BABEŞ-BOLYAI UNIVERSITY
FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

An extensive performance analysis of data engineering and model design in deep learning

PhD thesis summary

PhD student: Vlad-Ioan Tomescu
Scientific supervisor: Prof. dr. Czibula Gabriela

2022

Keywords: Deep learning, machine learning, feature engineering, convoloutional neural networks, depth maps, software defect prediction

Contents

Thesis table of contents

List of publications

The ranking of publications was performed according to the CNATDCU (National Council for the Recognition of University Degrees, Diplomas and Certificates) standards applicable for doctoral students enrolled after October 1, 2018. All rankings are listed according to the classification of journals[1] and conferences[2] in Computer Science.

Publications in Web of Science - Science Citation Index Expanded

[MVIC22] Diana-Lucia Miholca, **Vlad-Ioan Tomescu**, Gabriela Czibula. *An in-depth analysis of the software features impact on the performance of deep learning-based software defect predictors*. IEEE Access, Volume 10, 64801 – 64818, 2022 (**2021 IF=3.476**, Journal IF Quartile Q2)

Rank B, 4 points.

Publications in Web of Science, Conference Proceedings Citation Index

[TCNta21] **Tomescu, Vlad-Ioan** and Czibula, Gabriela and Niţică, Ştefan. A study on using deep autoencoders for imbalanced binary classification *Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 25th International Conference (KES 2021)*, Poland, Volume 192, 2021, Pages 119-128 (**indexed Web of Science**)

Rank B - CORE2021, 4 points.

[MCT20b] Diana-Lucia Miholca, Gabriela Czibula, **Vlad Tomescu**. COMET: A conceptual coupling based metrics suite for software defect prediction *Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 24nd International Conference (KES 2020)*, Budapest, Hungary, Procedia Computer Science, Volume 176, 2020, Pages 31–40. (**indexed Web of Science**)

Rank B - CORE2020, 4 points.

[CTC21] George Ciubotariu, **Vlad-Ioan Tomescu**, Gabriela Czibula. *Enhancing the performance of image classification through features automatically learned from depth-maps*, 13th International Conference on Computer Vision Systems (ICVS), September 22-24, 2021, LNCS 12899, Pages 68–81 (**indexed Scopus**).

Rank C - CORE2021, 2 points.

[1] https://uefiscdi.ro/premierea-rezultatelor-cercetarii-articole
[2] http://portal.core.edu.au/conf-ranks/

[NtaCT20] Ştefan Niţică, Gabriela Czibula, **Vlad-Ioan Tomescu**. *A comparative study on using unsupervised learning based data analysis techniques for breast cancer detection*. IEEE 14th International Symposium on Applied Computational Intelligence and Informatics, SACI 2020, Timisoara, Romania, 2020, Pages 99–104. (**indexed Web of Science**)

Rank D - CORE 2020, 1 point.

[Tom20] **Vlad-Ioan Tomescu**. *$FoRConvD$: An approch for food recognition on mobile devices using convolutional neural networks and depth maps*. IEEE 14th International Symposium on Applied Computational Intelligence and Informatics, SACI 2020, Timisoara, Romania, 2020, Pages 129–134. (**indexed Web of Science**)

Rank D - CORE 2020, 1 point.

Publications score: 16 points.

Introduction

The main research domain of this doctoral thesis is *deep learning*. The PhD thesis is entitled "An extensive research into the performance of data analysis and model design in deep learning" and aims to develop new ways to improve classification performance, both by employing smart data analysis and by designing efficient models.

Deep learning is a type of Machine Learning that is renown for having complex model architectures and training method and using lots of data. Deep learning models can in theory mimic any mathematical function and with the right training could correctly predict the class of a data instance. Therefore that would result in making any sort of data analysis and feature engineering, traditionally used for classical machine learning, obsolete. However it is a known fact that not all deep learning models perform equally well and even though most of them could mimic any function, many times their training does not reach it, but rather converges to a function corresponding to a local minimum.

This raises the question whether data analysis is still relevant in classification tasks, even though it is paired with a powerful and complex deep learning model. It hints to the possibility that smart feature engineering would smoothen the loss function plane of the model, making it easier for its optimiser to find the global minimum, or even a better local one. There is also a case to be made that not all data is equally relevant and some data sets are more difficult than others due to them containing lots of irrelevant data. Since many times the data sets are generated either automatically from all available data, or by a professional with no machine learning experience from an applied field, research into data analysis could eventually lead to better data extraction.

The next consideration are given to the deep learning models themselves. Since data can origin from various applied fields, that could lead to either some models being better at classification on a particular data sets, but also some being superior overall. Therefore it is vital to understand the reasoning behind choosing a model for a specific task. Factors of importance regarding deep learning models include its architecture and training methods, their relative relevance being part of the research. The entirety of these factors will be called model design and referred to as such, throughout the thesis.

The two complementary research directions dictate the main research objectives:

RO1 Analysing the relevance of data analysis and feature engineering in deep learning.

RO2 Determining the criteria for choosing a deep learning model for a particular classification task.

Since the research objectives are very general, answering them would require an extensive analysis into various different domains of application.

Domains of application

We have chosen 3 domains of application: Computer Vision, Software Defect Prediction and Breast Cancer Detection. The reasoning behind is twofold. First, while the application domains are very different and shows that the research is extensive, the nature of the data itself is relatively similar, leading to some models being applied to multiple domains and resulting in their generality. The second reasoning is due to the importance of the domains and the significant consequences that arise from correct predictions.

Computer Vision (CV) aims to cover a wide range of visual tasks in order to automate the processes of decision making in domains such as *autonomous driving*, *robotic process automation*, *product quality control*, or creating virtual environments for Virtual Reality/Augmented Reality. Recently, all CV tasks are performed using *Deep Learning* (DL) models that consist of multiple types of layers for processing input images at different resolutions. The most popular architecture in image processing is the one of *Deep Convolutional Neural Networks* (DCNN) which has the *convolution* as core concept. The aim of such networks is to encode pictures' spatial information with the help of convolutions while decreasing the resolution of images in order to grasp more of the scene context. Such convolutions are intensely researched in order to maximise the amount of information extracted and minimise the computational cost of network training.

