}$. The SMIR optimization problem is formulated as:

$$\min_{\boldsymbol{a}_1, \dots, \boldsymbol{a}_c \in \mathbb{R}^n} \quad \Delta(p, q) - \gamma \widehat{SMI} + \lambda \sum_{y} \frac{1}{2} \left\| \boldsymbol{\alpha}_y \right\|_2^2 \tag{2.18}$$

where $\gamma, \lambda > 0$ are regularization parameters. If the kernel function $\boldsymbol{k}$ is nonnegative and $\lambda > \frac{\gamma c}{n}$, equation (2.14) is convex.

Chapter III: The sampling impact

If we knew what (…) we were doing, it would not be called research (…).
Albert Einstein, theoretical physicist

3 Data Sampling for Semi-Supervised Learning in Vision-Based Concrete Defect Recognition

Appropriate selection of labeled data for semi-supervised learning techniques can be of great significance, for the attained efficacy of related approaches in pattern recognition problems. In this chapter, we scrutinize the effectiveness of different data sampling approaches, as labeled data generators to be used in semi-supervised learning. The adopted sampling approaches are based on techniques like the Kennard-Stone algorithm, sparse representative modeling selection, OPTICS algorithm, k-means centroids, and random selection. These approaches are explored in the context of three semi-supervised learning techniques, i.e. graph-based approaches (harmonic functions, anchor graph) and low-density separation, and evaluated in a real-world scenario of defect detection and recognition on concrete tunnel surfaces.

3.1 Sampling Approaches for Labeled Data Generation

In this Section, we present seven (7) data sampling approaches, which are based on the combination or adaptation of four (4) main known sampling techniques. These techniques are used for the creation of a labeled dataset. The idea is straightforward. At first, we gather all the available data instances (all of them unlabeled). Then sampling approaches provide a reduced amount of data that will be labeled by an expert. Finally, these data are used to train the algorithms / estimate the labels (categories) on the remaining ones. The same approach has been adopted, as in (E. Protopapadakis et al., 2017).

3.1.1 Main sampling techniques

The main purpose of data sampling is the selection of appropriate representative samples to provide a good training set and, thus, improve the classification performance of predictive models. The most important factor in data selection is the definition of distance function. For any two given data points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, $\boldsymbol{x} \in \mathbb{R}^{\boldsymbol{m}}$ let $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ denote the distance between them. Let $\boldsymbol{A} \in \mathbb{R}^{m \times m}$ be a symmetric matrix. Then, the distance measure is defined as:

$$d_{\boldsymbol{A}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{A} (\boldsymbol{x}_i - \boldsymbol{x}_j)} \tag{3.1}$$

Most of the proposed approaches are based on the Euclidean distance (i.e. $\boldsymbol{A} = \boldsymbol{I}$). Sampling algorithms are used over the entire data set $\mathcal{X}$ and create a new set, $\mathcal{X}_r \subset \mathcal{X}$, of the most representative cases. In this study, we need at least one observation from every possible class.

3.1.1.1 OPTICS algorithm

Ordering Points to Identify the Clustering Structure (OPTICS) is an algorithm for finding density-based clusters in spatial data (Ankerst et al., 1999); i.e. detect meaningful clusters in data of varying density. In order to do so, the points of the database are (linearly) ordered such that points which are spatially closest become neighbors in the ordering. OPTICS requires two parameters: the maximum distance (radius) to consider (ε), and the number of points required to form a cluster (*MinPts*) . A point p is a core point if at least *MinPts* points are found within its ε –neighborhood, $N_\varepsilon(p)$. Once the initial clustering is formed, we may proceed with any sampling approach (e.g. random selection among clusters).

3.1.1.2 k-means algorithm

k-means clustering (Protopapadakis et al., 2018c) aims to partition *n* observations into *k* clusters, such that each observation is assigned to the cluster it is most similar to (with the cluster centroid serving as a prototype of the cluster). It is a classical approach that can be implemented in many ways and for various distance metrics. The main drawback is that the number of clusters should be known a priori.

3.1.1.3 Sparse representative selection

Sparse modeling representative selection (SMRS) focuses on the identification of representative objects., through the solution of the following optimization problem (Elhamifar et al., 2012):

$$\min \lambda \|\boldsymbol{C}\|_{1,q} + \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{C}\|_F^2 \quad s.t. \, \mathbf{1}^T\boldsymbol{C} = \mathbf{1}^T \tag{3.2}$$

where $\boldsymbol{X}$ and $\boldsymbol{C}$ refer to data points and coefficient matrix respectively. This optimization problem can also be viewed as a compression scheme, where we want to choose a few representatives that can reconstruct the available data set.

3.1.1.4 Kennard–Stone algorithm

The classic KenStone algorithm is a uniform mapping algorithm; it yields a flat distribution of the data. It is a sequential method that uniformly covers the experimental region. The procedure consists of selecting, as the next sample (candidate object), the one that is most distant from those already selected (calibration objects). For initialization, one can select either the two observations that are most distant from each other, or preferably, the one closest to the mean.

From all the candidate points, the one is selected that is furthest from those already selected and added to the set of calibration points. To do this, we measure the distance from each candidate point $\boldsymbol{x}_0$ to each point $\boldsymbol{x}$, which has already been selected and determine which is smallest, i.e. $\min_i d(\boldsymbol{x}, \boldsymbol{x}_0)$. From these we select the one for which the distance is maximal:

$$d_{selected} = \max_{\mathrm{i}_0}\left(\min_i d(\boldsymbol{x}, \boldsymbol{x}_0)\right) \tag{3.3}$$

3.1.2 Hybrid sampling approaches

The primary goal of sampling approaches is the removal of redundant and uninformative data. The seven (7) sampling approaches were built through the algorithms described earlier in Sec. II.A. A brief introduction of each one follows:

- *OPTICS extrema*: After employing the OPTICS algorithm on the entire data set, the calculated reachability distances are plotted in the same order as data were processed. Over the generated waveform, we locate local maxima and minima. All the identified extrema cases are considered as labeled instances and the rest as unlabeled. This approach results in a limited training set.
- *Sparse modeling representative selection (SMRS)*: The SMRS technique is employed over the entire data set, resulting in a very limited training set, although larger than the one obtained with OPTICS. In contrast to OPTICS, the selected points are located only on the exterior cell of the available data volume. The algorithm has the same implementation as in (Elhamifar et al., 2012).
- *Combination of k-means and SMRS (k-means SMRS)*: We first divide the set into k sub-clusters. For each sub-cluster, we run the SMRS algorithm to get the representative samples among each sub-cluster. As such, the outcome provides points surrounding each sub-cluster. The number of clusters, k, was defined using the rule: $k = \left\lceil\sqrt{n/2}\right\rceil$, where n denotes the number of available data instances.
- *Combination of OPTICS and SMRS (OPTICS SMRS)*: SMRS is performed to the sub-clusters obtained through the OPTICS algorithm. This approach is similar to the work of (Protopapadakis et al., 2014). It creates a small subset of

representative samples from each sub-cluster formed through the OPTICS algorithm. The minimum number of data within a cluster, required by OPTICS, was defined as: $MinPts = \lfloor n/k \rfloor$.

- *Kennard and Stone (KenStone) sampling data points*: After executing the KenStone algorithm, we have data entries spanning uniformly the entire data space. The code was provided by (Daszykowski et al., 2002a).
- *Random selection*: A random selection that picks 10% of the available data as training data.
- *Improved random selection*: An alternative approach is the creation of k clusters (using k-means) and a random selection of n_k samples from each cluster (k-means random). It is an improvement of random selection, without involving any advanced techniques. Similar instances are likely to be clustered together. Thus, the few random samples from each cluster are expected to provide adequate information, over the data set.

3.2 Experimental Evaluation

The described approaches have been evaluated in a real-world tunnel inspection scenario, where images acquired by a robot inside a tunnel of Egnatia Motorway in Greece were used for detecting and recognizing defects on the concrete surfaces (Protopapadakis et al., 2019b). Raw captured tunnel and annotated ground truth images of resolution 600 × 900 pixels were provided. During image acquisition, no special setup took place, i.e. images are taken from any angle and distance from the tunnel surface. In this experiment, holdout cross validation was employed.

3.2.1 Tunnel surface images' feature extraction

To represent each pixel, we use low-level feature extraction techniques; in particular, each pixel p_{xy} is described by a feature vector $f_{xy} = \left[f_{1,xy}, f_{2,xy}, \dots, f_{k,xy}\right]^T$, where $f_{i,xy}$ are scalars corresponding to the presence and magnitude of the low-level features detected at location (x, y). Feature vectors along with the class labels of every pixel are used to form a data set. There are 5 different classes of defects: (1) crack, (2) staining, (3) spalling, (4) delamination, and (5) calcium leaching. We hereby briefly describe the features used to form vector f_{xy}.

1. **Edges.** To successfully exploit image edges, the system must be able to detect them in a very accurate way and it must preserve their magnitude. We therefore combined the Canny and Sobel operators. Canny operator is an image edge detector, which outputs zeros and ones for image edges absence and presence respectively. Sobel operator measures the strength of detected edges. Multiplying pixel-wise the output of two operators the system can detect edges, while at the same time it preserves their magnitude.
2. **Frequency.** Frequency feature is utilized to emphasize regions of high frequency in the image and at the same time suppress low frequency regions. Frequency components of an image I are computed as: $\mathcal{F}_I = \nabla^2 I$.
3. **Entropy.** Image entropy quantifies the information coded in an image, i.e. the amount of information, which can be coded by a compression algorithm. Images that depict large homogeneous regions, present low entropy, while highly textured images will present high entropy.
4. **Histogram of Oriented Gradients (HOG)**. HOG is a popular dense feature descriptor used for the task of object detection. It exploits image gradients to capture contour, silhouette, and texture information, in order to produce an encoding that is sensitive to local image content while remaining resistant to small changes in pose or appearance.
5. **Texture**. For texture identification, we used the Gabor filter. A Gabor filter is characterized by a frequency and an orientation. Frequency and orientation representations of Gabor filters are similar to those of the human visual system, and they have been found to be particularly appropriate for texture representation and discrimination. In the spatial domain, a 2D Gabor filter is a Gaussian kernel function modulated by a sinusoidal plane wave. In our implementation, we used Gabor filters with different frequencies and orientations. Specifically, we used twelve Gabor filters with orientations 0°, 30°, 60° and 90°, and frequencies 0.0, 0.1, and 0.4.

The combination of these features with the raw pixels' intensity result in a 1 × 17 feature vector, f_{xy}, containing visual information that characterizes each one of the image pixels.

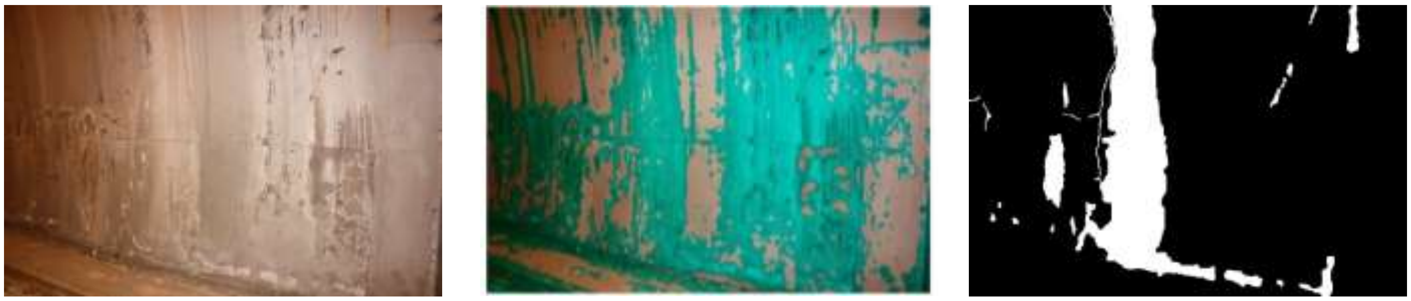


Figure 3.1. (left) Original image from tunnel surface; (center) LDS annotation; (right) Ground truth

3.2.2 Results

Each of the seven data sampling approaches described in Section 3.1 was evaluated with each of the three SSL techniques presented in section 2.4, resulting in 21 combinations in total. The results in terms of averaged F-measure for each combination are depicted in Figure 3.2. It appears that in SSL techniques the harmonic functions tend to provide higher F1-scores, whereas concerning data sampling approaches, KenStone and Random seem to give overall best results. Figure 3.3 provides an example of confusion matrices acquired for different sampler/classifier combinations.

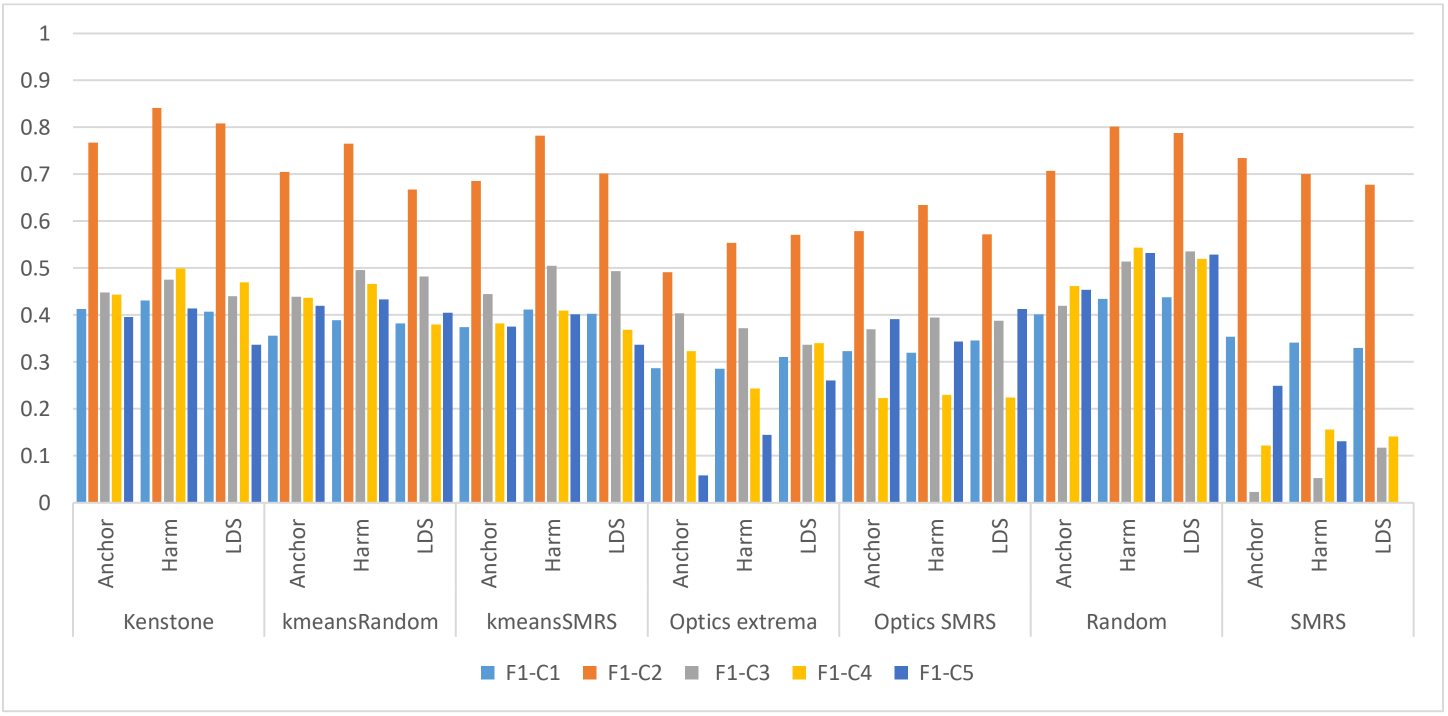


Figure 3.2. F1-score values for various combinations between SSL classifer and sampling approach.

To obtain further insights into the results and the relative performance of the different algorithms, we conducted an analysis of variance (ANOVA), on the F1-scores for the test samples. ANOVA enables the statistical assessment of the effects that the two main design factors of this analysis have (i.e., the sampling schemes and the SSL techniques). As shown in Table 3.1.

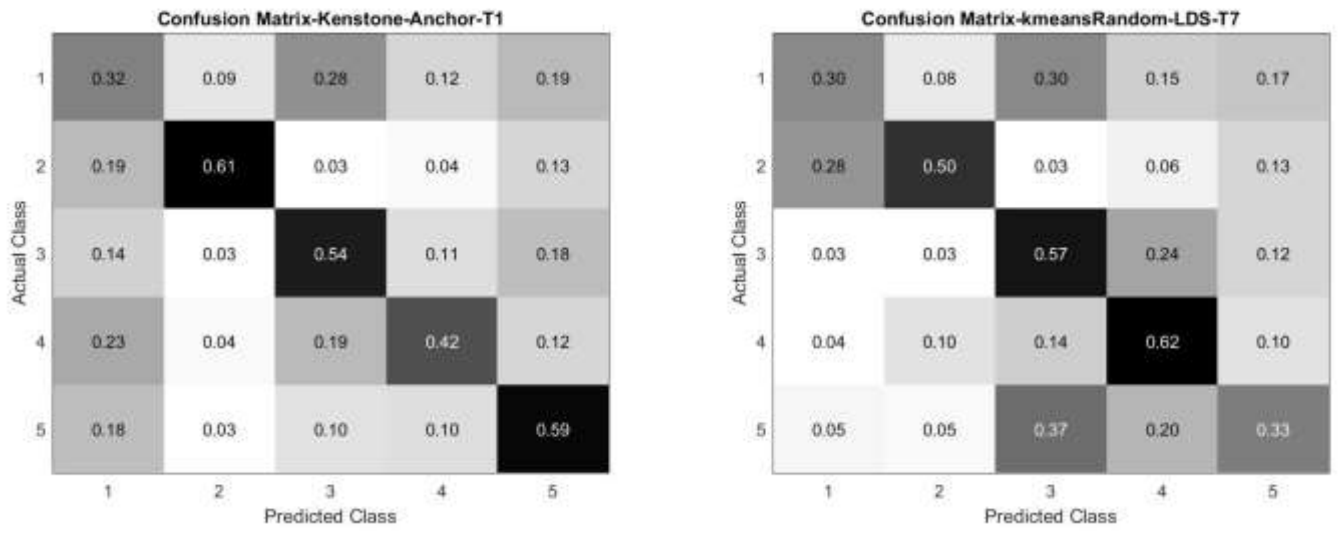


Figure 3.3. Illustration of confusion matrices for: (left) Kenstone sampling Anchor graph combination for holdout set T1, (right) kmeans/Random sampling LDS combination for holdout set T7.

