FACULTY OF ENGINEERING
AND ARCHITECTURE

Uncovering Patterns in Microscopic Images from Metallurgical Research with Unsupervised Learning Algorithms

Henri Vandendorpe
Student number: 01808971

Supervisor: Prof. dr. ir. Aleksandra Pizurica
Counsellors: Srdan Lazendic, Ir. Lode Duprez (OCAS - Arcelormittal Global R&amp;D Gent)

Master's dissertation submitted in order to obtain the academic degree of
Master of Science in Computer Science Engineering

Academic year 2022-2023

GHENT
UNIVERSITY

FACULTY OF ENGINEERING
AND ARCHITECTURE

Uncovering Patterns in Microscopic Images from Metallurgical Research with Unsupervised Learning Algorithms

Henri Vandendorpe
Student number: 01808971

Supervisor: Prof. dr. ir. Aleksandra Pizurica
Counsellors: Srdan Lazendic, Ir. Lode Duprez (OCAS - Arcelormittal Global R&amp;D Gent)

Master's dissertation submitted in order to obtain the academic degree of
Master of Science in Computer Science Engineering

Academic year 2022-2023

GHENT
UNIVERSITY

Permission of use on loan

Preface

I proudly present my masters dissertation, to conclude my studies at the Ghent University as a Computer Science Engineering student. This thesis represents the skills I have gained during my time as a student at this university.

I would like to thank my promoter: Prof. dr. ir. Aleksandra Pižurica for giving me a chance to work on this interesting subject and for supporting me with her knowledge on the subject. Additionally I would like to thank my counsellor Srđan Lazendić for his clear explanations in the technical aspects of this thesis and for helping me to grasp the necessary mathematical concepts. But also for support and encouraging words. I would like to thank my counsellor from OCAS: dr. ir. Lode Duprez, together with dr. ir. John Vande Voorde for their valuable insights in the metallurgical aspects of this thesis.

Last but not least I would like to thank my family and friends as without their support this thesis would not have been possible. Thank you for listening to my rants and supporting me through the highs and lows of this thesis.

Henri Vandendorpe

Uncovering Patterns in Microscopic Images from Metallurgical Research with Unsupervised Learning Algorithms

Henri Vandendorpe

Supervisor: Prof. dr. ir. Aleksandra Pižurica
Counsellors: Srđan Lazendić and ir. Lode Duprez

Master's dissertation submitted in order to obtain the academic degree of
Master of Science in Computer Science Engineering
Academic year 2022-2023

Abstract

Artificial intelligence has already been utilized in various areas of metallurgical research. However, as far as our knowledge extends, there has been no prior investigation on applying AI for pattern recognition of microscopic images of fracture surfaces to analyze the metallurgical properties of materials. This thesis aims to address this gap by developing a framework for clustering metallurgical images into groups with a similar toughness property. The first step of the proposed framework consists of splitting the images into patches of size 64×64. By employing and comparing different feature extraction techniques we obtain compact representations that contain the necessary information regarding the properties of each image patch. On these compact representations we applied various clustering algorithms to identify the optimal method for assigning these images into groups. The outcomes are evaluated using the Davies-Bouldin index. Minimizing this metric results in more compact and separated clusters. Using a majority vote, we combine the image patch cluster labels into a label for the whole image. Using these labels the toughness properties per cluster can be evaluated by visualizing the resulting toughness distributions using a boxplot. The gray level co-occurrence matrix feature extraction and a combination of the discrete wavelet transform and local binary patterns proved to be the two best feature extraction methods among the utilized techniques. When these feature vectors were used as input for k-means clustering, which is the best performing clustering algorithm that was investigated in this work, the metal samples could be separated based on their toughness property. In summary, this thesis presents a comprehensive framework for clustering metallurgical images based on their toughness properties. The use of various feature extraction techniques and clustering algorithms allows for effective analysis and classification of these images. The evaluation metrics and visualizations employed provide a clear understanding of the capabilities of the resulting models.

Index Terms — Pattern recognition, Feature extraction, Clustering, Metallurgy

Uncovering Patterns in Microscopic Images from Metallurgical Research with Unsupervised Learning Algorithms

Henri Vandendorpe
Supervisor: Prof. dr. ir. Aleksandra Pižurica
Counsellors: Srđan Lazendić and dr. ir. Lode Duprez

Abstract—**Artificial intelligence has already been used for a number of applications in metallurgical research, but to the best of our knowledge, no research has been done for pattern recognition of microscopic images of fracture surfaces to analyze the metallurgical properties of the material. In this thesis, we develop a framework to cluster metallurgical images into groups with a similar toughness property. We apply and compare feature extraction techniques to find a compact representation of the image contents and different clustering algorithms To assign the images into groups based on different similarity metrics. We will evaluate these result with the Davies-Bouldin index [1] and plot the resulting toughness distributions of the clusters using a boxplot to asses the capabilities of the resulting models. With both the gray level co-occurrence matrix feature extraction and a combination of first the discrete wavelet transform and next local binary patterns, the framework was able to find a set of sample groups that were related based on their toughness property.**