Software Defects Prediction represents an essential activity during software development, as it contributes to continuously improving *software quality*. By detecting defect-prone instances in software systems, software defect prediction contributes to improving software development. *Software Defects Prediction* (SDP) also identifies defects in new versions of those systems, therefore also being considered an essential activity during software maintenance and evolution. SDP is considered of great importance in software engineering, as it contributes to continuously improving the *software quality*. Developing high quality software systems is expensive and, therefore, SDP is used for increasing the cost effectiveness of quality assurance and testing. By detecting fault-prone modules, SDP helps to allocate the effort so as to test more thoroughly those modules.

As stated by the World Health Organisation, *breast cancer* (BC) is the most frequent form of cancer among women, being responsible for 15% of all cancer-related deaths in this group. The major goal of a *Machine Learning* (ML) model is to assist physicians by detecting hidden patterns of the disease in the data it was trained on. This assistance could make the work of experts more efficient, without loss of accuracy. The major challenges in BC detection are to maximize the true positive detection (i.e. to maximize the sensitivity the classifier), but in the meanwhile to minimize the false positive detection (or, equivalently to maximize the specificity - true negative rate - of the BC classifier). A false positive represents an abnormality discovered by the ML classifier, but which is not cancer. Minimizing the false positive rate may lead to reducing patients traumas (panic, unnecessary interventions or treatments) as well as decreasing the cost of treatments (e.g. unnecessary extra mammograms or tests). Therefore, Machine Learning models start to be used more often in order to help medical experts with the identification of early forms of breast cancer. With the development of computational power, Deep Learning models start to replace classical Machine Learning ones.

Original Contributions

Our original contributions can be organised into 2 types. The first is data related, feature selection and engineering that leads to an increase in performance. Second is model related, comparisons between different model designs on a particular data set. The contributions are split among the 3 main domains of application:

1. *Computer vision* is a domain that takes input images and uses deep learning models, especially *Convolutional Neural Networks* (CNNs), to perform a variety of tasks, including classification, the focus of study. In the own original papers [Tom20] and [CTC21], there are presented 2 experiments on images including depth data. The first one includes building a system that would eventually determine the mass of a food item, by determining the class and volume separately. The second experiment analyses the performance improvement resultant from adding depth data to classification. Therefore, the original contributions are:

   (a) The data type of focus in the computer vision research is depth data. Depth data can be extracted together with the original RGB image, using various methods, ranging from stereo to infrared sensors. It was chosen for its versatility and analyze the various uses of depth maps, both as extra features that can improve the performance in a classification task, as well as standalone means of volume reconstruction. The results show that simply adding depth data to an image and passing it through a classifier could greatly improve the performance, while further feature extraction and combination could improve it even more. Even when not used for Machine Learning, depth data can be used for volume reconstruction and estimation, with results close to the real values, a complementary task to classification.

   (b) On the model side, two classifiers with different architectures were used. One is a state of the art Deep Learning classifier, suitable for large data sets with many classes, while the second belongs to the classical Machine Learning. The first classifier was chosen in order maximize the performance on a difficult task, while the second was sufficient for binary classification, while also helping to emphasize the performance brought by the depth maps. The deep learning model, while complex, is designed to be lightweight enough to be compatible to live applications. It is a state of the art model, both in terms of architecture (Efficient [TL19]) and training method (Fastai [H+18]). The second model is a multi-layer perceptron, a standard ML model whose implementation available in scikit-learn [Sci21] serves to emphasise the reproducibility of the depth map experiments.

2. *Software defect prediction* is used to determine whether software entities are defective or not. In the own original paper, [MVIC22] various models and data features are researched, while in the other paper [MCT20a] a new set of features are introduced and their performance analysed. Therefore, the original contributions can be regarded as:

   (a) On the data side, an in-depth analysis on the software features' impact on the performance of deep learning-based software defect predictors has been conducted. There is also an extension of a large-scale feature set proposed in the literature for detecting defect-proneness, by adding conceptual software features that capture the semantics of the source code, including comments. The conceptual features are automatically engineered using Doc2Vec and LSI, artificial neural network based prediction models. The research even goes one step further in feature engineering and designing an own set of metrics, called

COMET. This set is designed to be relevant in helping identify defective instances and is compared to the standard metric set used in the literature, the Promise metrics [SM15], resulting in an improvement in performance.

(b) On the model side, a broad evaluation is performed on the Calcite software system highlights a statistically significant improvement obtained by applying deep learning-based classifiers for detecting software defects when using conceptual features extracted from the source code for characterizing the software entities. Furthermore, a series of experiments are designed to evaluate the performance of the newly introduced COMET metrics on 7 data sets. These include a correlation-based analysis, unsupervised learning representation and supervised learning binary classification.

3. *Breast cancer detection*, due the nature of its data, is an imbalanced classification task. The data analysis part includes dimensionality reduction using unsupervised learning, while the model design part introduces a new autoencoder based binary classifier.

   (a) Since the breast cancer data sets are highly imbalanced, the data analysis part focuses on how that could potentially affect performance. Therefore comes the proposal of three *unsupervised learning* (UL) models for finding patterns in the data. *t-Distributed Stochastic Neighbor Embedding* (t-SNE), AEs and *self-organizing maps* (SOMs) were used as non-linear dimensionality reduction techniques and for further unsupervised classification (i.e clustering). Experiments were performed on three BC data sets (two publicly available from the UCI repository [WSMa] and one representing SERS data previously used in the literature [CMM+15]).

   (b) As part of the model design, there is an introduction of a binary classifier based on autoencoders. Since autoencoders are trained to recognise and reproduce a particular set of instances, they can be used on all those of a particular class. Therefore, the first model version would contain an autoencoder capable of recognising one class, but not the other, resulting in a binary classifier. However, it is possible to train two autoencoders, one on each class and at inference time they would both give a loss function value, based on how well each recognised the instance. The decision would then be the class of the autoencoder with the smaller loss. It is possible to go even further and consider the pairs of losses from the two autoencoders as 2D points in a plane and the decision boundary to be determined by another classifier, in this case a *Support Vector Machine (SVM)* [PS20a]. Then, the system consisting of two autoencoders would act as a feature extractor from data, effectively belonging to the first, data related, type of original contributions.

Thesis Structure

The rest of the thesis is structured as follows. Chapter 1 gives a background on the models used and the importance of the problems. Since some models were employed in multiple experiments, their overall description has been centralised in Section 1.1. Section **??** gives a detailed explanation of each approached problem from the applied fields, as well as the related work from the literature.

Chapter 2 gives all the experiments, results and discussion from the research into Computer Vision. Section 2.1 gives insight into an application designed to determine the mass of a food item, by determining its volume and food class, while Section 2.2 researches the relevance of depth maps in improving indoor-outdoor classification.