Table 3.1, both the sampling scheme and the choice of classifier are strongly significant for explaining variations in AUC. However, their intra sampling-classifiers interaction does not appear to be significant.

Table 3.1. Statistical significance of sampling scheme and SSL classifier.

Source	*Sum Sq.*	*d.f.*	*Mean Sq.*	*F*	*Prob>F*
Sampling	1.47232	6	0.24539	96.15	0
Classifier	0.02305	2	0.01153	4.52	0.0125
*Sampling * Classifier*	0.04422	12	0.00368	1.44	0.1524
Error	0.37516	147	0.00255		
Total	1.91476	167			

In addition to the above basic ANOVA results, we use the Tukey honest significant difference (HSD) post-hoc test to identify sampling schemes and classifiers that provide the best results, while considering the statistical significance of the differences between the results. Figure 3.4 illustrates the results for the SSL techniques (a) and the sampling schemes (b). It is evident that using random sets leads to results that are better than other sampling schemes in a statistically significant way.

Among the remaining sampling approaches, the Kennard-Stone algorithm provides the best results outperforming the other approaches. On the other end, SMRS provides results significantly worse than all competing schemes. As far as classifiers are concerned, harmonic functions appear to have a statistically significant superiority over anchor graph, but not over LDS, although the mean performance is higher. Regarding the combined sampling-SSL evaluation, the (random, harmonic) combination has a statistically different result when compared to 13 (out of the remaining 20) combinations, and the same goes for the (random, LDS) combination.

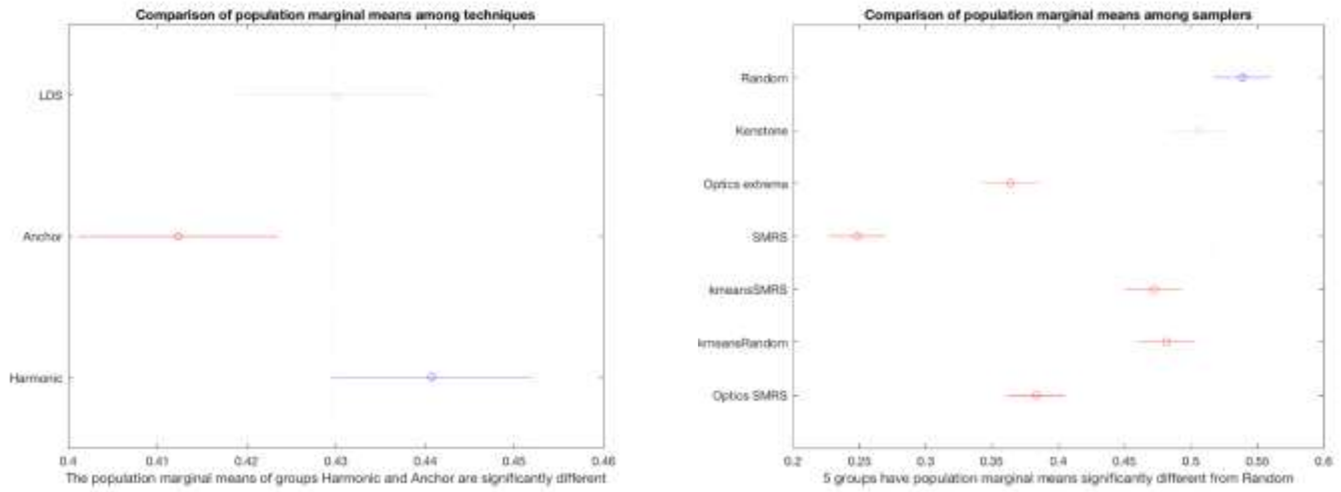


Figure 3.4. Comparison of population marginal means among (a) SSL techniques and (b) sampling schemes.

3.3 Concluding remarks

In this chapter, we explored the effectiveness of different data sampling approaches for labeled data generation. Selected instances were used with SSL models in the context of visual classification problems, emphasizing on concrete surface defect recognition scenario. We compared seven sampling approaches that were based on OPTICS, k-means, SMRS and Kenstone algorithm, juxtaposed to three SSL techniques, anchor graph, harmonic function, and low-density separation. Results indicate that harmonic function appear to has a statistically significant superiority over anchor graph, but not for LDS, although the mean performance is higher. In terms of sampling schemes, random selection appears to provide a better final F1-score compared to the other explored data selection approaches. However, the number of selected instances is a crucial factor and needs to be further investigated (Protopapadakis et al., 2018a).

Chapter IV: Semi- supervised multi-objective topology optimization

Failure is simply the opportunity to begin again, this time more intelligently.
Henry Ford, American automobile manufacturer

4 Semi supervised multi-objective optimization on artificial neural networks topology

In this chapter we investigate the applicability of SSL related assumptions in ANNs topology optimization. Determining the appropriate network topology for a feedforward neural classifier is an interesting problem that is usually addressed through evaluation against given performance metrics, such as accuracy, and F1-score. However, there is limited work based on SSL fitness evaluation. In this case, we consider three more evaluation metrics inspired by SSL field, i.e. cluster performance, smoothness performance, and graph-based metrics. Leveraging an island genetic algorithm, we investigate the applicability and efficacy of the derived multi-objective optimization strategy for defining the parameters of a feedforward neural network. The described approach is evaluated in the context of a flower classification task, but can be applied in a vast range of pattern analysis problems (Eftychios Protopapadakis et al., 2017b).

4.1 Introduction

Multiple SSL assumptions combination is something common (Chen and Wang, 2011). The combination of multiple regularizers appears to improve the model's correctness, e.g. accuracy rates (Protopapadakis et al., 2020). However, to our knowledge, there is no work over the trade-offs between the various assumptions, if any. In other words, we need to investigate to which extent the applied regularizers and the loss function can be optimized simultaneously. Thus, we deal with mathematical optimization problems involving more than one objective function, i.e. multiple regularizers.

The main difficulty in considering multi-objective optimization is that there is no accepted definition of optimum in this case, and therefore it is difficult to compare one solution with another one. In general, these problems may accept multiple solutions, each of which is considered acceptable and equivalent when the relative importance of the objectives is unknown. The best solution is subjective and depends on the need of the designer or decision maker (Eftychios Protopapadakis et al., 2017b).

In the literature, there can be found works on multi-objective optimization. For instance, the work of (Alok et al., 2014) utilizes multi-objective optimization for feature selection in semi-supervised clustering. The approach utilizes four objective functions. The first two objective functions represent, respectively, total symmetry present in the clusters and total compactness of the partitioning results. The third objective function is an external cluster validity index which measures the similarity of the clustering obtained on labeled data with the original labeling, and the fourth one counts numbers of features. Their objective is to optimize values of cluster validity indices, to remove the bias of internal cluster validity indices on lower dimensions. The core optimization algorithm is simulated annealing (Talbi, 2009).

The work of (Kobayashi et al., 2012) proposes a language model (LM), obtained from manually transcribed data. The LM is implemented as a log-linear model, which employs a set of linguistic features derived from word or phoneme n-grams. The concept employs objective functions that are designed for minimizing expected risks associated with word error rate, and its optimum solution leads to a discriminatively trained model. The objective functions for labeled or unlabeled training lattices are derived from the Bayes risk. The ε-constraint method (Talbi, 2009) is employed to solve a MOP problem. The work of (Cheng et al., 2012) proposes a particle swarm optimization based semi-learning classifier to solve Chinese text categorization problem. This classifier utilizes an iterative strategy, and the result of the classifier is determined by a document's previous prediction and its neighbors' information.

4.2 Multi-objective optimization

The multi-objective optimization problem (MOP) can be stated as follows (Coello et al., 2007; Deb, 2001):

$$\begin{gathered} \min \ \boldsymbol{f}(\boldsymbol{x}) \\ \text{s.t. } \boldsymbol{x} \in \boldsymbol{X} \end{gathered} \tag{4.1}$$

where $\boldsymbol{f}(\boldsymbol{x}) = \left(f_1(\boldsymbol{x}), \dots, f_k(\boldsymbol{x})\right)$, denotes the solution $\boldsymbol{x}$ scores in each objective function $f_j(\boldsymbol{x})$ for $j = 1, \dots, k$ and $k \geq 2$. In MOP, there does not exist a feasible solution that minimizes all objective functions simultaneously. Therefore, attention is paid to Pareto optimal solutions; that is, solutions that cannot be improved in any of the objectives without degrading at least one of the other objectives. An individual's performance is defined by a vector $\boldsymbol{\mathcal{P}} = [\mathcal{P}_{acc}\mathcal{P}_{prec}\mathcal{P}_{rec}\mathcal{P}_{F1}\ \mathcal{P}_C\ \mathcal{P}_S\mathcal{P}_M]$, where each $\mathcal{P}_i$ denotes respectively the model's performance in terms of: accuracy, precision, recall, F1-score (i.e. four traditional performance metrics), followed by cluster-based, smoothness-based, and graph-based (manifold) performance. In order to be able to compare two or more individuals, we introduce the domination term:

Definition 1*:* Individual I_i dominates individual I_j if $\boldsymbol{\mathcal{P}}_k^{(i)} \geq \boldsymbol{\mathcal{P}}_k^{(j)}, k = 1 \dots m$ and $\boldsymbol{\mathcal{P}}_k^{(i)} > \boldsymbol{\mathcal{P}}_k^{(j)}$ for at least one k. Otherwise, both solutions are considered incomparable.

4.2.1 Traditional performance metrics

Our case is built around typical multiclass classification problems, as is usually the case in visual pattern analysis. The results are analyzed through standard measures of predictive performance over confusion matrix $\mathcal{C}$. Confusion matrix is a $m \times m$ matrix, where m is the number of available classes. The value $\mathcal{C}_{ij}$ denotes the number of instances that belong to class i but are classified as class j. The ideal case would be a diagonal matrix. Using these elements, four well-known classification performance indices are calculated: Accuracy, precision, recall and F1-score (as in sec. 2.3.1).

We further calculate three more evaluation metrics, inspired by the SSL field; cluster performance, smoothness performance, and graph-based metrics; all defined over label and unlabeled data set. In order to calculate these metrics, we need a FFNN defined according the original training set.

4.2.2 Cluster based fitness evaluation

The cluster performance metric is defined over SSL cluster assumption; points belonging to the same cluster will probably have the same label. First, we use a simple clustering algorithm over the entire data set. Then, for each of the generated clusters we predict the labels over the unlabeled data, X_U, according to the available FFNN. Finally, we calculate the percentage of the dominant label index in each cluster according to the following equation:

$$\mathcal{P}_c = \sum_{i=1}^{K} C\,p_i \tag{4.2}$$

where K denotes the number of clusters and Cp_i is a cluster percentage, defined over i-th cluster as:

$$Cp_i = \frac{1}{N_i} \max_{\forall n} \left\{ \sum_{j=1}^{N_i} t_j^n \right\} \tag{4.3}$$

where N_i is the cardinality of cluster i and $t_j^n \in \{0,1\}$ a value defined as:

$$t_j^n = \begin{cases} 1 \; iff \; y_j = n \\ 0 \; otherwise \end{cases} \tag{4.4}$$

For each cluster, Cp_i is a bounded quantity, i.e. $0 < Cp_i \leq 1$, that indicates the FFNN classifier performance over the instances of the same cluster. Intuitively, clusters should have values of $Cp_i \approx 1$, since the cluster members likely share the same label. Thus, a NN classifier assigning similar labels to members of the same cluster will be a good classifier. According to the above formulation, $\mathcal{P}_c$ is also bounded: $0 < \mathcal{P}_c \leq K$; a score of $\mathcal{P}_c = K$ is the ideal case, which is not expected in real data sets due to presence of outliers.

Equation (4.2) does not penalize cases when most cluster members does not agree with the labeled examples, in the same cluster. Thus, we involve a cluster disagreement index, Cd_i, defined as:

$$Cd_i = \begin{cases} 1 \; if \; \mathcal{C}_i^{(L)} > \mathcal{C}_i^{(U)} \\ 0 \; otherwise \end{cases} \tag{4.5}$$

$\mathcal{C}_i^{(L)}$and $\mathcal{C}_i^{(U)}$indicate the cardinality of the dominant class for both the labled and unlabeled instances, in i-th cluster, respectively. Intuitively, we look at labeled points, $x \in X_L$, distribution in the clusters and compare it with the distribution of the unlabeled points, $x \in X_U$. We expect that most labeled instances are in accordance with most unlabeled instances in the same cluster. If cluster i contains no labeled data, then $Cd_i = 1$ by default. If $Cd_i = 0$, then we set $Cp_i = 0$.

4.2.3 Smoothness based fitness evaluation

The smoothness performance metric is defined based on the SSL smoothness assumption; given the data space, points close to each other probably share the same label. In this case, we calculate the distances among labeled and unlabeled data points. Then, a hinge-loss like score is calculated based on the distance and the label difference, for all the unlabeled data points, according to the following formulation:

$$\mathcal{H}_s = \sum_{i=1}^{l} \sum_{j=l+1}^{l+u} 1/d(x_i, x_j) \cdot \mathcal{I}(y_i, \hat{y}_i) \tag{4.6}$$

where $d(x_i, x_j)$ is the user defined distance between x_i and x_j, i.e. Euclidean distance and $\mathcal{I}(y_i, \hat{y}_j)$ is a function defined as:

$$\mathcal{I}(y_i, \hat{y}_j) = \begin{cases} 0 \; iff \; y_i - \hat{y}_j = 0 \\ 1 \; otherwise \end{cases} \tag{4.7}$$

Intuitively, points close to each other should have the same labels. Thus, we penalize points close to each other that are assigned different labels. Note that practically, the above distance is calculated for the k nearest labeled neighbors, for each unlabeled datum. In this case, we direct involve labeled, X_L, data points, since $x_i \in X_L$, with the unlabeled, X_U, data points, $x_j \in X_U$.

4.2.4 Graph based fitness evaluation

Graph based performance calculates the differences between FFNN soft labels and graph propagation algorithm soft labels over the unlabeled data set.

A label propagation function estimates unlabeled instances $f(\boldsymbol{x}_j)$ on the defined graph, which satisfies two conditions. Firstly, f has the same values as given labels on the labeled data, i.e. $f(\boldsymbol{x}_i) = \boldsymbol{y}_i, i = 1, \dots, l$. Secondly, f satisfies the weighted average property on the unlabeled data, as in harmonic function approach (Zhu, 2003). The manifold score is the mean absolute error between network outputs $\{\hat{y}_i\}_{\mathrm{i}=1}^{\mathrm{l}}$ and $\{y_i\}_{\mathrm{i}=1}^{\mathrm{l}}$ over the unlabeled data.

$$\mathcal{P}_M = \frac{1}{l}\sum_{i=1}^{l}|y_i - \hat{y}_i| \tag{4.8}$$

4.2.5 Island multi-objective genetic algorithm

An island genetic algorithm is used to generate new candidate FFNNs. Assume a population (set) of n FFNN topologies so that $\mathrm{S}_1 = \{\mathrm{I}_\mathrm{k}\}_{\mathrm{k}=1}^{\mathrm{n}}$. The formulation of the k-th individual has the form of $\mathrm{I}_\mathrm{k} = \{\mathrm{P}_1, \mathrm{P}_2, \ldots, \mathrm{P}_\mathrm{m}\}$, where P_j denotes a corresponding topology parameter. Such parameters may refer to the number of training epochs, the number of hidden layers, the number of neurons per layer, and the form of activation functions. The main idea is the creation of high-level individuals I^* according to a specific performance score, calculated using the fitness function. The GA operates in such a way that a set of random individuals will evolve to very efficient classifiers, through a nature inspired procedure (Carse et al., 1997).

Let us denote as $\mathrm{S} = \{\mathrm{S}_1, \mathrm{S}_2, \ldots, \mathrm{S}_\mathrm{l}\}$ the number of initial population sets, also called islands. Then, a genetic algorithm is employed in each island. Firstly, individuals are picked randomly in pairs of two. Then, a crossover operator, defined as a function $\mathrm{f}_\mathrm{c}\left((\mathrm{I}_\mathrm{j}, \mathrm{I}_\mathrm{k})\right) \rightarrow \left(\mathrm{I}_1^{(\mathrm{new})}, \mathrm{I}_2^{(\mathrm{new})}\right)$, produces two new individuals. These offspring mutate with a probability m_p, so that $\mathrm{f}_\mathrm{m}\left(\mathrm{I}_\mathrm{k}^{(\mathrm{new})}\right) \rightarrow \mathrm{I}_\mathrm{k}^{(\mathrm{new,mutated})}$. Finally, a tournament selection scheme is employed, among parents and offspring, to keep the two best (topologies) and then update the population. Also, individuals among islands may swap according to predefined conditions. We thus derive an island GA. An example of such approach can be found in (Protopapadakis et al., 2016).