Chapter 3 shows the contributions in the applied field of Software Defect Prediction. Various features and models have been analysed in Section 3.1, while section 3.2 introduces a new set of features called COMET.

Chapter 4 is related to Breast Cancer Detection. It starts by presenting in Section **??** the data sets used in all the experiments. In Section 4.1 various models are used for Breast Cancer Detection, while Section 4.2 introduces a new approach in using autoencoders for binary classification.

The thesis ends with a Conclusions Chapter and a list of References.

Chapter 1

Background

In this chapter we are giving context to the approached problems from our thesis. Most of our original contributions are supervised and unsupervised learning-based approaches, with some models being applied across various fields. Therefore, we are giving an overview of the Machine Learning models in Section 1.1. This chapter of the thesis also presents the description and importance of the approached problems, as well as a literature review describing similar related work.

1.1 Machine Learning models used

In this section we are presenting the models that were used in our study. The main roles they are required to fulfil are: qualitative analysis, feature extraction and data classification. A successful qualitative analysis should match the numerical results of classification, while extracting feature and feeding them into a classifier should improve its overall metrics. While some models excel only at one role, there are some which can be used in multiple situations, with considerable success. More details on their applications can be found in Chapter 2.

1.1.1 Unsupervised learning

t-SNE [vdMH08] is a nonlinear dimensionality reduction technique that aims at maximizing the similarity between the probability distribution in the original input space and the probability distribution in the low-dimensional space. The distances between two data points are converted into conditional probabilities and Kullback-Leibler divergence is used to express the similarity of two probability distributions. Through a hyperparameter called *perplexity*, t-SNE preserves the local structure of the data, while also maintaining its global structure.

Self-organising maps (SOMs) are unsupervised learning models widely used in the ML literature for data visualisation and nonlinear dimensionality reduction, as well as for clustering. The SOM [Koh13] model is connected to the *competitive learning* paradigm. The so called *map* is usually a 2D lattice of neurons obtained by a non-liner mapping of the high dimensional input instances in a 2D space. A main characteristic of the SOM mapping is that of preserving the topological properties of the input space, thus instances which are similar in the high-dimensional input space are mapped on neurons that are close to each other [KOS09]. Due to this property, a trained SOM provides clusters of similar input instances.

1.1.2 Autoencoders

An *autoencoder* (AE) [GBC16] is a *self-supervised* feed forward neural network which aims to learn the identity function, more specifically to recreate the input. There are two main components of an AE: (1) the *encoder* which maps the n-dimensional input space into an m-dimensional hidden space; and (2) a *decoder* which learns to reconstruct the original input space from the hidden space. If the dimensionality of the hidden space is less than the dimensionality of the input space, the encoder may be used for unsupervised dimensionality reduction, i.e the hidden state express a meaningful low dimensional representation of the input data.

1.1.3 Convolutional Neural Networks

The Convolutional Neural Networks are a specialized type of Neural Networks that gets its name from using a layer called convolutional layer. Their main application is on two and three dimensional image data, although they have have also been successfully used on different types of data, like tabular. CNNs are long chains of convolutional layers, that are usually accompanied by activation functions for non-linearity and Batch Normalization layers for smoother training. They are trained using back propagation. To solve the problem of the vanishing gradient, modern CNN architecture contain skip connections and to avoid overfitting, drop connections are employed. If the task is classification, the CNN ends with a fully-connected layer and a softmax activation function.

1.1.4 Support Vector Machines

Support vector machines (SVMs) are supervised learning methods used for both classification and regression [PS20b]. Given a set of high dimensional real valued vectors (data points in $\mathbb{R}^d$, an SVM constructs an optimal separating hyperplane or set of hyperplanes in a high dimensional space, a good separation being achieved by the hyperplane(s) with the maximal *functional margin* (i.e. having the largest distance to the closest training data points of any class). Due to the large margin, SVMs are known as large margin classifiers. The intuition is that by maximizing the margin of the classifier, the risk of uncertain classification decisions (i.e. data points near the decision boundary) will be minimized.

The SVM optimisation problem can be formulated as finding an optimal separating hyperplane (i.e. leaving the largest possible fraction of points of the same class on the same side of the hyperplane and maximizing the distance of each class from the hyperplane) and minimizing the probability of misclassifying the training instances and the unseen test instances. SVMs are known to implement automatic complexity control for avoiding overfitting and due to the large margins they are simple classifiers, even if they have a lot of hyperparameters.

1.1.5 Multi-Layer Perceptron

The Multi-Layer Perceptron (MLP) [AC20] is a type of Artificial Neural Network, that is formed of multiple fully connected layers. It contains an input layer, output layer and at least one hidden layer, which contain nodes that are all connected with each other. The network contains weights at each node, that can be trained using back-propagation with the chain rule and then make prediction at inference time. The training is done using an optimiser like Stochastic Gradient Descent (SGD) or Adam, until convergence is reached. Activation functions are usually introduced to add non-linearity to the system, which in theory could then simulate any function. Choosing the appropriate size, based on the difficulty of the data set, helps avoid undesirable situations like over-fitting and under-fitting. MLP can be used on multiple types of data.

Chapter 2

Deep learning models in Computer Vision

The two main research directions of our thesis are: (1) analysing how the nature of the data impacts performance in classification and (2) how to design optimal models for specific tasks on a particular data set. Since this is an extensive analysis and the results should generalize, we are researching various domains of application very different from one another. One of these domains is Computer Vision, which is presented in this chapter.

Computer Vision (CV) aims to cover a wide range of visual tasks in order to automate the processes of decision making in domains such as *autonomous driving*, *robotic process automation*, *product quality control*, or creating virtual environments for Virtual Reality/Augmented Reality. Recently, all CV tasks are performed using *Deep Learning* (DL) models that consist of multiple types of layers for processing input images at different resolutions. The most popular architecture in image processing is the one of *Deep Convolutional Neural Networks* (DCNN) which has the *convolution* as core concept. The aim of such networks is to encode pictures' spatial information with the help of convolutions while decreasing the resolution of images in order to grasp more of the scene context. Such convolutions are intensely researched in order to maximise the amount of information extracted and minimise the computational cost of network training.

Therefore, on the model side, we are studying various CNN architectures and training methods, while on the data side, we are focusing our research on the utility of depth data in the context of Deep Learning applied to Computer Vision. Our original contributions can be summarized as:

1. Two classifiers with different architectures were used. One is a state of the art Deep Learning classifier, suitable for large data sets with many classes, while the second belongs to the classical Machine Learning. The first classifier was chosen in order maximize the performance on a difficult task, while the second was sufficient for binary classification, while also helping to emphasize the performance brought by the depth maps.

2. We analyze the various uses of depth maps, both as extra features that can improve the performance in a classification task, as well as standalone means of volume reconstruction. Our results show that simply adding depth data to an image and passing it through a classifier could greatly improve the performance, while further feature extraction and combination could improve it even more. Even when not used for Machine Learning, depth data can be used for volume reconstruction and estimation, with results close to the real values, therefore making a case for its versatility.

The remainder of the chapter is organized as follows:

Section 2.1 as a whole is based on our original paper [Tom20] and introduces a system that would use vision data to eventually determine the mass of a food item. The system needs to be lightweight, yet performant, in order to be applied to a mobile device. Section 2.2 represents another original paper [CTC21], which researches the utility of depth data in improving classification on indoor-outdoor images.

2.1 $FoRConvD$: An approch for food recognition on mobile devices using convolutional neural networks and depth maps

Proper nutrition is fundamental for a healthy lifestyle, especially in our modern world where many diseases could be avoided with the right diet. We have introduced [Tom20] an approach $FoRConvD$ (*Food Recognition using Convolutional neural networks and depth maps*) for detecting types of food and their mass on mobile devices, by using only the phone camera. Our approach consists of two main components: *food type detection* and *volume estimation*. The detection of the food type is done with EfficientNet, a state-of-the-art *convolutional neural network* model suitable for mobile platforms, trained on a data set of over 80000 images and 382 classes. As far as it is known, this is the first time EfficientNet was used on a large food data set with many classes.

The method used to estimate the volume is called *depth map fusion* and involves taking different images from various angles, along with their depth maps and computing a 3D model of the object. It is split into four tasks: *model reconstruction*, *point cloud fusion*, *point cloud to voxel grid* and *volume calculation*. The method introduced in this paper converts depth maps to point cloud directly and uses a point cloud merging technique called *Iterative Closest Point* (ICP) [CM92] [BM92] to get the fused model. The point cloud is then given volume by converting it into a voxel grid and that volume is then determined using summation of finite surface elements and multiplying them by their respective height.

A challenge in our approach was making the ICP more robust against translation and rotation errors in the expected environment: an object on a planar surface. This was achieved by splitting the point clouds into object and plane and performing multiple ICP iterations in a specific way. Also, in order to best determine the volume, the model has undergone various transformations including a noise removal. Another factor that needed to be optimized is the size of the unit volume. If it is too small, the point cloud is not dense enough, leaving holes, while if it is too large, unwanted corners of the unit cubes add redundant volume.

The reliability and accuracy of the method proposed for food volume estimation is empirically proven by the experimental results, resulting in a slight volume overestimation of 0% -10%, depending on the shape of the object.

2.2 Enhancing the performance of image classification through features automatically learned from depth-maps

In our original paper [CTC21] we approached a problem from Computer Vision, that of classifying indoor and outdoor images using *machine learning* and *deep learning* models. To do so, we have performed an unsupervised learning based analysis with the aim of determining the relevance of depth maps in the context of classification. For further tests to decide on the granularity of information extraction means, features were aggregated from sub-images of different sizes from DIODE data set to compare multiple scales of region attention. Four feature sets were proposed and comparatively

analysed in the context of indoor-outdoor image classification. To empirically confirm the advantage of using features automatically learned from depth maps, the features were fed into a supervised classification model. The performance of the classification using the proposed feature sets is then compared with the results of existing related work, highlighting the clear advantage of using features encoding depth information. Overall accuracy improvements consist of 18.8% for the machine learning approach and 1.1% for the deep learning one when depth information was added to the data.

Chapter 3

Deep learning models for software defect prediction

Software Defects Prediction represents an essential activity during software development, as it contributes to continuously improving *software quality*. By detecting defect-prone modules in new versions of a software system, software defect prediction contributes to improving software maintenance and evolution.

Software Defects Prediction (SDP) consists in identifying defective software components, being considered an essential activity during software development. It represents the activity of identifying defective software modules in new versions of a software system [hCMZ11]. SDP is considered of great importance in software engineering, as it contributes to continuously improving the *software quality*. Developing high quality software systems is expensive and, therefore, SDP is used for increasing the cost effectiveness of quality assurance and testing [CMC14]. By detecting fault-prone modules in new versions of a software system, SDP helps to allocate the effort so as to test more thoroughly those modules [hCMZ11].

SDP assists measuring project evolution, supports process management [CZ01], predicts software reliability [Zhe09], guides testing and code review [hCMZ11]. All these activities allow to significantly reduce the costs involved in developing and maintaining software products [HM18]. Moreover, particularly in the case of safety-critical systems, SDP helps in detecting software anomalies with possible negative effects on human lives.

As the software systems complexity increases, the number of software defects generated during the software development will also significantly increase. This growing complexity of software projects requires an increasing attention to their analysis and testing. Numerous researches from the SDP literature are based on mining historical and code information during the software development process and then building a prediction model (statistical, machine learning-based or other) to predict software defects [ZZYW20].

There are two main types of software defect predictors in the SDP literature: within-project and cross-project defect predictors. For the within-project defect prediction, some defect data from a software project is used as training set, to build the prediction model, while the remaining data is used to test the performance of the model [ZZYW20]. On the other hand, the cross-project defect predictor allows predicting defects in a target software system based on historical data from other systems. Therefore, they are more general and allow predicting defects in projects with limited historical data. The cross-project SDP models are created based on data extracted from a set of software systems, but applied and tested on different software systems.

In our original paper [MVIC22], we have conducted an in-depth analysis on the software features'

impact on the performance of deep learning-based software defect predictors. We further extend a large-scale feature set proposed in the literature for detecting defect-proneness, by adding conceptual software features that capture the semantics of the source code, including comments. The conceptual features are automatically engineered using Doc2Vec and LSI, artificial neural network based prediction models. A broad evaluation performed on the Calcite software system highlights a statistically significant improvement obtained by applying deep learning-based classifiers for detecting software defects when using conceptual features extracted from the source code for characterizing the software entities.