The fitness of individual I_k is decided using a fitness function. However, according the conditions stated in Section 4.2, I_i domination over I_j appears to be unlikely. However, for the smooth execution of a GA, we need new solutions every few eras. Therefore, with a small possibility we allow accepting a new solution even if the dominance criteria are not met.

A tournament selection process is adopted among two parents and two children. Dominant solutions are selected by default. Alternatively, we rank the solutions according to the criteria values, using a simple voting mechanism. The two solutions with most wins in pair comparisons among all optimization criteria advance to the next era, by updating existing population. The process is described through the following flowchart (Figure 4.1).

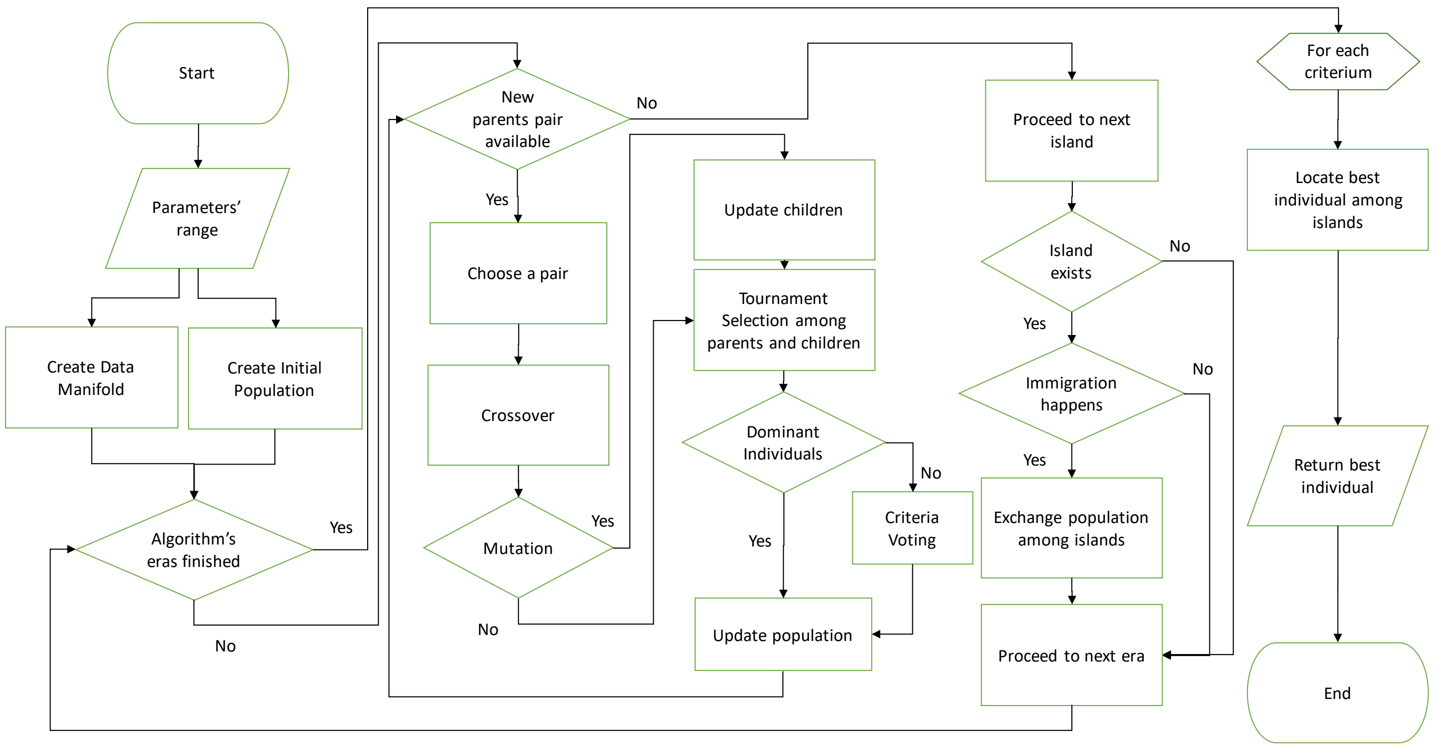


Figure 4.1. The proposed multi-objective island genetic algorithm.

4.3 Experimental evaluation

The proposed methodology was implemented using MATLAB software and native codes. The IGA code was built on the work of (Protopapadakis et al., 2016). Third party codes utilized were: The OPTICS implementation by (Daszykowski et al., 2002b) and harmonic function implementation by (Zhu, 2003). In this set of experiments, the Iris dataset was used. A total of 3 islands with 35 individuals each, for 25 eras were considered. Clusters are formed using OPTICS algorithm's outputs as in (Protopapadakis and Doulamis, 2014). The impact of the k nearest instances selection is also investigated (Figure 4.2).

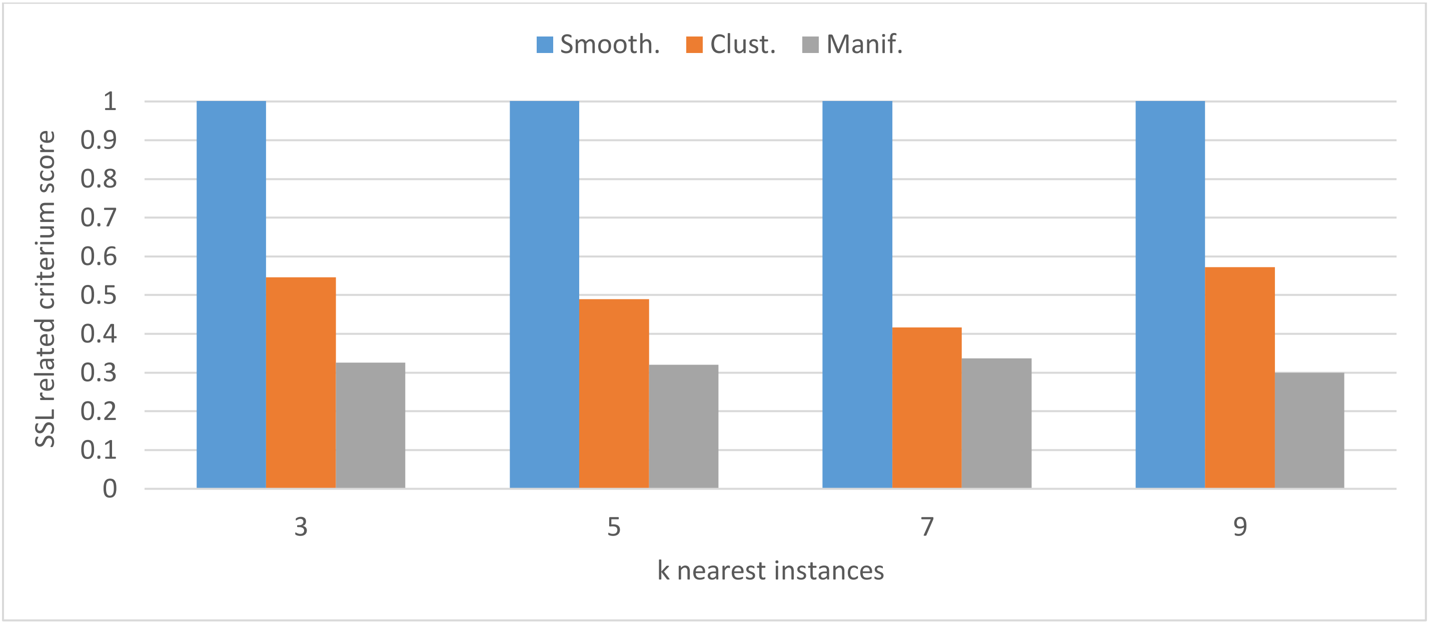


Figure 4.2. The impact of the k nearest instances selection over SSL-inspired performance scores.

Population update time is another parameter worth investigating. Every new accepted solution result in a better overall fitness score, which is a desirable situation. Depending on the problem at hand, the creation of new (improved) solutions may not be possible; the GA will stop. Figure 4.3 illustrates the average survivability time (eras) of any solution, given a selected performance criterion.

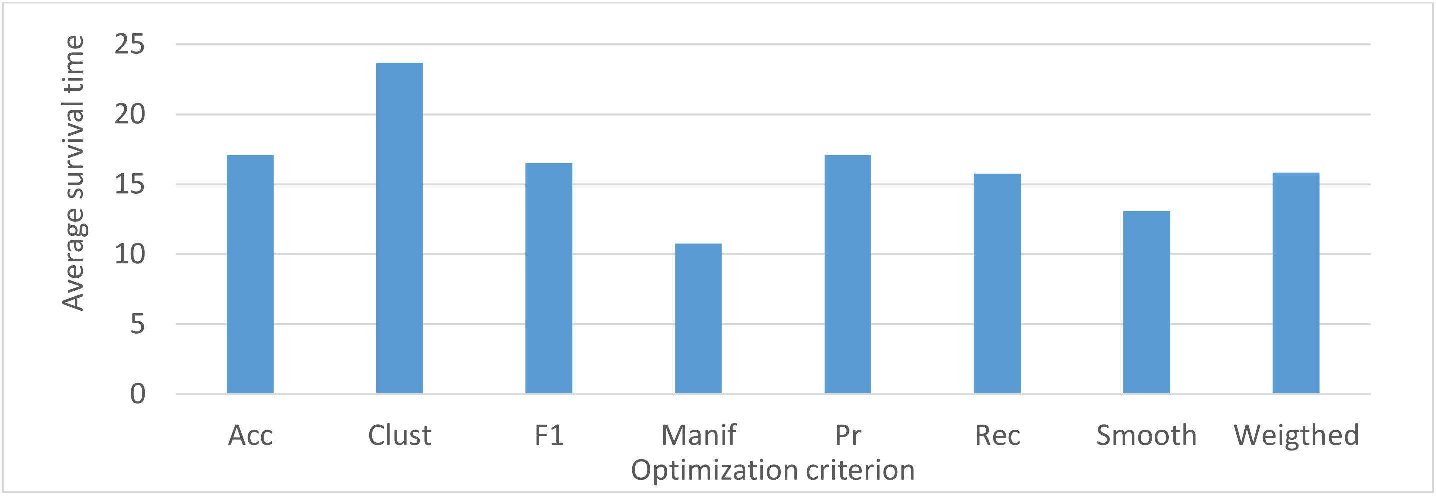


Figure 4.3. Average survivability time (eras) of any solution, given a specific fitness function.

All the employed SSL performance scores utilize the information of the k nearest labeled neighbors for each of the unlabeled data. Smoothness assumption considers only the k closest labeled data. Manifold assumption utilizes a weight matrix W, typicaly sparse; each datum has k edges. Cluster assumption is based on the generated clusters. In our case, OPTICS minimum cluster size is set equal to k. An illustration of the k parameter impact in the SSL criteria performance is illustrated in Figure 4.2.

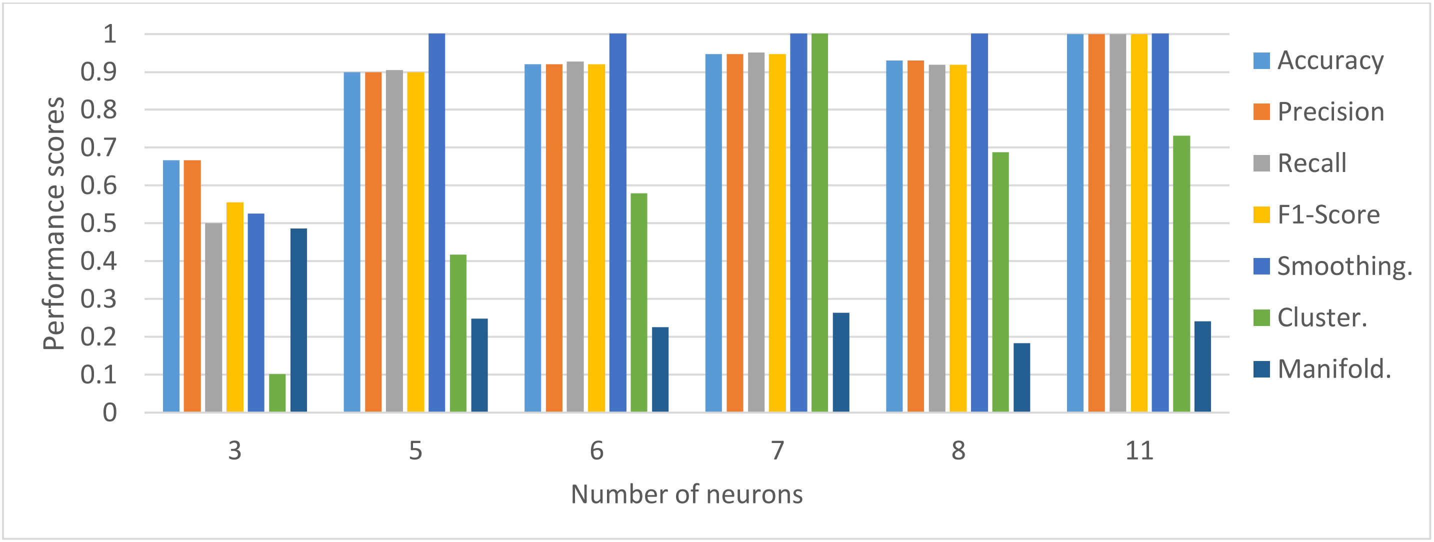


Figure 4.4. Neurons' number effect on performance, for the one-hidden-layer case.

Recall that for all metrics, except smoothness and manifold related, higher the score, better the model. Figure 4.4 illustrates the effect of the neurons' number, for the one-hidden-layer case. Results volatility indicates that there are trade-offs in performance depending on the type of fitness function.

Figure 4.5 illustrates the performance of the optimal topologies, for each of the performance criteria, over test (unseen) data. One of the first conclusions that can be drawn from the experimental evaluation is that, even in the case of a relatively simple dataset such as the Iris dataset, there was not one single dominant solution that outperforms all others in terms of all the criteria used. This observation in a sense corroborates the fact that the problem at hand is a multi-objective optimization problem. On another note, taking into consideration the fact that both the cluster- and the smoothness assumption are based on the initial feature space distribution, the comparable performance can be explained.

Furthermore, the weighted average approach does not appear to excel in any of the performance indices, which is in accordance with existing literature findings. The manifold score approach shows similar trends with the traditional F1-score approach. It should also be noted that the selection of SSL performance metrics is "subjective", in the sense that alternative metrics on the same assumptions may possibly vary greatly. Finally, to gain further insight into the applicability and efficacy of multi-objective optimization as a means for parameter and topology selection of FFNN, more investigation must be made in terms of performance metrics used and classification datasets exploited.

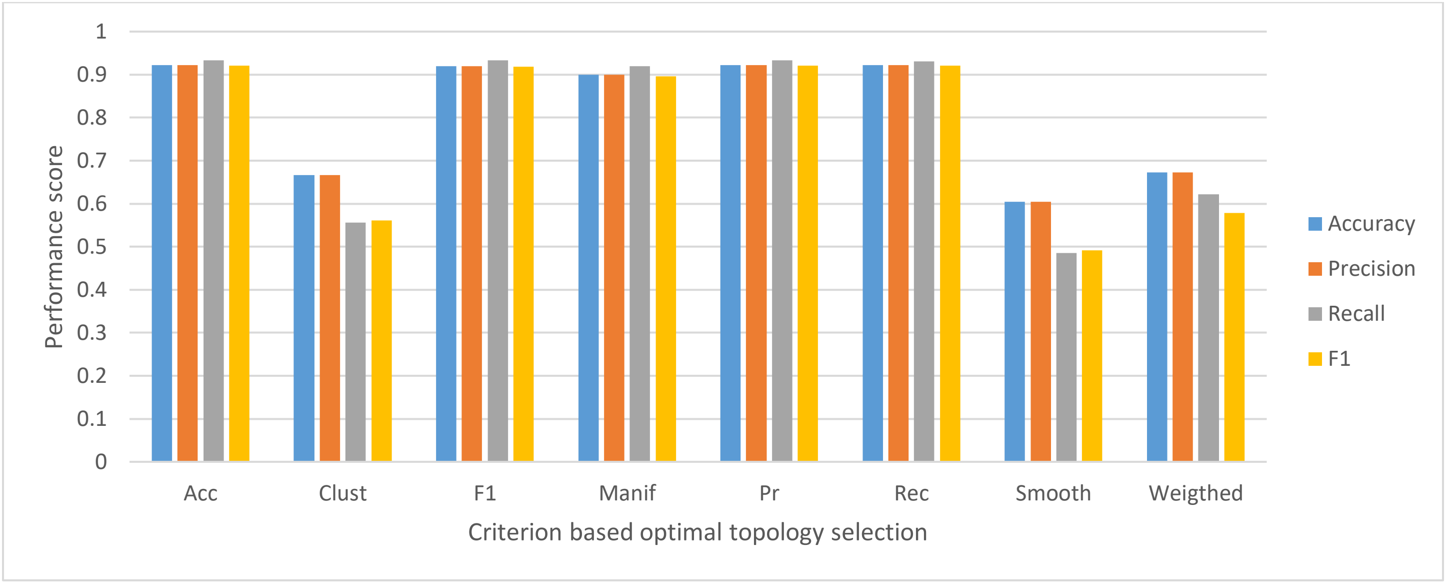


Figure 4.5. Illustration of the four performance scores , for each of the applied fitness functions, over test data.