In our second original paper regarding software defect prediction [MCT20a], we go one step further in feature engineering and design our own set of metrics. They are also based on conceptual coupling, the semantic relationship between entities at the source code level. Features are extracted from entities using Doc2Vec and LSI and aggregation is performed on them, resulting in specific measures (*maximum*, *mean* and *standard deviation*) that are used the form a new set of metrics called COMET. This set is designed to be relevant in helping identify defective instances and is compared to the standard metric set used in the literature, the Promise metrics [SM15], in a series of experiments: (1) correlation-based analysis, (2) unsupervised learning representation and (3) supervised learning based classification.

As mentioned in the previous chapters, the main goal of our thesis is to identify the relevance of feature engineering, as well as model design, in the context of machine learning in general and more specifically deep learning. Section 3.1 focuses more on the model side, comparing different DL and classical ML models, with DL architectures trained using complex methods emerging superior. However, it also contains a significant portion of feature engineering, which is used to improve model performance. In contrast, Section 3.2 analyses more the feature side, by introducing a new set of features, which is thoroughly tested and compared to the literature, resulting in an improved performance over the current standard.

The results from this chapter show that both feature engineering and model design are important for achieving high performance in the domain of software defect prediction.

3.1 An in-depth analysis of the software features' impact on the performance of deep learning-based software defect predictors

Despite its importance and extensive applicability, SDP remains a difficult problem, especially in large-scale complex systems, and a very active research area [HBB+11]. The conditions for a software module to have defects are hard to identify and, therefore, the defect prediction problem is computationally difficult. From a supervised learning viewpoint, predicting defects is a difficult task as the training data used for building the defect predictors is highly imbalanced. The faulty modules in a software system are greatly outnumbered by the error-free modules. Thus, conventional learning algorithms are often biased towards the non-defective class. Another important issue in SDP is related to the features used for characterizing software entities (an entity may be a component, class, module depending on the targeted level of granularity). As, generally, in *machine learning* (ML), the classical approach is to use manually engineered features, traditional software metrics are usually used in SDP as features characterizing the given software entities. Literature reviews in SDP revealed that about 87% [Mal15] of the case studies used procedural or object-oriented metrics, while more than 95% [HTG19] of studies for cross-project defect prediction relied on software metrics. A third issue is that, while defects can be of various types (e.g. numeric errors, complexity issues, pointer issues, etc.) and it is very likely that each defect type has its own properties, existing SDP approaches

consider all defect types together and try to come up with a universal defect prediction model.

The two prevalent research directions in the SDP literature are: proposing software features relevant to the discrimination between defective and non-defective software entities and building or recommending high-performing defects prediction models.

When it comes to large amounts of data, deep learning models are some of the best at making accurate predictions, regardless of the origin of that data. As long as there is correlation between the input information and the output, the models will discover it. In order to use deep learning, the input software features are written in tabular form, a data form that has been extensively researched and for which many models are available [B+21].

In the original paper [MVIC22], we followed both above-mentioned directions. Our work originated from three research questions:

RQ1 Could the performance of predicting software defects be enhanced by enlarging the software features proposed for SDP with conceptual features extracted from the source code? Which is the most appropriate feature set to distinguish between defective and non-defective software entities and to what extent is the performance improvement significant from a statistical perspective?

RQ2 Could the relevance of the conceptual-based software features be empirically sustained by both unsupervised and supervised analyses conducted on a large scale software system?

RQ3 Does deep learning-based defect prediction bring a statistically significant improvement when compared to traditional supervised classifiers?

With these research questions in mind, we have performed an in-depth analysis of the software features' impact on the performance of software defect predictors. We have extended the large collection of SDP features proposed by Herbond et al. [HTTL22] with Doc2Vec-based conceptual software features that capture the semantics of the source code (including comments). An extensive study conducted on different versions of the Calcite data set highlight, through both unsupervised and supervised learning-based analyses, that the conceptual features bring a statistically significant improvement on the performance of SDP. As a second line of research, we have extensively examined the effect of the feature set identified as the most relevant on the performance of various defect predictors. To the best of our knowledge, a study similar to ours has not been proposed in the literature, so far.

3.2 COMET: A new set of metrics for software defect prediction using conceptual coupling

The area of Software Defect Prediction research is vast, due to the complex and difficult nature of its task. While there are many possible research directions that could lead to a better identification of defects in software systems, 2 of the main ones, as we have seen in the previous section, are identifying relevant metrics that are correlated to the presence of defects and creating classification models that would extract significant information from those metrics. In this section we are considering mostly the first direction, by focusing our research on how *conceptual coupling* could help extract superior metrics from the source code and it is based on our original paper [MCT20b].

Conceptual coupling refers to the relationships between software entities at the semantic level (classes, methods) and its relevance in the context of SDP has been described in the literature [WLT16, MC19]. Therefore a new set of metrics based on conceptual coupling is introduced in this chapter,

which is called COMET. In order to validate the relevance of the COMET metrics for SDP, multiple experiments were conducted including a correlation based analysis, unsupervised learning modeling and supervised binary classification. A direct comparison was done to the standard Promise metrics (available here [dat]), that are widely used in the literature. The results show a better performance of the COMET metrics over the Promise ones, marking them as more relevant in the context of SDP.

Our research directions could be summarised in 2 distinct questions, each with its afferent subsection that would focus on answering it. These are:

RQ1 How can conceptual coupling be used to generate a suite of metrics, that could be able to predict whether software systems contain any bugs? Regarding this, a new set of 36 metrics in defined, called COMET.

RQ2 What is the performance of this COMET metric set and how does it relate to others from literature? To determine this, multiple experiments are conducted, including a correlation based analysis, as well as supervised and unsupervised learning approaches.

Chapter 4

Deep learning models for breast cancer detection

The third domain of application is medicine, more specifically breast cancer detection, where the task at hand is to successfully predict the presence of malign tumors in patients. The nature of the data sets containing patient information coupled with the relatively rare occurrence of cancer in the general population results in an imbalanced supervised classification. As an application domain we consider *breast cancer* detection, because it is a classification problem of great interest in the medical domain. According to the World Health Organisation, breast cancer represents the primary cause of cancer mortality in women.

Within the *supervised learning* domain, *imbalanced classification* represents a challenge for supervised learning, as an unequal distribution of classes in the training data set is mainly connected to poor predictive performance for the minority class. However, usually the minority class is the most relevant one, from a practical perspective. But due to the imbalancement of the training data, the classification errors for the minority class are higher, as the classifiers are usually biased to predict the majority class.

In the broader context of our thesis, which aims both to research how the nature of the data affects the classification performance, as well as design models with improved performance, it can be observed that this chapter focuses more on model design.