4.4 Concluding remarks

Inspired by the three main assumptions of SSL, we have formulated a multi-objective problem for the topology optimization of a feedforward neural network. Traditional performance metrics are considered in conjunction with the SSL ones in a scheme that, also, employs an island genetic algorithm for the generation of new candidates, i.e. FFNN topology solutions. The results indicate that the problem at hand was correctly considered as a multi-objective problem. Certain SSL inspired metrics may be appropriate, but it is necessary to conduct further exploration on metrics and Pareto-based optimization techniques, as well as, experimentation on more complex datasets to pinpoint and highlight the effectiveness and merits of such an approach.

Chapter V: Custom performance training functions

The science of today is the technology of tomorrow.

Edward Teller, Hungarian-American theoretical physicist

5 Custom training functions for deep neural network training, using semi-supervised assumptions

In this chapter we present a combinatory approach of two well-known fields: deep learning and semi supervised learning. The proposed methodology investigates how the soft label estimations on the unlabeled data can be used to fine-tune a deep architecture. The adopted approach utilizes at first stacked autoencoders to reduce, non-linearly, data dimensionality. Then multiple SSL approaches operate over the compressed feature space and provide soft labels for the unlabeled data. Finally, all data (labeled and soft-labeled) were used by networks training performance functions. Results suggest that SSL can be beneficial in models' performance. However, it is not clear what type of SSL approach should be adopted, nor its statistical significance.

5.1 Introduction

A typical DNN training approach consists of two steps: a) training per layer and b) fine tuning of the entire network. Training per layer is an unsupervised approach that exploits all available data, labeled or not (see sec. 2.1.2). The fine-tuning approach, however, is limited only to available labeled data instances, which are a small portion of the available data entries.

Training per layer is a straightforward approach. Each layer learns to reconstruct the input values using fewer computational nodes than the number of input feature values. This is a compression scheme; we try to maintain the input's information using fewer neurons. At the end, when we stack all these layers, we have a deep architecture, also known as stacked autoencoders, capable of performing non-linear principal component analysis. The mapped inputs, which lie in a reduced dimension space, can be fed to any classifier.

Then, the stacked autoencoders network, and an additional softmax layer at the end of the network, form a classifier with exceptionally good performance. Such DNN, in many cases, provide better outcomes, compared to traditional machine learning approaches. However, before testing the models, fine-tuning is required. In this case, the entire network is trained, using backpropagation algorithm, with performance functions as the mean square error (MSE) or cross-entropy. Both performance functions are calculated over the available labeled data instances.

At this point, someone could understand the impact of using unlabeled data in the performance function, employing SSL techniques. SSL provides a variety of tools that allow the usage of unlabeled data in conjunction with a small amount of labeled data. These tools are mainly regularizers; i.e. functionals, defined by the user and the adopted SSL assumption(s). Details on the SSL assumptions are provided in section 2.2.2. The advance of such approach is that fine-tuning can be done using much more data. This is expected to improve the model's generalization capabilities.

5.2 The back-propagation algorithm

The goal of backpropagation is to compute the partial derivatives $\frac{\theta C}{\partial w}$ and $\frac{\partial C}{\partial b}$ of the cost function C with respect to any weight w or bias b in the network. Prior to any explanation, we should note that:

1. The cost function can be written as an average $C = \frac{1}{n}\sum_x C_x$ over the cost functions C_x for individual training examples, $\boldsymbol{x}$. The reason we need this assumption is because backpropagation allow for the computation of the partial derivatives $\frac{\theta C_x}{\partial w}$ and $\frac{\partial C_x}{\partial b}$, for a single training example. The $\frac{\theta C}{\partial w}$ and $\frac{\partial C}{\partial b}$ are, then, calculated by averaging over training examples.
2. The cost function can be written as a function of the outputs from the neural network, which depends from the network's topology.

If the above conditions are met, we can update any synaptic weight value, between nodes i and j, w_{ij}^t, at a time instance t according to the following equation:

$$\Delta w_{ij}^t = -\alpha \cdot \delta_j \cdot o_i \tag{5.1}$$

where $a > 0$ is the learning rate, o_i is the output value of neuron i. The term δ_j is defined as:

$$\delta_j = \begin{cases} (o_j - t_j) \cdot o_i \cdot (1 - o_j) & , if\ j\ an\ output\ neuron \\ \left(\sum_{l \in L} w_{jl} \cdot \delta_l - t_j\right) \cdot o_i \cdot (1 - o_j) & , if\ j\ an\ inner\ neuron \end{cases} \tag{5.2}$$

where t_j is the target value. As we can see, equation (5.2), requires the actual target values, $t_j, j = 1, \dots, l$. Thus, in the current form, backpropagation can be applied on labeled data.

5.3 SSL based performance functions

Assume a loss function $\mathcal{E}(\cdot)$, e.g. MAE or MSE. Then, for a given artificial network's topology the loss over the labeled data is set as:

$$\mathcal{E}(\widehat{\boldsymbol{Y}} - \boldsymbol{Y}) = \mathcal{E}\left(\begin{bmatrix} \widehat{\boldsymbol{y}}_1 \\ \vdots \\ \widehat{\boldsymbol{y}}_l \end{bmatrix} - \begin{bmatrix} \boldsymbol{t}_1 \\ \vdots \\ \boldsymbol{t}_l \end{bmatrix}\right) \tag{5.3}$$

where $\widehat{\boldsymbol{y}}_i, i = 1, \dots, l$ is the networks output for input $\boldsymbol{x}_i$, and $\boldsymbol{t}_i$ is the actual target corresponding value. That is, $\boldsymbol{t}_i$ are the annotations (targets) for labeled data. Our approach utilizes, additionally, the unlabeled instances by extending equation (5.3) to (5.4):

$$\mathcal{E}(\widehat{\boldsymbol{Y}} - \boldsymbol{Y}) = \mathcal{E}\left(\begin{bmatrix} \widehat{\boldsymbol{y}}_1 \\ \vdots \\ \widehat{\boldsymbol{y}}_l \\ \widehat{\boldsymbol{y}}_{l+1} \\ \vdots \\ \widehat{\boldsymbol{y}}_n \end{bmatrix} - \begin{bmatrix} \boldsymbol{t}_1 \\ \vdots \\ \boldsymbol{t}_l \\ \widehat{\boldsymbol{t}}_{l+1} \\ \vdots \\ \widehat{\boldsymbol{t}}_n \end{bmatrix}\right) \tag{5.4}$$

where $\widehat{\boldsymbol{y}}_i$, $\widehat{\boldsymbol{t}}_k$, $k = l + 1, \dots, n$ is the estimated soft target values, i.e. $\widehat{\boldsymbol{y}}_i, \widehat{\boldsymbol{t}}_k \in \mathbb{R}^c$, where c denotes the number of available classes, over the remaining unlabeled data. In this setup, target value $\widehat{\boldsymbol{t}}_k$ is an estimation for the unlabeled instance $\boldsymbol{x}_k$, provided by any SSL technique described in section 2.4, or other approaches. At this point we should note that utilization of estimations, as actual targets, is extremely risky; back-propagation adjust the weights so that the outputs match the targets. If targets' values are incorrect the DNN will perform poorly.

5.4 Experimental evaluation

In this chapter, CIFAR-10 and MNIST datasets were used for the experiments. The available data passed progressively through autoencoders. During that step, a stacked encoder was created. Once the unsupervised training phase (labeled and unlabeled data) was completed, SSL approaches were applied over the stacked encoder's outputs. The original data entries were projected in a low dimensionality feature space. A reduced feature space allows for better performance given a SSL approach. The stacked autoencoder topology was different for each dataset. The learning related hyperparameters were the same in both cases. Table 5.1 provides further details on the setup. All performance scores are average scores over six holdout sets.

As a final step, the autoencoders are stacked with a traditional, one-layer, feed-forward network. Then, the fine-tuning takes place; the entire network is trained using the proposed performance scores. Additional termination criteria, which require the actual output values $\boldsymbol{y}_i$are applied only to labeled data. The final DNN is used to classify new unseen data. Performance scores were calculated for the labeled, unlabeled and test (new, previously unseen) data.

Table 5.1. Autoencoders training parameters setup, for each of the utilized datasets.

	CIFAR-10	**MNIST**
Layer topology	[250 60]	[130 60]
L_2 weight regularization	0.0031	0.0031
Sparsity regularization	1.3	1.3
Sparsity proportion	0.12	0.12
Training epochs	1000	1000

5.4.1 Investigating naïve approaches capabilities

Calculating the $\hat{\boldsymbol{t}}_k$ values in a different way, was also considered. This was done deliberately, to illustrate that early-stage SSL approaches, or simplified interpretation of the assumptions, can cause many problems and reduce model's performance. The applied techniques were based on naïve cluster and smoothness approaches, harmonic function implementation, and a weighted average of all the above.

5.4.1.1 Cluster approach

At first available data (labeled and unlabeled), $\{\boldsymbol{x}_i\}_{i=1}^{n}$ are clustered into K clusters, each denoted as $C_j, j = 1, \dots, K$. Then for each cluster, and only for the unlabeled instances, the estimated target value is defined as

$$\hat{\boldsymbol{t}}_j = \frac{1}{p_L}\sum_{i=1}^{p_L} \boldsymbol{y}_i \ , j = 1, \dots, p_U \tag{5.5}$$

where p_L, p_U denote the number of labeled and unlabeled instances in the cluster. However, equation (5.5) fails to produce results if cluster $C_j$has no labeled data. In this case, the following equation is applied:

$$\hat{\boldsymbol{t}}_j = \frac{1}{c_v}\sum_{i=1}^{c_v} \hat{\boldsymbol{y}}_i \ \ , j = 1, \dots, p_U \tag{5.6}$$

where c_v is the number of the most confident network's predictions, for the unlabeled instances, in the cluster. Equation (5.6) is a wrapped solution; it utilizes network's predictions in each training iteration.

5.4.1.2 Smoothness approach

It is a k nearest neighbors' approach; for each of the unlabeled data $\{x_i\}_{i=l+1}^{n}$ we calculate the k closest to them from the labeled set $\{x_i\}_{i=1}^{l}$. The estimated target value is defined as:

$$\hat{\boldsymbol{t}}_j = \sum_{i=1}^{k} \frac{1}{1 - logsig\left(d(\boldsymbol{x}_j, \boldsymbol{x}_i)\right)} \cdot \boldsymbol{y}_i \ \ , j = l+1, \dots, n \tag{5.7}$$

5.4.1.3 Other approaches

Two more approaches, for the estimation of $\hat{\boldsymbol{t}}_k$, are adopted: (i) harmonic function, as described in section 2.4.1.1 and (ii) weighted average, where $\hat{\boldsymbol{t}}_j = \frac{1}{3} \cdot \left(\hat{\boldsymbol{t}}_j^{(C)} + \hat{\boldsymbol{t}}_j^{(S)} + \hat{\boldsymbol{t}}_j^{(M)} \right)$.

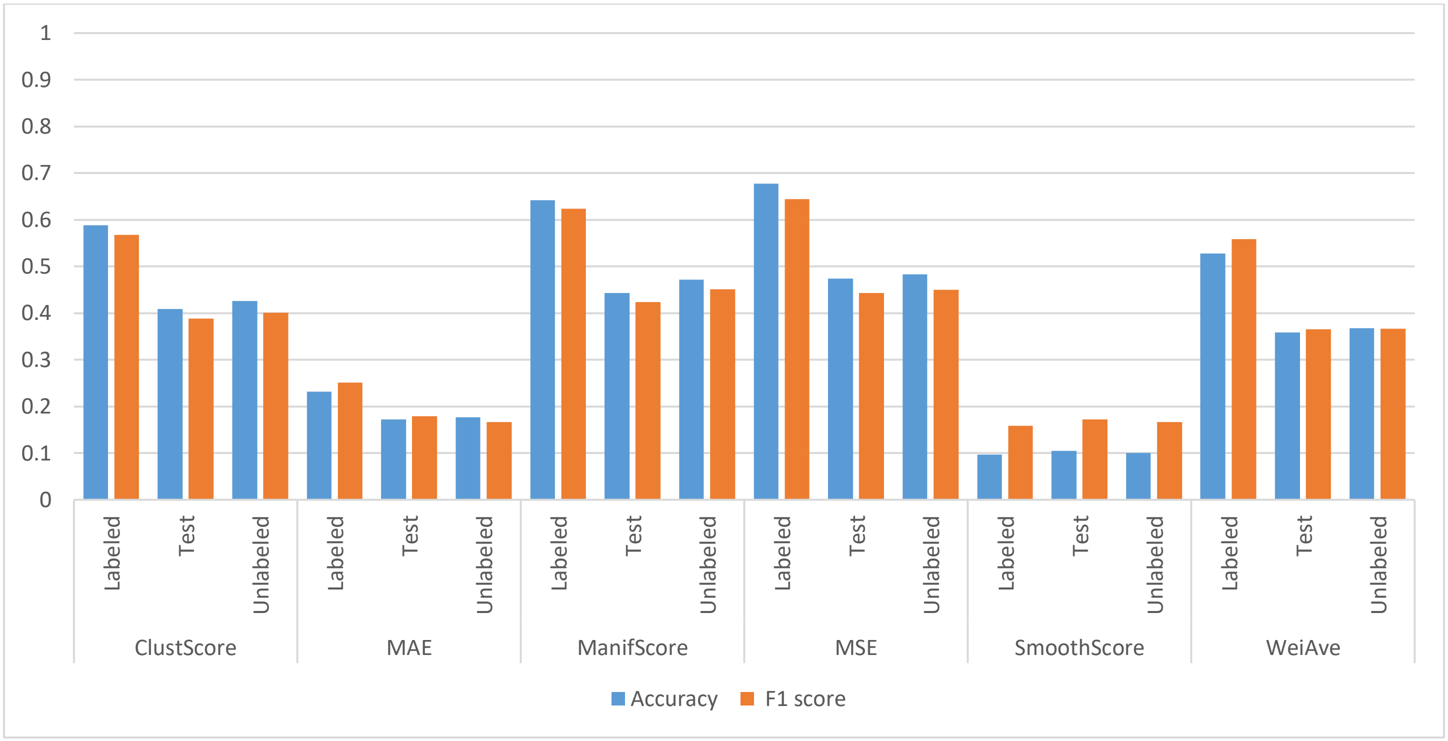


Figure 5.1. Simplified SSL approaches impact on network's performance scores on MNIST dataset.

At this point, we should note that the core performance function $\mathcal{E}(\cdot)$ is the MSE. If we use SSL the MSE is calculated over all data (labeled and unlabeled). Thus, MSE and MAE refer to labeled data-based calculations. ClustScore, ManifScore, SmoothScore, and WeiAve are using MSE as an error function, on both labeled and unlabeled data.

Figure 5.1 provides a further insight on how the approaches described in section 5.4.1 affect the DNN's performance. A simple smoothness inspired SSL technique fails to produce comparable results to other approaches. The manifold approach performance is close to the MSE one but does not surpass it. The best case appears to be a traditional training process, over labeled data only.

We should point out that the current setup advises against the usage of simple SSL techniques; All the investigated techniques fail to produce better results compared to the traditional MSE over the labeled data. Even the manifold approach is not advised. Harmonic functions are not efficient in handling large datasets; they require hardware resources and time. The weighted average, also, fails since smoothness score affects the overall outcome.

5.4.2 Investigating advanced methods capabilities

As a second step, methods that are more complicated were adopted. Further details on these methods can be found in section 2.4. Figure 5.2 illustrates the performance results for the MNIST dataset. In this case, SMIR provides the best results. However, all but MAE based approaches, produce similar scores.

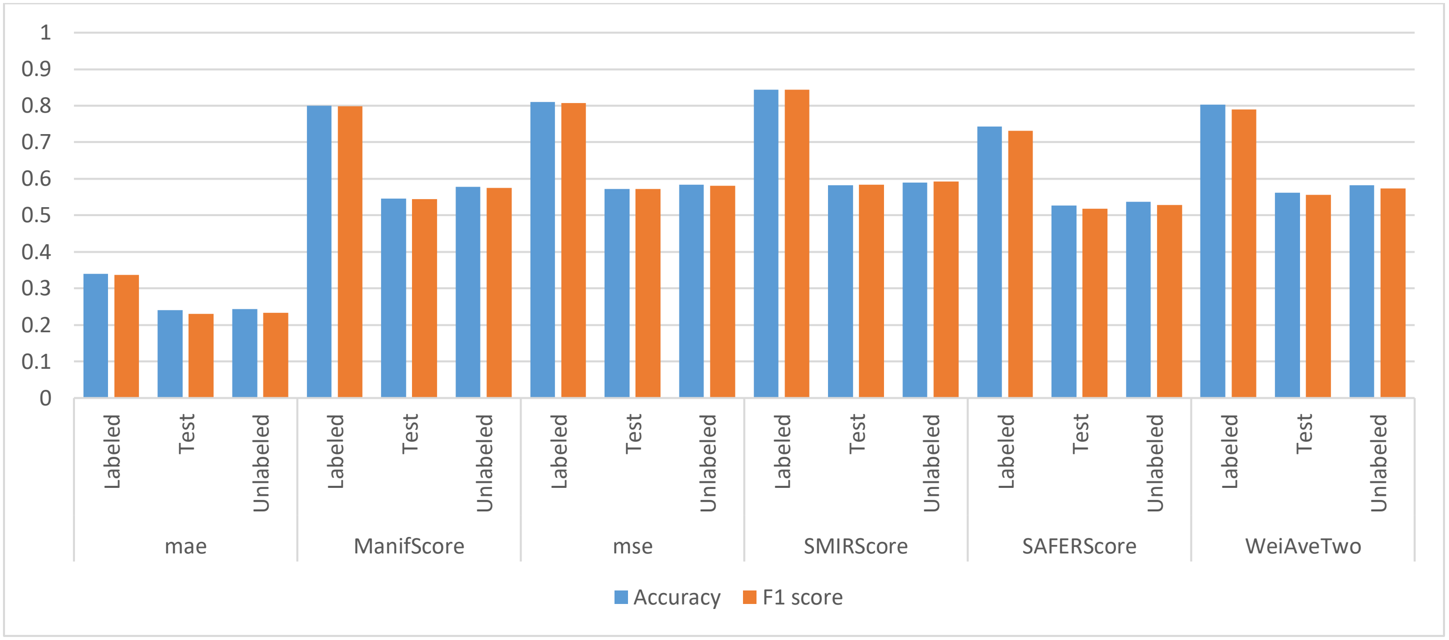


Figure 5.2. Advanced SSL approaches impact on network's performance scores on MNIST dataset.

Figure 5.3 illustrates the performance scores on CIFAR dataset. In this case, we have significant changes compared to the findings over MNIST dataset. Firstly, SMIR and manifold (Anchor graph) perform worse compared to the other SSL approaches. Secondly, SAFER performed better than MSE. It is also intriguing that the weighted average performed better than MSE, even if two out of three components (i.e. Manifold and SMIR) performed worse than MSE. Using MAE during back-propagation resulted in poor performance.

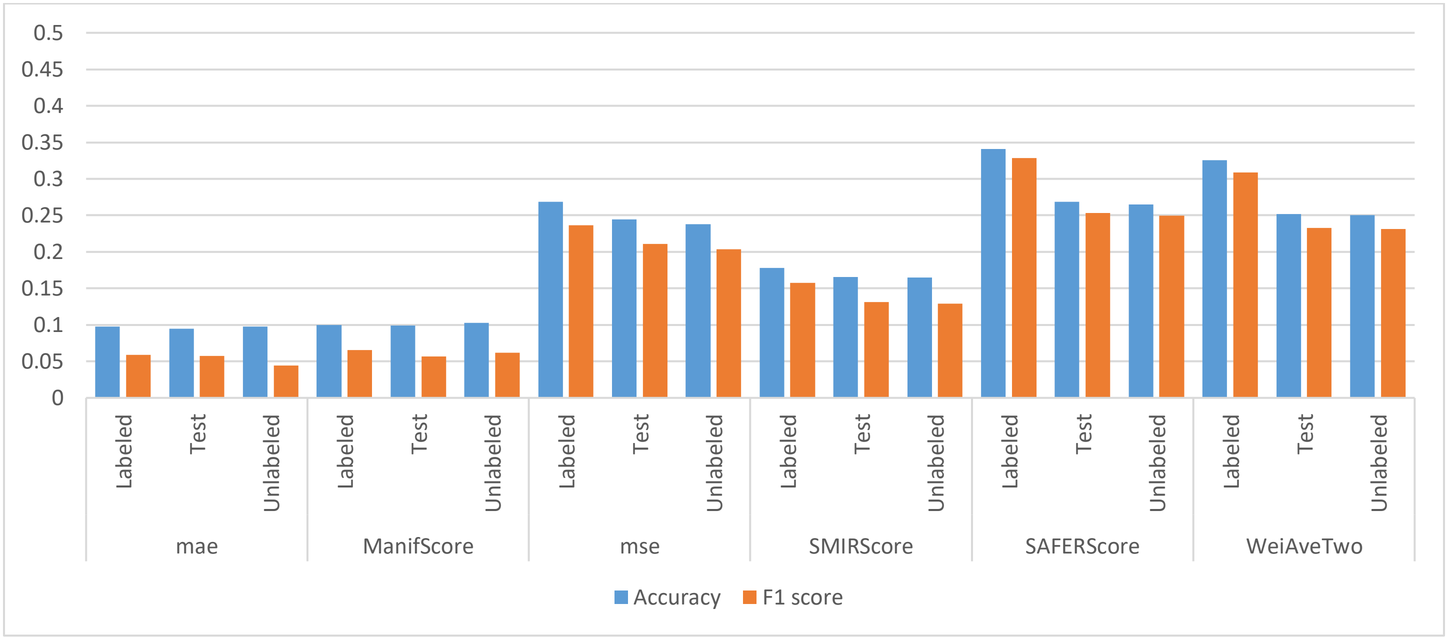


Figure 5.3. Advanced SSL approaches impact on network's performance scores on CIFAR dataset.

5.5 Concluding remarks

The applicability of various SSL approaches, during fine-tuning phase of a DNN, using stacked autoencoders, has been investigated. The utilization of unlabeled data during training could improve the DNN's performance. However, any performance improvements are limited to the suitability of the adopted SSL approach. As such, there are no indications towards a dominant SSL approach.

Chapter VI: Buildings segmentation over orthoimages

Experimentation is an active science.

Claude Bernard, French physiologist

6 Combinatory decision support models in surveyors' applications

This chapter focus on the application of deep neural networks, trained using SSL approaches, on land cover classification problems. Building detection from satellite images can be useful in several remote-sensing applications, such as city planning, urban mapping, and urban change detection. The problem at hand entails to the detection of building territories, using as input color infrared orthoimages. Data were extracted from camera devices and were feed to a semi-supervised trained, deep network scheme. The main contribution lies in the incorporation of all available data (aerial images combined with the corresponding Dense Image Matching point clouds), during all steps of the training process. Annotations became available using crowdsourcing, on a limited amount of data.

6.1 Introduction

Land cover classification is a widely studied field since the appearance of the first satellite images. In the last two decades, the sensors attached to satellites have evolved in a way that nowadays allows the capture of high-resolution multispectral satellite images. This technological advance made the detection and classification of buildings and other man-made structures from satellite images possible. The automatic identification of buildings in urban areas, using remote sensing data, can be beneficial in many applications including cadastre, urban and rural planning, urban change detection, mapping, updating geographic information systems, monitoring, housing value and navigation.

Typically, sensory data have the form of RGB, thermal, multispectral or LIDAR images. Available information allows the researchers to mitigate any drawbacks related to occlusions, shadows, and vegetation. It, also, supports the buildings' detection under different geometric and radiometric diversities, which is a common case in complex scenes. Building detection from 2–D images has been achieved using a variety of methods, where a building can be described either as a group of pixels sharing some common properties or as an object described by specific features or geometric properties (Konstantinidis et al., 2017). Pixel-based methods attempt to extract buildings by appropriately clustering image pixels into homogeneous regions.

Deep Convolutional Neural Networks (CNNs) have been considered extremely beneficial for semantic segmentation tasks in multiple remote sensing applications (Chen et al., 2018; Konstantinos Makantasis et al., 2015; Maltezos et al., 2018; Schenkel and Middelmann, 2019). Stacked autoencoders or similar deep neural network (DNN) are also used (Li et al., 2016; Liang et al., 2018), and provide accurate results. A typical DNN's training approach consists of two steps: a) *training per layer* and b) *fine tuning of the entire network*. Training per layer is an unsupervised process exploiting all available data, labeled or not. On the other hand, the fine-tuning approach is limited only to available labeled data instances, that is a supervised process. However, in remote sensing applications, the available training data are a small portion of the total data entities.

In training per layer each layer learns to reconstruct the input values using fewer computational nodes than the number of input feature values. This is *a compression scheme*; we maintain the input information using fewer neurons. When we stack all these layers, we have a deep architecture, also known as *stacked autoencoders* (Eftychios Protopapadakis et al., 2017a), capable of performing non-linear principal component analysis. The mapped inputs, which lie in a reduced dimension space, can be fed to any classifier. Fine tuning is the process of inducing minor adjustments to the weights of a network. It is used when we have stacked pretrained layers to form a DNN, suitable for the problem at hand.

The stacked autoencoders network, along with an additional softmax layer at the end, forms a classifier with exceptionally good performance. The performance, in many cases, bests traditional machine learning approaches (Qi et al., 2017). Typically, the entire network is fine-tuned, using backpropagation algorithm, with performance functions either the mean square error (MSE) or cross-entropy. Both performance functions are calculated over the available labeled data instances, i.e., a limited set. In this scenario, we enhance the performance functions to utilize unlabeled data, as described in sec. 5.3.

6.2 Semi-supervised semantic segmentation for buildings detection

Pixel level building detection can be seen as a traditional multiclass classification approach. The problem at hand entails to the creation of a robust classifier, capable to understand the difference of an area depicting a building from anything else, e.g. roads and vegetation. This can be achieved by incorporating meaningful information as input values and create a model capable of handling complex patterns.

6.2.1 Dataset description

Study areas namely Area 1, Area 2 and Area 3, situated in Vaihingen city in Germany, were used for training and evaluation purposes. Area 1 mainly consists of historic buildings with notably complex building structure but also has sporadically some, often high, vegetation. Area 2 mainly has high residential buildings with horizontal multiple planes surrounded by long arrays or groups of dense high trees. Area 3 is a purely residential area with small, detached houses that consist of sloped surfaces but there also exists a relatively low vegetation.

A Multi-Dimensional Feature Vector (MDFV) was created to feed the classifiers, as in (Maltezos et al., 2018). The MDFV includes image information from the color-infrared (CIR) aerial images, the DIM point clouds and the vegetation index. Additional to infrared intensities, 3D information was also considered. The cloth simulation (Zhang et al., 2016) and the closest point method are applied to estimate a normalized height DIM/DSM, denoted as nDSM.

The selected hyperparameters of the cloth simulation algorithm for all the test sites were: (i) steep slope and slope processing for the scene, (ii) cloth resolution=1.5, (iii) max iterations=500 and (iv) classification threshold=0.5. This approach addresses issues related with the cost of acquisition and processing, as well as, co-registration aspects, arising when data from multi-modal data sources are fused together, such as LIDAR/DSM and image information. Lastly, the vegetation index for every pixel p_{ij} is considered and estimated through the near infrared NIR band, as:

$$NDVI = \frac{NIR - R}{NIR + R} \tag{6.1}$$

where R and NIR refer to the red and near infrared image band. It should be mentioned that NDVI is computed only for datasets where the NIR channel is available.

Table 6.1 shows the flying parameters and supplementary information about the used dataset of the Vaihingen study areas, as well as, the software instruments we use.

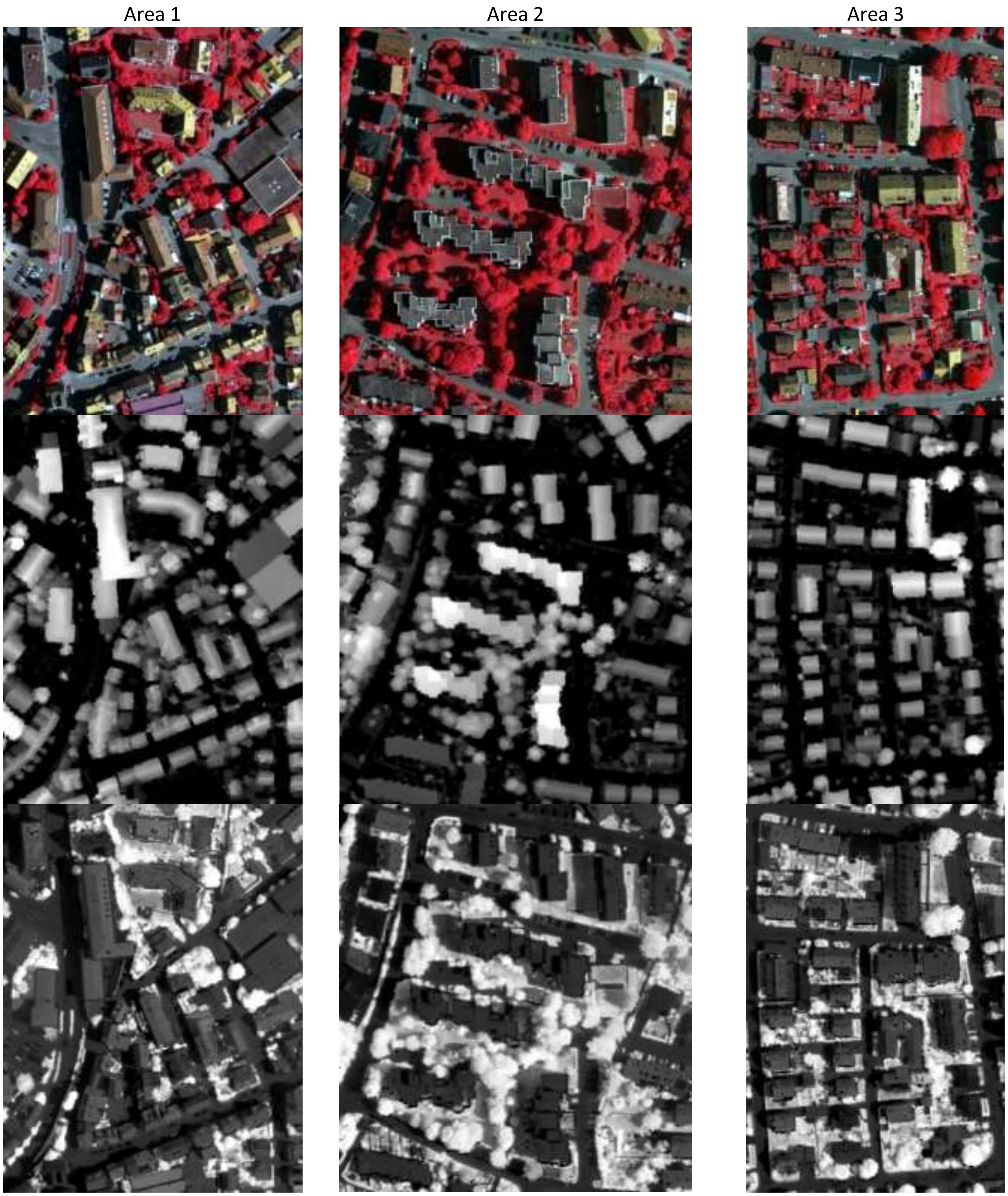


Figure 6.1. Illustrating selected areas for the experiments. Top line: orthoimages. Middle line: nDSM images, Bottom line: NDVI images.

Table 6.1 Related parameters for the used datasets.

Type of images	*Multiple/Digital*
Focal length	120.00mm
Flying height above ground	900m
Forward overlap	60%
Side lap	60%
Ground resolution	8cm
Spectral bands	NIR/R/G
Ground Control Points	20
Triangulation accuracy	<1pixel
Software for DIM	Trimble Inpho (Match-AT, Match-t DSM, Scop++, DTMaster)
GSD of the DIM/DSM	9cm
Software for nDSM	CloudCompare
Software for orthoimages	Trimble Inpho orthovista
GSD of the orthoimages	9cm
Additional descriptors	NDVI via MATLAB
Rows and columns of the block tile	2529×1949 (Area1) 2359×2148 (Area2) 2533×1680 (Area3)
Feature bands of MDFV	NIR/R/G/NDVI/nDSM
Labeled training set	Buildings (1), Vegetation (2), Ground (3)

6.3 Experimental evaluation

A deep neural network classifier has been implemented. Figure 6.2 illustrates the proposed approach. The first two encoder layers, were set using the autoencoder approach; i.e. an unsupervised training approach, where inputs and outputs are the same (Eftychios Protopapadakis et al., 2017a). Parameters for the hidden and output layers were randomly initialized. Then, a fine-tuning training step, using backpropagation algorithm, is applied.Fine-tuning training process involved six different performance functions: standard approaches, i.e. MSE and MAE and four custom ones. The four custom performance functions are:

1. "Manifold" approach, based on the anchor graph SSL technique, described in section 2.4.1.2.
2. "SMIR" approach, based on the maximal performance Gain SSL technique, described in section 2.4.3.
3. "SAFER" approach, based on the squared-loss mutual information SSL technique, described in section 2.4.4.
4. "Weighted average" approach, based on a weighted average of the previous three methods and "MSE" and "MAE" errors.

The initial image is separated into overlapping blogs (patches) of size $15 \times 15 \times 5$. The DNN classifier utilizes these 1,125 values and decides the corresponding class for the pixel at the center of the patch. The first two hidden layers serve as non-linear mappers, reducing the dimensionality of the feature space from 1,125 to 400 and then to 80. Then, a hidden layer of 27 neurons performs the final mapping, allowing for the classification in one of the three pre-defined classes. Figure 6.2 illustrates the proposed DNN topology.