Our original contribution in this chapter is twofold. First, we are using machine learning models in their traditional role, with improved performance, for example t-SNE for qualitative analysis. Second, we apply some models to different tasks successfully. Autoencoders are mainly used for feature extraction, but we also managed to employ them in binary classification, with performance comparable and sometimes better than other classifiers.

Regarding the data analysis part, we compare various breast cancer data sets and observe that the performance greatly depends on the nature of the data, with some data sets being more difficult than others.

This chapter introduces our contributions in the field of BC detection. Section 4.1 introduces three *unsupervised learning* models (*t-Distributed Stochastic Neighbor Embedding*, *autoencoders* and *self-organizing maps*) analysed in our original work [NtaCT20] with the aim of unsupervisedly detecting the classes of *benign* and *malignant* instances. Section 4.2 investigates the ability of *deep autoencoders* to learn patterns within the classes of *benign* and *malignant* instances proposes two autoencoders-based classification models for breast cancer detection. The approach described in Section 4.2 was introduced in our original work [TCNta21].

4.1 A comparative study on using unsupervised learning based data analysis techniques for breast cancer detection

As stated by the World Health Organisation, *breast cancer* is the most frequent form of cancer among women, being responsible for 15% of all cancer-related deaths in this group. A lot of research has been carried out, so far, in using various *machine learning* models for breast cancer prediction, ranging from conventional classifiers to deep learning techniques. Three *unsupervised learning* models (*t-Distributed Stochastic Neighbor Embedding*, *autoencoders* and *self-organizing maps*) were introduced and comparatively analysed in our original work [NtaCT20] with the aim of unsupervisedly detecting the classes of *benign* and *malignant* instances. Experiments performed on data sets previously used in the literature for breast cancer detection reveal a good performance of the proposed unsupervised learning models. The best performance was obtained using *autoencoders*, which provided values higher than 0.935 for the *area under the ROC curve* evaluation measure.

The contribution of the section is twofold [NtaCT20]. First, we are proposing three *unsupervised learning* (UL) models for unsupervised BC detection. *t-Distributed Stochastic Neighbor Embedding* (t-SNE), AEs and *self-organizing maps* (SOMs) were used as nonlinear dimensionality reduction techniques and for further unsupervised classification (i.e clustering). Experiments were performed on three BC data sets (two publicly available from the UCI repository [WSMa] and one representing SERS data previously used in the literature [CMM+15]). A second goal of our study was to comparatively examine the performance of the proposed UL models: t-SNE, AEs and SOMs were applied for mapping the original space of input instances in a two dimensional space. The obtained experimental results highlight a good correlation between the clusters unsupervisedly obtained in data (after applying the previously mentioned UL models) and the ground truth partitions (i.e. the known sets of *benign* and *malignant* instances). A slightly better performance was obtained using AEs, which provided an *Area under the ROC curve* value higher than 0.935 on all data sets. Whilst similar UL models were reported in the literature for analyzing the data from the UCI repository [WSMa], none of them were applied for analyzing SER spectra acquired on blood serum in order to detect breast cancer, as far as we know.

4.2 A study on using deep autoencoders for breast cancer classification

In this section we investigate the use of deep *autoencoders* (AEs) for improving the predictive performance for imbalanced binary classification problems. AEs are usually used in the literature in an unsupervised learning context to extract relevant features (e.g. from histopathology images) that are further fed to classifiers in order to detect anomalies [FZM18, XXHW14, XXL+15]. From a supervised learning perspective, AEs are mostly used as anomaly detectors [CCT21, RDBV20] and less often as binary classifiers, as in our proposal.

In our original paper [TCNta21] we investigated the ability of *deep autoencoders* to learn patterns within the classes of *benign* and *malignant* instances. Secondly, we propose and compare two autoencoders-based classification models for breast cancer detection.

The contribution of the section is threefold. First, we are investigating the ability of AEs to learn patterns within the classes of *benign* and *malignant* instances. Secondly, we propose and compare three AE-based classification models for BC detection, as follows. The first classifier proposed C_{1AE} is based on using autoencoders as one-class classifiers (an AE is trained on the set of *benign* instances and at the classification stage the *malignant* instances are detected as anomalies with respect to the class learned by the AE). The second classifier C_{2AE} is based on the idea of training two autoencoders,

one autoencoder for each class (*benign* and *malignant*) and then a new instance will be assigned to the class represented by the "closest" AE (i.e. the AE which provides the minimum loss value for the instance). Our third classifier $C_{2AE-SVM}$ enhances C_{2AE} with an SVM model used for deciding the decision boundary between the classes. Experiments will be conducted on three BC data sets (two publicly available from the UCI repository [WSMb] and one representing SERS data previously used in the literature [CMM+15]) with the goal of comparatively analyzing the performance of our models. To the best of our knowledge, the models introduced in this section are new in the ML-based BC detection literature.

To summarize, the study conducted in this section is organized around three research questions:

RQ1 Are AEs able to encode hidden relationships between instances (patients) belonging to the same class (*benign* or *malignant*)? How to use an AE for encoding hidden patterns in *benign* instances and for detecting *malignant* instances as being anomalies with respect to the trained AE?

RQ2 How to use two AEs (i.e. one AE for each of the *benign* and *malignant* classes) to supervisedly classify instances, based on the encoded relationships between the instances belonging to the same class?

RQ3 How performant are the approaches introduced for answering RQ1 and RQ2 and how does the approaches compare to similar existing work for BC detection?

We note that the binary classifier C_{2AE} introduced in this section can be easily extended to a multiclass classifier, C_{nAE} (e.g. for cancer staging or other multiclass classification problems). Assuming that multiple classes are given, the classifier would have one autoencoder for each class with the goal of recognizing the instances from that particular class. Afterwards, at the clasification stage, a new instance will be assigned to the class whose AE provides the best reconstruction for the instance (i.e. it minimizes the loss value for the instance).

Conclusions

The two main research directions of our PhD thesis were: (1) analysing the data in order to determine the extent of its importance in classification and (2) designing efficient models for improved performance. The first direction can be described as *data analysis and feature engineering*, while the second is entitled *model design*. The domains of application are *Computer Vision*, *Software Defect Prediction* and *Breast Cancer Detection*.

In Computer Vision, we focused our research on depth data. Since it can be extracted together with the RGB image, but it is not automatically done so, demonstrating its importance could lead to a shift in general vision data collection. Our research discovered that depth data is versatile. First, it can be used to improve classification performance, therefore answering our first research question. Simply adding depth data to RGB features has been shown to increase the accuracy from 0.69% to 0.88%, while using powerful deep learning semantic segmentation models would increase the performance even further.