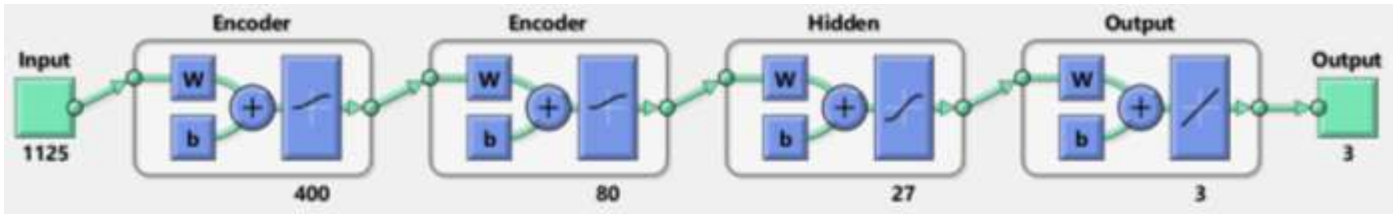


Figure 6.2. Proposed SAE topology for the semantic segmentation of buildings over multiple channel orthoimages.

6.3.1 Amount of data required

This chapter aims to demonstrate the usability of SSL techniques, when limited data are available. Towards that direction, a crowdsourcing approach was considered. We asked the users to draw few polygons over the images. The only limitation was the number of classes. Users had to annotate, i.e. create at least one polygon, for each of the following three categories: Buildings (1), Vegetation (2) and Ground (3). The final areas are shown in Figure 6.3. At this point, we need to clarify that less than 10% of the pixels are annotated. This was done on purpose since we tried to keep the users' effort at minimum.

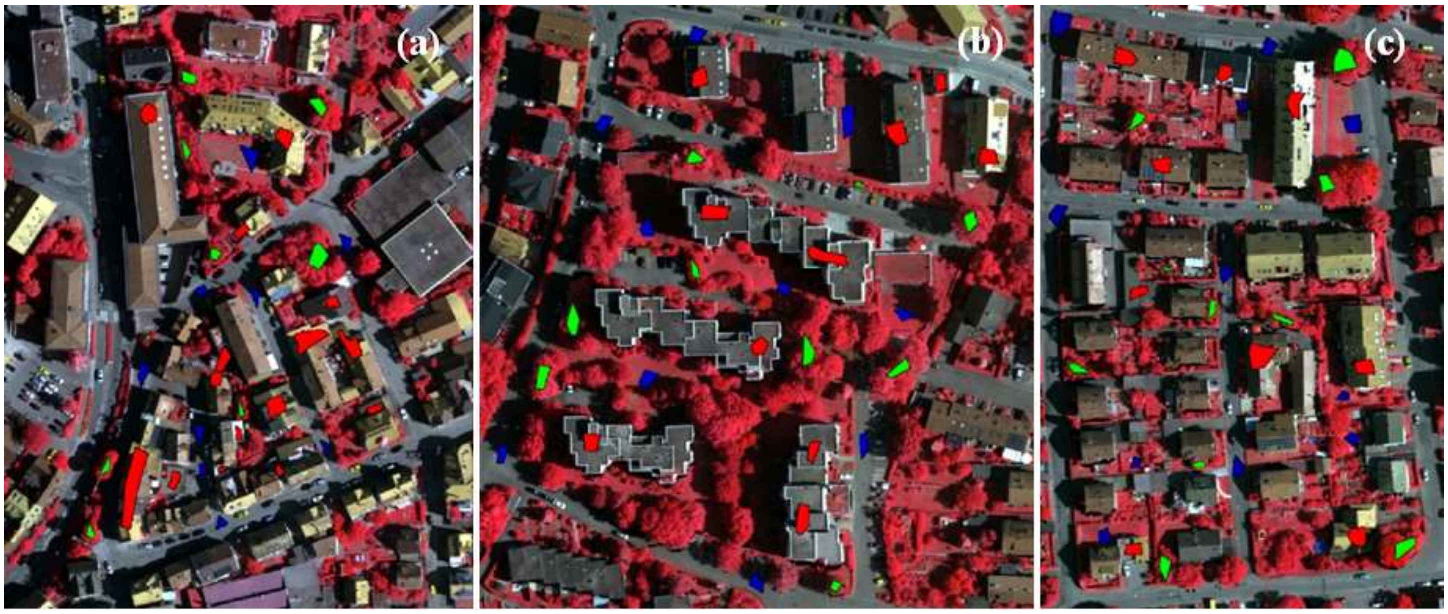


Figure 6.3. Illustrating the manually annotated regions in each of the three investigated areas. Polygons in color correspond to annotations, obtained using crowdsourcing. Areas annotated in green correspond to vegetation. Red and blue correspond to buildings and ground, respectively.

Then, from the available data, within the polygons, we collect less than half of them, and use them to train/validate the classifiers. Concerning the vegetation class, trees with medium and high height are considered as "good" indicative samples. The ground class contains the bare-earth, roads, and low vegetation (grass, low shrubs, etc.). The class buildings contain all the man-made building structures. To improve the classification process, shadowed areas of each class are also included. In addition, the training sample polygons are spatially created to improve representativity of each class and consider the spatial coherency of the content.

Table 6.2. Data distribution utilized for the training process

	Area 1			Area 2			Area 3		
Available annotated data (pixels)	21428 instances, 1125 feature values.			19775 instances, 1125 feature values.			22452 instances, 1125 feature values.		
Initial instances (pixel level) distribution	Class	Count	Percentage	Class	Count	Percentage	Class	Count	Percentage
	1	13730	64.08%	1	9271	46.88%	1	9205	41.00%
	2	3971	18.53%	2	4969	25.13%	2	6003	26.74%
	3	3727	17.39%	3	5535	27.99%	3	7244	32.26%
Labeled data distribution (used for training)	Class	Count	Percent	Class	Count	Percent	Class	Count	Percent
	1	2197	64.05%	1	1484	46.87%	1	1473	40.98%
	2	636	18.54%	2	796	25.14%	2	961	26.74%
	3	597	17.41%	3	886	27.98%	3	1160	32.28%
Unlabeled data distribution (used for training)	Class	Count	Percentage	Class	Count	Percentage	Class	Count	Percentage
	1	8787	64.08%	1	5933	46.88%	1	5891	41.00%
	2	2541	18.53%	2	3180	25.13%	2	3842	26.74%
	3	2385	17.39%	3	3542	27.99%	3	4636	32.26%
Unseen (test) data (used for evaluation)	Class	Count	Percentage	Class	Count	Percentage	Class	Count	Percentage
	1	2746	64.08%	1	1854	46.89%	1	1841	41.01%
	2	794	18.53%	2	993	25.11%	2	1200	26.73%
	3	745	17.39%	3	1107	28.00%	3	1448	32.26%

6.3.2 Experimental results

To purify the output of the classifier from the noisy data, only the building category is selected from the available classes. Thus, only the pixels associated with the buildings are extracted. The building mask is refined by post-processing techniques. The goal of the post processing is to remove noisy regions, such as isolated pixels or tiny blobs of pixels and retains local coherency of the data. Towards this, initially a majority voting technique with a radius of 21 pixels is implemented. Also, an erosion filter of a 7×7 window is applied.

The majority voting filter categorizes the potential building block, with respect to the outputs of the neighboring output data. This filter addresses the spatial coherency that a building has. Since the orthoimages generated based on DSMs, the building boundaries are blurred due to mismatches during the application DIM algorithm. This affects the building identification results by dilating their boundaries. Thus, the erosion filter was applied to "absorb" possible excessive interpolations on the boundaries by reducing their dilated size.

Two alternative approaches were considered for the evaluation of the models' performance: i) over the polygons-bounded areas and ii) over the original annotations provided with the dataset. The former case involves the test data, as described in

Table 6.2. This is a typical multiclass classification approach. The latter case entails to a binary classification problem: buildings and non-buildings.

Table 6.3 demonstrates the ranking among the proposed approaches. The highest F1-scores are achieved when the weighted average approach is adopted, for areas 1 and 2. Area 3 best score was achieved using SAFER. The same is observed for the Critical Success Index (CSI), which is a more descriptive metric regarding a specific class description. There are also no clear outcomes regarding the second-best technique. Area 1 second best F1-score is achieved when trained using MAE (89.9%); areas 2 and 3 second best scores are achieved using SAFER (92.6%) and Manifold (Anchor Graph - 90.4%) approaches.

Table 6.3. Performance comparison for the proposed methodologies in building detection. Index in parenthesis corresponds to ranking score.

Area	Performance Function	Rec (%)	Pr (%)	CSI (%)	F1
Area 1	MSE	95.4 (2)	84.9 (3)	81.5 (3)	89.8 (3)
	MAE	94.9 (4)	85.4 (2)	81.6 (2)	89.9 (2)
	ManifScore	95.0 (3)	84.3 (4)	80.8 (4)	89.3 (4)
	SMIRScore	**95.9** (1)	83.1 (6)	80.3 (5)	89.0 (5)
	SAFERScore	93.2 (6)	84.3 (4)	79.4 (6)	88.5 (6)
	WeiAveTwo	94.3 (5)	**87.1** (1)	**82.7** (1)	**90.6** (1)
Area 2	MSE	**92.8** (1)	91.8 (6)	85.7 (3)	92.3 (3)
	MAE	85.3 (6)	**95.5** (1)	82.0 (6)	90.1 (6)
	ManifScore	88.6 (5)	95.1 (2)	84.7 (5)	91.7 (5)
	SMIRScore	90.3 (4)	94.1 (4)	85.4 (4)	92.2 (4)
	SAFERScore	91.6 (2)	93.7 (5)	86.3 (2)	92.6 (2)
	WeiAveTwo	91.2 (3)	94.6 (3)	**86.7** (1)	**92.9** (1)
Area 3	MSE	86.0 (6)	**94.9** (1)	82.2 (4)	90.2 (4)
	MAE	**91.3** (1)	86.9 (6)	80.3 (6)	89.0 (6)
	ManifScore	87.7 (5)	93.3 (2)	82.5 (2)	90.4 (2)
	SMIRScore	87.8 (4)	92.7 (3)	82.1 (5)	90.2 (4)
	SAFERScore	88.7 (3)	92.6 (4)	**82.9** (1)	**90.6** (1)
	WeiAveTwo	89.6 (2)	91.1 (5)	82.4 (3)	90.3 (3)

Table 6.4 summarizes the performance of the proposed approach, against other state of the art techniques. At first site, the proposed methodology appears to provide average results. However, let us first investigate a bit more the alternatives. Firstly, we have model-based approaches (Maltezos and Ioannidis, 2015); these are typically rule based. Rules are defined by experts, depending on the evaluation scenario. Therefore, models are prone to human errors, require a lot of time and are not able to generalize. Secondly, LIDAR data are expensive. The large-scale implementations are hard to achieve and require a lot of effort and human resources. The utilization of heterogenous data sources e.g. images and LIDAR, makes the creation and training/evaluation of the models even more difficult.

Table 6.4 Comparative results against other SotA techniques.

Related work (approach)	Data type	Implementation type	Average scores (all areas)		
			Rec (%)	**Pr (%)**	**CSI (%)**
(Rottensteiner, 2013) (DLR)	Orthoimages+DIM/DSM	Model based	93.3	96.0	89.8
(Rottensteiner, 2013) (LJU2)	Orthoimages+LIDAR/DSM	Model based	94.6	94.4	89.5
(Rottensteiner, 2013) (ZJU)	Orthoimages+LIDAR/DSM	Machine learning	92.8	96.4	89.7
(Bulatov et al., 2014)	Orthoimages+DIM/DSM	Model based	89.0	86.9	78.5
(Rottensteiner, 2013)	Orthoimages+DIM/DSM	Model based	93.6	90.3	85.0
(Maltezos et al., 2018)	Orthoimages+DIM/DSM	Machine learning	94.9	86.6	82.7
Proposed DNN (MSE)	Orthoimages+DIM/DSM	Machine learning	91.4	90.5	83.2
Proposed DNN (MAE)	Orthoimages+DIM/DSM	Machine learning	90.5	89.3	81.3
Proposed DNN (ManifScore)	Orthoimages+DIM/DSM	Machine learning	90.4	90.9	82.7
Proposed DNN (SMIRScore)	Orthoimages+DIM/DSM	Machine learning	91.3	90.0	82.6
Proposed DNN (SAFERScore)	Orthoimages+DIM/DSM	Machine learning	91.2	90.2	82.8
Proposed DNN (WeiAveTwo)	Orthoimages+DIM/DSM	Machine learning	91.7	90.9	83.9
(Mongus et al., 2014)	LIDAR/DSM	Model based	89.7	95.2	85.8
(Moussa and El-Sheimy, 2012)	Orthoimages+LIDAR/DSM	Model based	89.8	95.1	85.8

(Grigillo and Kanjir, 2012)	Orthoimages+LIDAR/DSM	Model based	94.2	94.6	89.4
(Wei et al., 2012)	Orthoimages+LIDAR/DSM	Machine learning	89.7	92.9	83.9
(Niemeyer et al., 2011)	LIDAR (as point cloud)	Machine learning	91.5	92.5	85.2
(Niemeyer et al., 2013)	LIDAR (as point cloud)	Machine learning	90.2	93.2	84.6
(Gerke and Xiao, 2014)	LIDAR (as point cloud) + images	Machine learning	91.4	90.6	83.5

Contrary to the most techniques of Table 6.4, we are limited to less than 5% of data for the creation of a training set, when 80% is the common case. Also, our approach utilizes a single data source saving time and effort. The feature creation occurs automatically, using the encoders, and data annotation process requires approximate crowdsourced polygon regions, instead of pixel level annotations by experts. Lastly, the models can generalize relatively easy. Considering all the above, a small tradeoff between performance and ease of usage is considered acceptable.

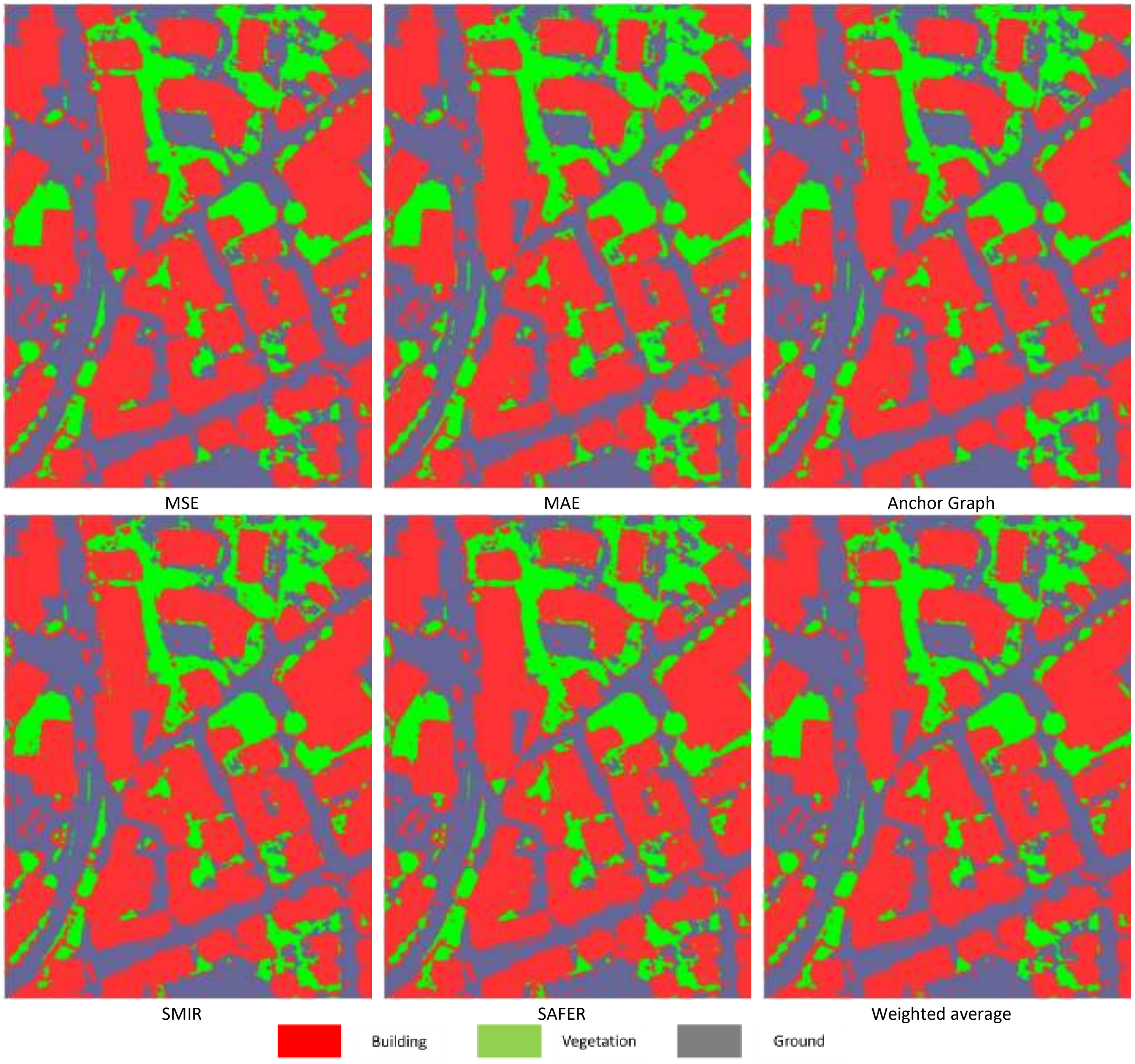


Figure 6.4 Area 1 annotations for the DNN classifier, using six different performance functions for fine-tuning.

Figure 6.4 illustrates the DNN classifiers annotations over Area 1, for all six different performance function trained models. Pixel annotation similarity exceeds 93% for all images. A closer look allows us to identify the main differences. Most of them are located on the boundary regions of the objects. The best DNN model, in this case, was trained using a weighted average as a performance function.

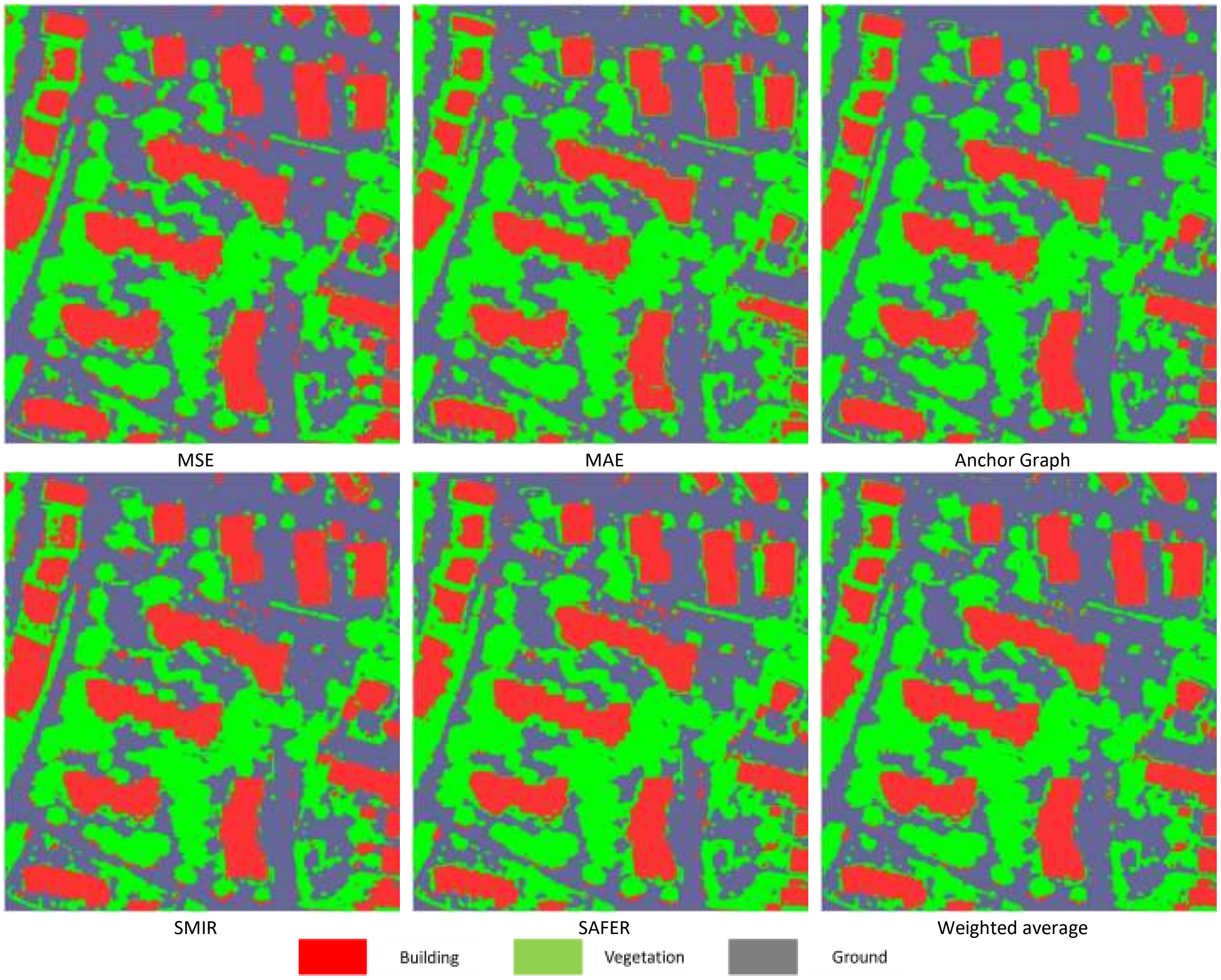


Figure 6.5 Area 2 annotations for the DNN classifier, using six different performance functions for fine-tuning.

Figure 6.5 demonstrates the DNN classifier's performance over small objects (e.g. single trees). Pixel annotation similarity exceeds 85% for all images. However, significant changes in annotations are observed over single objects typically vegetation. Generally, when the object spans less than 10×10 pixels, detection capabilities decline. This could be partially explained since most of the block pixels, i.e. $(15 \times 15) - (10 \times 10) = 125$ pixels, describe something different. The best DNN model, in this case, was trained using a weighted average as a performance function.

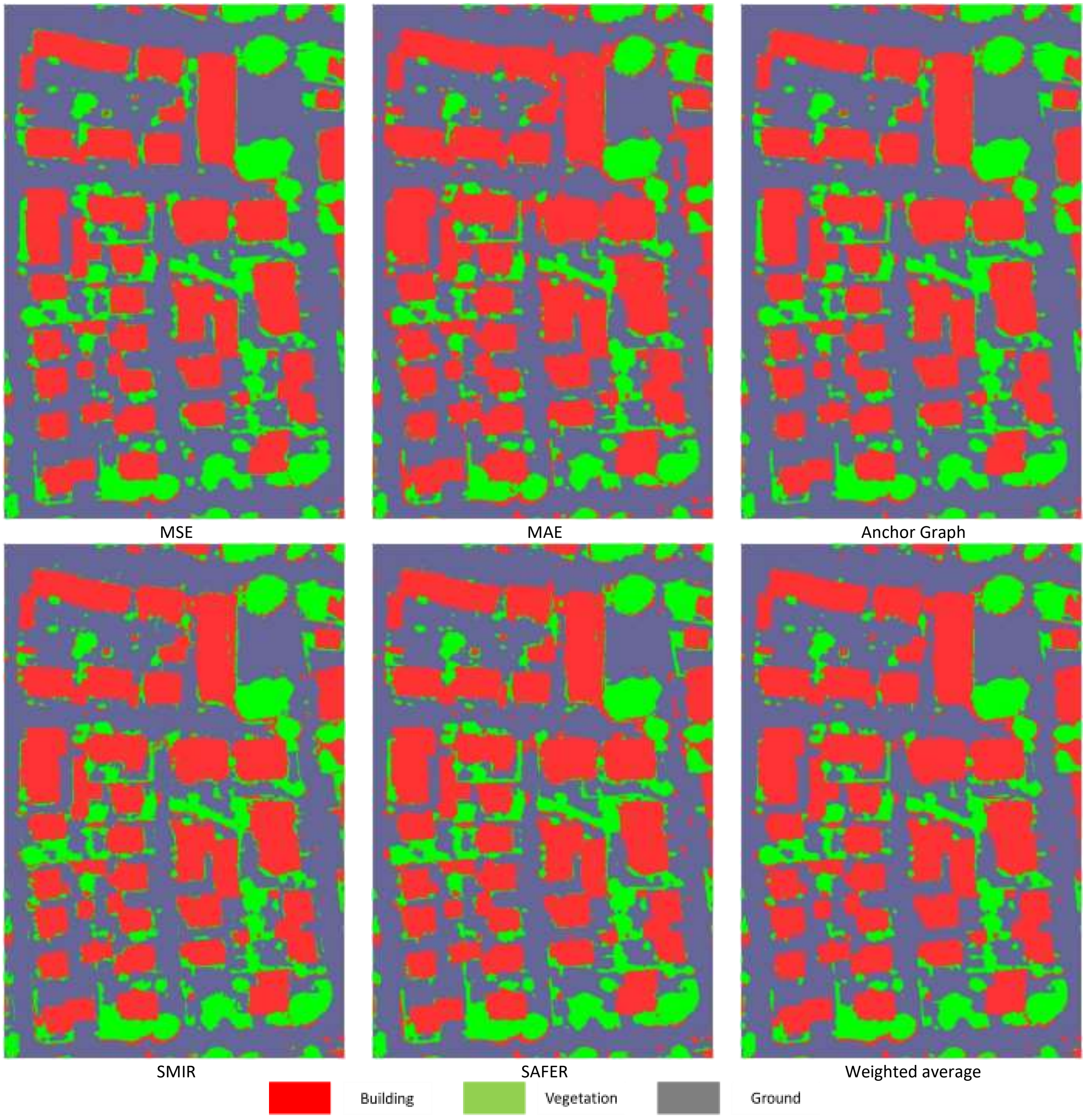


Figure 6.6. Area 3 annotations *for the DNN classifier, using six different performance functions for fine-tuning.*

Figure *6.6* is in accordance with the previous findings. The object can be adequately segmented. Similarity among images exceeds 87%. Misclassifications are observed over the adjacent objects' boundaries and in cases of limited area span. The best DNN model, in this case, was trained using SAFER SSL method for the utilization of the unlabeled data, as a performance function.

Figure 6.7 illustrates the best building detection outcomes for the proposed approaches. Yellow color corresponds to pixels showing a building and model classified them as building (True Positive). Red color indicates pixels that model classified as buildings, but the actual label was either vegetation or ground (False Positive). Finally, blue color indicates areas that were buildings, but model failed to recognize them (False Negative).

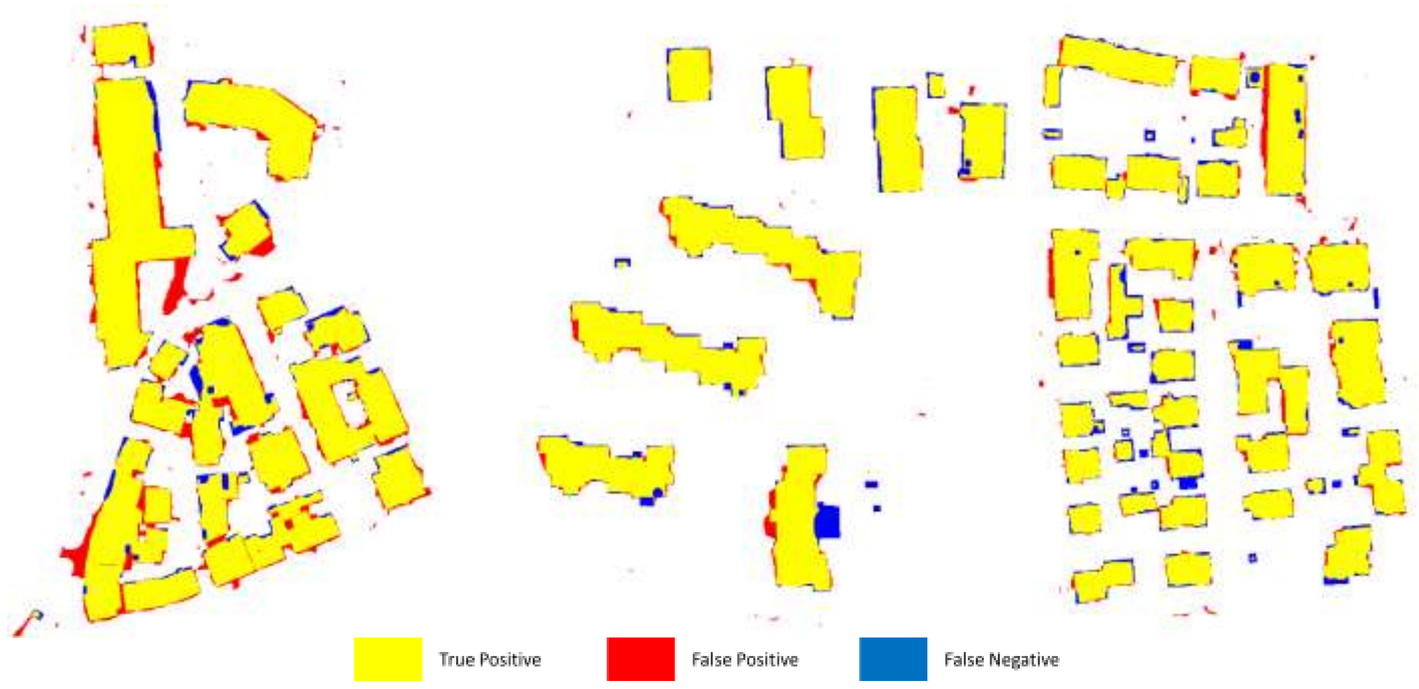


Figure 6.7. Illustrating best building detection results per area. Left: Area 1 (weighted average), center: Area 2 (weighted average), right: Area 3 (SAFER).

In this case, DNN classification capabilities become apparent. Segmentation for building blocks is extremely accurate considering the limited training sample. Misclassification involved inner yards, kiosk size buildings (e.g. bus stations), and the edges of the buildings. An increase of the training samples could mitigate such effects.

6.4 Concluding remarks

Semi-supervised inspired performance functions have been utilized to fine-tune DNN for the detection of buildings over orthoimages. Results indicate that using crowdsourcing and limited train data instances, i.e. selected polygons span less than 5% of the total images area, suffice for the creation of robust detectors.

The approach is based on stacked autoencoders capabilities of non-linear dimensionality reduction. That way the information of $15 \times 15 \times 5$ is summarized to 80-values feature space. The reduced dimensionality allows the implementation of SSL approaches in a way that unlabeled data can be beneficial to the model's training process.

The DNN is a stacked network using the encoding layers of the stacked autoencoders. The fine-tuning performance function is designed from scratch, so that the information provided from the unlabeled data is also considered. At the end, the generated network can handle any kind of new data.

Chapter VII: In the End

I am turned into a sort of machine for observing facts and grinding out conclusions.

Charles Darwin, English naturalist, geologist and biologist

7 Conclusions

In this study, we investigated various fields related to semi-supervised and deep learning approaches. At first, the selection impact, for the creation of training data sets, was investigated. It appears that random selection, i.e. the default method in almost all the experiments, on labeled data can provide adequate results. However, more structured approaches, like uniform feature space sampling, or random sampling over sub clusters can provide additional information and lead to better performance.

The second stage dealt with the impact of various SSL related approaches on the topology of NNs. In this case, a multi-objective island genetic algorithm was employed. Results suggested that none of the investigated performance scores (traditional or SSL based ones) could be considered as dominant. Even in relative easily to handle datasets, like Iris dataset, different assumptions optimize different performance scores. In other words, the SSL assumptions appropriateness is strongly related to the problem at hand.

The third experimental stage involved the custom performance function implementation, during the training phase. In particular, stacked auto encoders were trained and then stacked to form a DNN. Then, during the fine-tuning phase, the back-propagation algorithm used error functions over all (labeled and unlabeled) data. Results suggest that the same SSL approach produce great performance variations, depending on the application scenario.

The usefulness of combinatory DL/SSL approaches had been evaluated on land cover classification problems. The problem focused the detection of building territories, using as input color infrared orthoimages. The main contribution lied in the utilization of all available data (aerial images combined with the corresponding Dense Image Matching point clouds), during all steps of the training process, using a limited amount of annotated data (gathered using crowdsourcing).

The proposed approach used less than 5% of the available data for the creation of a training set, when 80% is the common case. Additionally, using a single data source saved time and effort. The feature creation occurs automatically, through the deep network. The data annotation process requires approximate crowdsourced polygon regions, instead of pixel level annotations by experts. Lastly, the models can generalize relatively easy. Experimental results indicated that is feasible to achieve state of the art performance scores, using considerably less data for training and less effort for setting up the models.

The end.

8 References

Alok, A.K., Saha, S., Ekbal, A., 2014. Feature Selection and Semi-supervised Clustering Using Multiobjective Optimization, in: 2014 International Conference on Soft Computing and Machine Intelligence (ISCMI). Presented at the 2014 International Conference on Soft Computing and Machine Intelligence (ISCMI), pp. 126–129. https://doi.org/10.1109/ISCMI.2014.19

Ankerst, M., Breunig, M.M., Kriegel, H.-P., Sander, J., 1999. OPTICS: Ordering Points to Identify the Clustering Structure, in: Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data, SIGMOD ’99. ACM, New York, NY, USA, pp. 49–60. https://doi.org/10.1145/304182.304187

Belkin, M., Matveeva, I., Niyogi, P., 2004. Regularization and semi-supervised learning on large graphs, in: Learning Theory. Springer, pp. 624–638.

Belkin, M., Niyogi, P., Sindhwani, V., 2006. Manifold Regularization: A Geometric Framework for Learning from Labeled and Unlabeled Examples. J Mach Learn Res 7, 2399–2434.

Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., Raffel, C.A., 2019. MixMatch: A Holistic Approach to Semi-Supervised Learning, in: Wallach, H., Larochelle, H., Beygelzimer, A., Alché-Buc, F. d\textquotesingle, Fox, E., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 32. Curran Associates, Inc., pp. 5049–5059.

Bruzzone, L., Chi, M., Marconcini, M., 2006. A Novel Transductive SVM for Semisupervised Classification of Remote-Sensing Images. IEEE Trans. Geosci. Remote Sens. 44, 3363–3373. https://doi.org/10.1109/TGRS.2006.877950

Bulatov, D., Häufel, G., Meidow, J., Pohl, M., Solbrig, P., Wernerus, P., 2014. Context-based automatic reconstruction and texturing of 3D urban terrain for quick-response tasks. ISPRS J. Photogramm. Remote Sens. 93, 157–170. https://doi.org/10.1016/j.isprsjprs.2014.02.016

Carse, B., Pipe, A.G., Davies, O., 1997. Parallel evolutionary learning of fuzzy rule bases using the island injection genetic algorithm, in: , 1997 IEEE International Conference on Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation. Presented at the , 1997 IEEE International Conference on Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation, pp. 3692–3697 vol.4. https://doi.org/10.1109/ICSMC.1997.633243

Chapelle, O., Zien, A., 2004. Semi-Supervised Classification by Low Density Separation [WWW Document]. URL http://eprints.pascal-network.org/archive/00000388/ (accessed 12.29.13).