Second, even if depth data is not used directly for classification, it can be used for a complementary task. In order to calculate the mass of a food item, the volume and density are required. Depth maps are employed for volume estimation and the density is standard for a food item, as long as its class can be determined from classification. With the volume calculation method, the results were within 10% of the actual values, very accurate for the required task. Therefore, the versatile usefulness of the depth data makes a strong case for its general extraction along RGB data in computer vision.

On the model side, while it is known that computer vision uses convolutional neural networks for classification, our research suggests that a model containing an EfficientNet architecture, trained using the Fastai *fit one cycle* method would be both lightweight and performant, therefore suitable for mobile applications. Sometimes however, the task is not to obtain state-of-the art performance on images directly, but rather have a model that is to help evidentiate the main original contribution: a new set of features. One such model should be readily-available and easy to implement, making the experiments reproductible. The Multi-Layer Perceptron was the choice to perform such a task, a standard ML model implemented in scikit-learn, that obtains good results when paired with intelligent feature engineering.

The Software Defect Prediction data is generated based on conceptual coupling, which represents the relationship between software entities at the source code level. There are multiple ways to extract information form the same code, but they are not equally relevant. On the calcite data sources, our experiments revealed a better performance of a set containing 60 Doc2Vec and LSI features than of one containing over 4000 features. The lower feature number would also mean a faster inference, which is relevant to live applications. Therefore it is proven that feature engineering is extremely relevant, even when the model used is a powerful deep learning one.

We even go one step further and design our own set of features called COMET, based on conceptual coupling and using Doc2Vec and LSI for extraction. The newly introduced set was thoroughly analysed and compared to the standard literature Promise set in various tests: correlation based analy-

sis, unsupervised learning representation and supervised learning binary classification. The COMET metrics outperformed the Promise ones, proving their relevance in the context of software defect prediction.

Multiple models were employed in our SDP experiments, both unsupervised and supervised, but one whose performance stands out is a deep learning classifier with an Artificial Neural Network architecture, trained using the Fastai *fit one cycle* method. It can be observed that Fastai trained models outperform other models, even those with the same architecture but different implementations. Furthermore, it appears that the *fit one cycle* method can be applied regardless of architecture and data type, from vision image data to vectorial tabular data, resulting in high performance.

For both *Software Defect Prediction* and *Breast Cancer Detection*, we encounter imbalanced data sets, due to the nature of the data generated from those domains of application. To quantify exactly much harder the tasks become under such conditions, we introduce a measure called difficulty, which relates to whether instances and especially positive ones, have a nearest neighbour from the same class. For our SDP experiments we also compare our results to a random guessing classifier, which for a balanced data set would have a 50% chance of correctly predicting a class. However, for the positive class (given by the precision metric), that chance can drop to as low as 3.3% in some cases, making the task of the models much more difficult.

The models employed for *Breast Cancer Detection* are generally the same as for the other domains of application (for example unsupervised learning analysis using t-SNE). However, we introduce a new autoencoder based binary classifier as our main original contribution from that domain of application. Autoencoders are used to reconstruct the original input and are trained on one particular class. The first classification based method proposed is having one autoencoder trained on the majority class and used for testing both type of instances. The ones from the class it was trained on would have a small loss function, while the others would have a higher one resulting in a differentiation between them.

The second model would consist of two autoencoders, each trained on one class, making predictions on all instances, the decision resulting from the smaller loss. Furthermore, the losses could be regarded as 2D points in a plane, the decision boundary then dictated by another classifier, resulting in the autoencoder based model essentially becoming a feature extractor. *Model design* would lead to *feature engineering*, both our research directions combined.

The conclusion that both *data analysis* and *model design* are important can be drawn from the extensive analysis in various distinct applied fields.

Regarding the first objective, RO1, based on *data analysis*, the main findings are: (1) Depth data is useful and versatile and should be extracted along other vision data and (2) Feature engineering is still very relevant even in the context of deep learning.

By pursuing the second objective, RO2, we can conclude that: (1) The fastai [H+18] training method is consistently effective across various types of data and (2) Autoencoders can be successfully used both for feature extraction and binary classification.

Bibliography

[AC20] S. Abirami and P. Chitra. Energy-efficient edge based real-time healthcare support system. *Advances in Computers*, 117(1):339–368, 2020.

[B+21] Vadim Borisov et al. Deep neural networks and tabular data: A survey. *CoRR*, abs/2110.01889, 2021.

[BM92] Paul J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(2):239–256, 1992.

[CCT21] Gabriela Czibula, Carmina Codre, and Mihai Teletin. Anomalp: An approach for detecting anomalous protein conformations using deep autoencoders. *Expert Systems with Applications*, 166:114070, 2021.

[CM92] Yang Chen and Grard G. Medioni. Object modelling by registration of multiple range images. *Image Vision Comput.*, 10(3):145–155, 1992.

[CMC14] Gabriela Czibula, Zsuzsanna Marian, and Istvan Gergely Czibula. Software defect prediction using relational association rule mining. *Information Sciences*, 264:260–278, 2014. Serious Games.

[CMM+15] Silvia Cervo, Elena Mansutti, Greta Mistro, Riccardo Spizzo, Alfonso Colombatti, Agostino Steffan, Valter Sergo, and Alois Bonifacio. Sers analysis of serum for detection of early and locally advanced breast cancer. *Analytical and Bioanalytical Chemistry*, 407:7503–7509, 2015.

[CTC21] George Ciubotariu, Vlad-Ioan Tomescu, and Gabriela Czibula. Enhancing the performance of image classification through features automatically learned from depth-maps. In Markus Vincze, Timothy Patten, Henrik I Christensen, Lazaros Nalpantidis, and Ming Liu, editors, *13th International Conference on Computer Vision Systems, ICVS 2021, Virtual Event, September 22-24, 2021, Proceedings*, volume 12899 of *Lecture Notes in Computer Science*, pages 68–81. Springer, 2021.

[CZ01] B. Clark and D. Zubrow. How good is a software: a review on defect prediction techniques. In *Software Engineering Symposium*, pages 1–35, Carneige Mellon University, 2001.

[dat] Tera-promise repository. http://openscience.us/repo/.

[FZM18] Yangqin Feng, Lei Zhang, and Juan Mo. Deep manifold preserving autoencoder for classifying breast cancer histopathological images. *IEEE/ACM transactions on computational biology and bioinformatics*, 17(1):91–101, 2018.