Chen, H., Chiang, R.H.L., Storey, V.C., 2012. Business Intelligence and Analytics: From Big Data to Big Impact. MIS Q 36, 1165–1188.

Chen, Jiazhong, Ma, B., Cao, H., Chen, Jie, Fan, Y., Li, R., Wu, W., 2017. Updating initial labels from spectral graph by manifold regularization for saliency detection. Neurocomputing 266, 79–90. https://doi.org/10.1016/j.neucom.2017.04.066

Chen, K., Wang, S., 2011. Semi-supervised learning via regularized boosting working on multiple semi-supervised assumptions. Pattern Anal. Mach. Intell. IEEE Trans. On 33, 129–143.

Chen, K., Weinmann, Michael, Gao, X., Yan, M., Hinz, S., Jutzi, B., Weinmann, Martin, 2018. RESIDUAL SHUFFLING CONVOLUTIONAL NEURAL NETWORKS FOR DEEP SEMANTIC IMAGE SEGMENTATION USING MULTI-MODAL DATA. ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci. 4.

Cheng, S., Shi, Y., Qin, Q., 2012. Particle swarm optimization based semi-supervised learning on Chinese text categorization, in: 2012 IEEE Congress on Evolutionary Computation (CEC). Presented at the 2012 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8. https://doi.org/10.1109/CEC.2012.6252959

Coello, C.C., Lamont, G.B., Veldhuizen, D.A. van, 2007. Evolutionary Algorithms for Solving Multi-Objective Problems. Springer Science & Business Media.

Cox, M.A.A., Cox, T.F., 2008. Multidimensional Scaling, in: Handbook of Data Visualization, Springer Handbooks Comp.Statistics. Springer Berlin Heidelberg, pp. 315–347.

Cybenko, G., 1989. Approximation by superpositions of a sigmoidal function. Math. Control Signals Syst. 2, 303–314. https://doi.org/10.1007/BF02551274

Daszykowski, M., Walczak, B., Massart, D.L., 2002a. Representative subset selection. Anal. Chim. Acta 468, 91–103. https://doi.org/10.1016/S0003-2670(02)00651-7

Daszykowski, M., Walczak, B., Massart, D.L., 2002b. Looking for Natural Patterns in Analytical Data. 2. Tracing Local Density with OPTICS. J. Chem. Inf. Comput. Sci. 42, 500–507. https://doi.org/10.1021/ci010384s de Sousa, C.A.R., Souza, V.M.A., Batista, G.E.A.P.A., 2015. An experimental analysis on time series transductive classification on graphs, in: 2015 International Joint Conference on Neural Networks (IJCNN). Presented at the 2015 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. https://doi.org/10.1109/IJCNN.2015.7280338

Deb, K., 2001. Multi-Objective Optimization Using Evolutionary Algorithms. John Wiley & Sons.

Demirkan, H., Delen, D., 2013. Leveraging the capabilities of service-oriented decision support systems: Putting analytics and big data in cloud. Decis. Support Syst. 55, 412–421. https://doi.org/10.1016/j.dss.2012.05.048

Deng, L., Yu, D., 2014. Deep Learning: Methods and Applications. Found. Trends® Signal Process. 7, 197–387. https://doi.org/10.1561/2000000039

Elhamifar, E., Sapiro, G., Vidal, R., 2012. See all by looking at a few: Sparse modeling for finding representative objects, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Presented at the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1600–1607. https://doi.org/10.1109/CVPR.2012.6247852

Fan, Z., Xu, Y., Zhang, D., 2011. Local Linear Discriminant Analysis Framework Using Sample Neighbors. IEEE Trans. Neural Netw. 22, 1119–1132. https://doi.org/10.1109/TNN.2011.2152852

Fukushima, K., Miyake, S., 1982. Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Visual Pattern Recognition, in: Competition and Cooperation in Neural Nets. Springer, Berlin, Heidelberg, pp. 267–285. https://doi.org/10.1007/978-3-642-46466-9_18

Gerke, M., Xiao, J., 2014. Fusion of airborne laserscanning point clouds and images for supervised and unsupervised scene classification. ISPRS J. Photogramm. Remote Sens. 87, 78–92. https://doi.org/10.1016/j.isprsjprs.2013.10.011

Glorot, X., Bengio, Y., 2010. Understanding the difficulty of training deep feedforward neural networks., in: Aistats. pp. 249–256.

Grigillo, D., Kanjir, U., 2012. Urban object extraction from digital surface model and digital aerial images. ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci. 1, 215–220.

Hornik, K., 1991. Approximation capabilities of multilayer feedforward networks. Neural Netw. 4, 251–257. https://doi.org/10.1016/0893-6080(91)90009-T

Ivakhnenko, A.G., Lapa, V.G., 1967. Cybernetics and forecasting techniques [WWW Document]. CERN Doc. Serv. URL http://cds.cern.ch/record/209675 (accessed 5.15.17).

Kaselimi, M., Protopapadakis, E., Voulodimos, A., Doulamis, N., Doulamis, A., 2019. Multi-Channel Recurrent Convolutional Neural Networks for Energy Disaggregation. IEEE Access 7, 81047–81056. https://doi.org/10.1109/ACCESS.2019.2923742

Kobayashi, A., Oku, T., Imai, T., Nakagawa, S., 2012. Multi-objective optimization for semi-supervised discriminative language modeling, in: 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Presented at the 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4997–5000. https://doi.org/10.1109/ICASSP.2012.6289042

Konstantinidis, D., Stathaki, T., Argyriou, V., Grammalidis, N., 2017. Building Detection Using Enhanced HOG–LBP Features and Region Refinement Processes. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 10, 888–905. https://doi.org/10.1109/JSTARS.2016.2602439

Kosmopoulos, D.I., Doulamis, N.D., Voulodimos, A.S., 2011. Bayesian filter based behavior recognition in workflows allowing for user feedback. Comput. Vis. Image Underst.

Li, W., Fu, H., Yu, L., Gong, P., Feng, D., Li, C., Clinton, N., 2016. Stacked Autoencoder-based deep learning for remote-sensing image classification: a case study of African land-cover mapping. Int. J. Remote Sens. 37, 5632–5646.

Li, Y.-F., Zha, H.-W., Zhou, Z.-H., 2017. Learning Safe Prediction for Semi-Supervised Regression., in: AAAI. pp. 2217–2223.

Liang, P., Shi, W., Zhang, X., 2018. Remote sensing image classification based on stacked denoising autoencoder. Remote Sens. 10, 16.

Liu, W., He, J., Chang, S.-F., 2010. Large graph construction for scalable semi-supervised learning, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10). pp. 679–686.

Luo, Y., Tao, D., Geng, B., Xu, C., Maybank, S.J., 2013. Manifold Regularized Multitask Learning for Semi-Supervised Multilabel Image Classification. IEEE Trans. Image Process. 22, 523–536. https://doi.org/10.1109/TIP.2012.2218825

Luo, Y., Zhu, J., Li, M., Ren, Y., Zhang, B., 2018. Smooth Neighbors on Teacher Graphs for Semi-Supervised Learning. Presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8896–8905.

Makantasis, K., Doulamis, A., Doulamis, N., 2013. A Non-parametric Unsupervised Approach for Content Based Image Retrieval and Clustering, in: Proceedings of the 4th ACM/IEEE International Workshop on Analysis and Retrieval of Tracked Events and Motion in Imagery Stream, ARTEMIS '13. ACM, New York, NY, USA, pp. 33–40. https://doi.org/10.1145/2510650.2510656

Makantasis, Konstantinos, Karantzalos, K., Doulamis, A., Doulamis, N., 2015. Deep supervised learning for hyperspectral data classification through convolutional neural networks, in: 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS). IEEE, pp. 4959–4962.

Makantasis, K., Protopapadakis, E., Doulamis, A., Doulamis, N., Loupos, C., 2015. Deep Convolutional Neural Networks for efficient vision based tunnel inspection, in: 2015 IEEE International Conference on Intelligent Computer Communication and Processing (ICCP). Presented at the 2015 IEEE International Conference on Intelligent Computer Communication and Processing (ICCP), pp. 335–342. https://doi.org/10.1109/ICCP.2015.7312681

Maltezos, E., Ioannidis, C., 2015. Automatic detection of building points from LIDAR and dense image matching point clouds. ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci. II-3/W5.

Maltezos, E., Protopapadakis, E., Doulamis, N., Doulamis, A., Ioannidis, C., 2018. Understanding Historical Cityscapes from Aerial Imagery Through Machine Learning, in: Ioannides, M., Fink, E., Brumana, R., Patias, P., Doulamis, A., Martins, J., Wallace, M. (Eds.), Digital Heritage. Progress in Cultural Heritage: Documentation, Preservation, and Protection, Lecture Notes in Computer Science. Springer International Publishing, pp. 200–211.

Mongus, D., Lukač, N., Žalik, B., 2014. Ground and building extraction from LiDAR data based on differential morphological profiles and locally fitted surfaces. ISPRS J. Photogramm. Remote Sens. 93, 145–156. https://doi.org/10.1016/j.isprsjprs.2013.12.002

Morin, F., Bengio, Y., 2005. Hierarchical Probabilistic Neural Network Language Model., in: Aistats. Citeseer, pp. 246–252.

Moussa, A., El-Sheimy, N., 2012. A new object based method for automated extraction of urban objects from airborne sensors data. Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. 39, B3.

Nesterov, Y., 2013. Introductory lectures on convex optimization: A basic course. Springer Science & Business Media.

Niemeyer, J., Rottensteiner, F., Soergel, U., 2013. Classification of urban LiDAR data using conditional random field and random forests, in: Joint Urban Remote Sensing Event 2013. Presented at the Joint Urban Remote Sensing Event 2013, pp. 139–142. https://doi.org/10.1109/JURSE.2013.6550685

Niemeyer, J., Wegner, J.D., Mallet, C., Rottensteiner, F., Soergel, U., 2011. Conditional Random Fields for Urban Scene Classification with Full Waveform LiDAR Data, in: Stilla, U., Rottensteiner, F., Mayer, H., Jutzi, B., Butenuth, M. (Eds.), Photogrammetric Image Analysis, Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, pp. 233–244. https://doi.org/10.1007/978-3-642-24393-6_20

Niu, G., Jitkrittum, W., Dai, B., Hachiya, H., Sugiyama, M., 2013. Squared-loss mutual information regularization: A novel information-theoretic approach to semi-supervised learning, in: International Conference on Machine Learning. pp. 10–18.

Niyogi, P., 2013. Manifold regularization and semi-supervised learning: some theoretical analyses. J. Mach. Learn. Res. 14, 1229–1250.

Oliver, A., Odena, A., Raffel, C.A., Cubuk, E.D., Goodfellow, I., 2018. Realistic evaluation of deep semi-supervised learning algorithms, in: Advances in Neural Information Processing Systems. pp. 3235–3246.

Protopapadakis, E., 2016. Decision making via semi-supervised machine learning techniques. ArXiv160609022 Cs.

Protopapadakis, E., Doulamis, A., 2014. Semi-Supervised Image Meta-Filtering Using Relevance Feedback in Cultural Heritage Applications. Int. J. Herit. Digit. Era 3, 613–627. https://doi.org/10.1260/2047-4970.3.4.613

Protopapadakis, E., Doulamis, A., Doulamis, N., Maltezos, E., 2020. Semi-Supervised Fine-Tuning for Deep Learning Models in Remote Sensing Applications. ArXiv200600345 Cs.

Protopapadakis, E., Doulamis, A., Matsatsinis, N., 2014. Semi-supervised Image Meta-filtering in Cultural Heritage Applications, in: Ioannides, M., Magnenat-Thalmann, N., Fink, E., Žarnić, R., Yen, A.-Y., Quak, E. (Eds.), Digital Heritage. Progress in Cultural Heritaage: Documentation, Preservation, and Protection, Lecture Notes in Computer Science. Springer International Publishing, pp. 102–110.

Protopapadakis, E., Niklis, D., Doumpos, M., Doulamis, A., Zopounidis, C., 2019a. Sample selection algorithms for credit risk modelling through data mining techniques. Int J Data Min. Model. Manag. 11, 103–128. https://doi.org/10.1504/IJDMMM.2019.10019369

Protopapadakis, E., Schauer, M., Doulamis, A., Stavroulakis, G.E., Bhrnsen, J., Langer, S., 2015. Semi supervised identification of numerically simulated pile defects using graph label propagation, in: 8th GRACM International Congress on Computational Mechanics, N. Pelekasis, GE Stavroulakis, University of Thessaly Press, Volos, Greece.

Protopapadakis, E., Schauer, M., Pierri, E., Doulamis, A.D., Stavroulakis, G.E., Böhrnsen, J., Langer, S., 2016. A genetically optimized neural classifier applied to numerical pile integrity tests considering concrete piles. Comput. Struct. 162, 68–79. https://doi.org/10.1016/j.compstruc.2015.08.005

Protopapadakis, E., Voulodimos, A., Doulamis, A., 2018a. On the Impact of Labeled Sample Selection in Semisupervised Learning for Complex Visual Recognition Tasks. Complexity. https://doi.org/10.1155/2018/6531203

Protopapadakis, E., Voulodimos, A., Doulamis, A., 2017. Data sampling for semi-supervised learning in vision-based concrete defect recognition, in: 2017 8th International Conference on Information, Intelligence, Systems Applications (IISA). Presented at the 2017 8th International Conference on Information, Intelligence, Systems Applications (IISA), pp. 1–6. https://doi.org/10.1109/IISA.2017.8316454

Protopapadakis, E., Voulodimos, A., Doulamis, A., Camarinopoulos, S., Doulamis, N., Miaoulis, G., 2018b. Dance Pose Identification from Motion Capture Data: A Comparison of Classifiers. Technologies 6, 31.

Protopapadakis, Eftychios, Voulodimos, A., Doulamis, A., Doulamis, N., Dres, D., Bimpas, M., 2017a. Stacked autoencoders for outlier detection in over-the-horizon radar signals. Comput. Intell. Neurosci. 2017.

Protopapadakis, E., Voulodimos, A., Doulamis, A., Doulamis, N., Stathaki, T., 2019b. Automatic crack detection for tunnel inspection using deep learning and heuristic image post-processing. Appl. Intell. https://doi.org/10.1007/s10489-018-01396-y

Protopapadakis, E., Voulodimos, A., Doulamis, N., 2018c. Multidimensional Trajectory Similarity Estimation via Spatial-Temporal Keyframe Selection and Signal Correlation Analysis, in: Proceedings of the 11th PErvasive Technologies Related to Assistive Environments Conference, PETRA '18. ACM, New York, NY, USA, pp. 91–97. https://doi.org/10.1145/3197768.3201533

Protopapadakis, Eftychios, Voulodimos, A., Doulamis, N., 2017b. An investigation on multi-objective optimization of feedforward neural network topology, in: Information, Intelligence, Systems & Applications (IISA), 2017 8th International Conference On. IEEE, pp. 1–6.

Qi, Y., Shen, C., Wang, D., Shi, J., Jiang, X., Zhu, Z., 2017. Stacked sparse autoencoder-based deep network for fault diagnosis of rotating machinery. Ieee Access 5, 15066–15079.

Rottensteiner, F., 2013. ISPRS Test Project on Urban Classification and 3D Building Reconstruction: Evaluation of Building Reconstruction Results. Technical report.

Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1986. Learning representations by back-propagating errors. Nature 323, 533–536. https://doi.org/10.1038/323533a0

Schenkel, F., Middelmann, W., 2019. Domain Adaptation for Semantic Segmentation Using Convolutional Neural Networks, in: IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium. Presented at the IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium, pp. 728–731. https://doi.org/10.1109/IGARSS.2019.8899796

Schmidhuber, J., 2015. Deep learning in neural networks: An overview. Neural Netw. 61, 85–117. https://doi.org/10.1016/j.neunet.2014.09.003

Singla, A., Patra, S., Bruzzone, L., 2014. A novel classification technique based on progressive transductive SVM learning. Pattern Recognit. Lett. 42, 101–106. https://doi.org/10.1016/j.patrec.2014.02.003

Talbi, E.-G., 2009. Metaheuristics: From Design to Implementation. John Wiley & Sons.

van Engelen, J.E., Hoos, H.H., 2020. A survey on semi-supervised learning. Mach. Learn. 109, 373–440. https://doi.org/10.1007/s10994-019-05855-6

von Kügelgen, J., Loog, M., Mey, A., Schölkopf, B., 2019. Semi-supervised learning, causality and the conditional cluster assumption. ArXiv Prepr. ArXiv190512081.

Voulodimos, A., Doulamis, N., Doulamis, A., Protopapadakis, E., 2018. Deep learning for computer vision: A brief review. Comput. Intell. Neurosci. 2018.

Wei, Y., Yao, W., Wu, J., Schmitt, M., Stilla, U., 2012. Adaboost-based feature relevance assessment in fusing lidar and image data for classification of trees and vehicles in urban scenes. ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci. 1, 323–328.

Zhang, W., Qi, J., Wan, P., Wang, H., Xie, D., Wang, X., Yan, G., 2016. An Easy-to-Use Airborne LiDAR Data Filtering Method Based on Cloth Simulation. Remote Sens. 8, 501. https://doi.org/10.3390/rs8060501

Zhu, X., 2003. Semi-supervised learning using gaussian fields and harmonic functions, in: Proceedings of the 20th International Conference on Machine Learning (ICML-2003). p. 912.