[GBC16] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[H+18] Jeremy Howard et al. fastai. https://github.com/fastai/fastai, 2018.

[HBB+11] Tracy Hall, Sarah Beecham, David Bowes, David Gray, and Steve Counsell. A systematic review of fault prediction performance in software engineering. *IEEE Transactions on Software Engineering*, 38(6):1276–1304, 2011.

[hCMZ11] Rui hua Chang, Xiaodong Mu, and Li Zhang. Software defect prediction using non-negative matrix factorization. *JSW*, 6(11):2114–2120, 2011.

[HM18] Jaroslaw Hryszko and Lech Madeyski. Cost effectiveness of software defect prediction in an industrial project. *Foundations of Computing and Decision Sciences*, 43(1):7 – 35, 2018.

[HTG19] Seyedrebvar Hosseini, Burak Turhan, and Dimuthu Gunarathna. A systematic literature review and meta-analysis on cross project defect prediction. *IEEE Transactions on Software Engineering*, 45(2):111–147, 2019.

[HTTL22] Steffen Herbold, Alexander Trautsch, Fabian Trautsch, and Benjamin Ledel. Problems with szz and features: An empirical study of the state of practice of defect prediction data collection. *Empirical Software Engineering*, 27(2), Jan 2022.

[Koh13] Teuvo Kohonen. Essentials of the self-organizing map. *Neural Networks*, 37:52–65, 2013.

[KOS09] Andreas Khler, Matthias Ohrnberger, and Frank Scherbaum. Unsupervised feature selection and general pattern discovery using self-organizing maps for gaining insights into the nature of seismic wavefields. *Computers & Geosciences*, 35(9):1757 – 1767, 2009.

[Mal15] Ruchika Malhotra. A systematic review of machine learning techniques for software fault prediction. *Applied Soft Computing*, 27:504 – 518, 2015.

[MC19] Diana-Lucia Miholca and Gabriela Czibula. Software defect prediction using a hybrid model based on semantic features learned from the source code. In Christos Douligeris, Dimitris Karagiannis, and Dimitris Apostolou, editors, *Knowledge Science, Engineering and Management -LNCS, volume 11775*, pages 262–274, Cham, 2019. Springer International Publishing.

[MCT20a] Diana-Lucia Miholca, Gabriela Czibula, and Vlad Tomescu. COMET: A conceptual coupling based metrics suite for software defect prediction. *Procedia Computer Science*, 176:31–40, 2020. Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 24th International Conference KES2020.

[MCT20b] D.L Miholca, Gabriela Czibula, and Vlad Tomescu. Comet: A conceptual coupling based metrics suite software defect prediction. In *Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 24nd International Conference (KES 2020)*, volume Procedia Computer Science, Vol. 176, pages 31–40, 2020.

[MVIC22] Diana-Lucia Miholca, Tomescu Vlad-Ioan, and Gabriela Czibula. An in-depth analysis of the software features impact on the performance of deep learning-based software defect predictors. *IEEE Access*, page submitted, 2022.

[NtaCT20] ¸Stefan Ni¸tică, Gabriela Czibula, and Vlad-Ioan Tomescu. A comparative study on using unsupervised learning based data analysis techniques for breast cancer detection. In *2020 IEEE 14th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, pages 99–104, 2020.

[PS20a] Derek A. Pisner and David M. Schnyer. Chapter 6 - support vector machine. In Andrea Mechelli and Sandra Vieira, editors, *Machine Learning*, pages 101–121. Academic Press, 2020.

[PS20b] Derek A. Pisner and David M. Schnyer. Chapter 6 - support vector machine. In Andrea Mechelli and Sandra Vieira, editors, *Machine Learning*, pages 101–121. Academic Press, 2020.

[RDBV20] Stefania Russo, Andy Disch, Frank Blumensaat, and Kris Villez. Anomaly detection using deep autoencoders for in-situ wastewater systems monitoring data, 2020.

[Sci21] Scikit-learn. Machine learning in Python, 2021. http://scikit-learn.org/stable/.

[SM15] Shirabad Sayyad and Tim Menzies. The PROMISE Repository of Software Engineering Databases, University of Ottowa, Kanada. School of Information Technology and Engineering, University of Ottawa, Canada, 2015.

[TCNta21] Vlad-Ioan Tomescu, Gabriela Czibula, and ¸Stefan Ni¸tică. A study on using deep autoencoders for imbalanced binary classification. In *Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 25th International Conference (KES 2021)*, volume Procedia Computer Science, Vol., page to be published, 2021.

[TL19] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *CoRR*, abs/1905.11946, 2019.

[Tom20] Vlad-Ioan Tomescu. $forconvd$: An approch for food recognition on mobile devices using convolutional neural networks and depth maps. In *2020 IEEE 14th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, pages 129–134, 2020.

[vdMH08] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[WLT16] Song Wang, Taiyue Liu, and Lin Tan. Automatically learning semantic features for defect prediction. In *Proc. of the 38th Int. Conf. on Softw. Engineering*, ICSE '16, pages 297–308, New York, NY, USA, 2016. ACM.

[WSMa] William H. Wolberg, W. Nick Street, and Olvi L. Mangasarian. Breast cancer wisconsin (diagnostic) data set.

[WSMb] William H. Wolberg, W. Nick Street, and Olvi L. Mangasarian. Breast cancer wisconsin (diagnostic) data set.

[XXHW14] Jun Xu, Lei Xiang, Renlong Hang, and Jianzhong Wu. Stacked sparse autoencoder (ssae) based framework for nuclei patch classification on breast cancer histopathology. In *2014 IEEE 11th international symposium on biomedical imaging (ISBI)*, pages 999–1002. IEEE, 2014.

[XXL+15] Jun Xu, Lei Xiang, Qingshan Liu, Hannah Gilmore, Jianzhong Wu, Jinghai Tang, and Anant Madabhushi. Stacked sparse autoencoder (ssae) for nuclei detection on breast cancer histopathology images. *IEEE transactions on medical imaging*, 35(1):119–130, 2015.

[Zhe09] Jun Zheng. Predicting software reliability with neural network ensembles. *Expert Systems with Applications*, 36(2):2116–2122, 2009.

[ZZYW20] Kun Zhu, Nana Zhang, Shi Ying, and Xu Wang. Within-project and cross-project software defect prediction based on improved transfer naive bayes algorithm. *Computers, Materials & Continua*, 63(2):891–910, 2020.