Index Terms—**Pattern recognition, Feature extraction, Clustering, Metallurgy**

I. Introduction

Metals have already been used by humanity for millennia and are still a very common material. Currently a wide variety of metals is used in numerous applications. Metallurgical materials are used from industries like the aerospace industry, to electrical applications like computers and power distribution. The reasons that metals are this often used is due to the fact that metals are a collection of material with each their own specific properties that suit different applications. The common characteristics for metals are their high tensile strength and durability [2], making them often a good choice instead of other materials. When choosing a kind of metal, the key properties to take into account are: stiffness, strength, hardness and toughness.

Metallurgical development consists of researching the process to make a material with properties that suit the chosen application. The engineering process consists of different sequential steps. A metal composition must be chosen to create a pure metal or an alloy. Using different heated processes, the microstructure and phase of the material are shaped to express the desired properties. Cold processing techniques can then deform the metal with the aim to change the mechanical properties of said metal. There exist numerous processes with each their own parameters that influence the resulting properties, making the process of developing the 'perfect' metal a difficult process. Executing these processes, studying the results and optimizing is financially expensive and time consuming [3]. In this thesis we specifically focus on the toughness property of a material. The toughness property is in summary the fracture resistance of a material. If the material has a high toughness, then it is considered a ductile material and it absorbs a high amount of energy, in the form of plastic deformation, before fracturing. Otherwise it is a brittle material and instead of plastic deformation, the material will just fracture without absorbing much energy. In some applications like nuclear reactors, aerospace applications and gas pipelines, fractures can have catastrophic consequences. For this reason, depending on the application, the toughness property of the used materials is a crucial factor to consider [4].

Using scanning electron microscopic images from fracture surfaces, experts are able to identify the features that contribute to the amount of energy a material absorbs before fracturing. A quantitative analysis of the influence that each feature exerts on the toughness property of a material is not yet possible. Artificial intelligence and in particular feature extraction algorithms can be utilized to aid in the quantitative analysis of the microstructure appearing in the microscopic images.

Artificial intelligence and pattern recognition techniques have already been applied to metallurgical research. AI has been used to optimize the parameters in the production processes [5]. Pattern recognition and AI also have already shown the capabilities transform metallurgical microstructures in a compact low-dimensional representation [6] and to classify metals regarding their chemical composition [6]. However, to the best of our knowledge, there has been no research into grouping microscopic images from fracture surfaces together into groups of related metallurgical properties, in this case the toughness property. In this thesis we present methods to optimize the separation of clusters from metallurgical images and to find groups of similar samples relating to their toughness property. We evaluate the clustering results using the Davies-Bouldin index [1] and the toughness distribution present in the clusters.

The structure of this paper can be described as follows, first, we cover the necessary preliminaries. Next, we discuss the problem formulation and we present our original work. Finally we discuss the experimental results and draw a conclusion.

II. Preliminaries

A. Metallurgical Background

The toughness of a material is defined as the amount of energy a material absorbs before fracturing [2]. The absorption of energy is done in the form of plastic deformation. Fracture toughness is a similar property that is expressed as the amount of energy absorbed before a preexisting the crack propagates [7]. A material with a high toughness is called ductile and will deform before a fracture initiates. Brittle materials have a low toughness and will barely deform before fracturing. A good example of a brittle and ductile material are cold glass and a steel wire. When glass is put under strain it will deform very little and shatter into pieces, it is thus considered a brittle material. A steel wire, however, is able to bend a significant amount before it breaks.

Toughness is heavily influenced by temperature. Increasing the temperature of a material will increase its toughness and vice versa. So when taking toughness into account, it is important to evaluate the materials at the temperatures it will experience while being used.

Toughness can be assessed by using different methods, with a Charpy impact test being one of the more common ones [8]. The first step of the test is to drill a notch in the metal sample to facilitate the breaking process. Then a hammer is dropped on the sample via a pendulum. After the sample is broken, using the maximum height the hammer reaches in the remainder of the swing, the absorbed energy can be calculated and thus the toughness can be determined. Microscopic images of the obtained fracture surfaces can then be used to analyze the properties of the material. A dataset of images from samples that were fractured in the Charpy test is being used in this thesis.

B. Feature extraction algorithms

Using a raw image as a feature representation will results in high-dimensional input vectors. High-dimensional input spaces lead to high computational complexity when analyzing the data and due to the fact that each feature introduces some kind of noise, noisy inputs, which leads to high chances of overfitting. However, the high dimensionality of the data is not truly high dimensional. The important features normally reside in a much lower dimensional space. For this reason, feature extraction techniques are used to transform the data to such a low-dimensional vector. In this thesis, the feature extraction techniques used are: first order statistics (FOS) [9], gray level co-occurrence matrix (GLCM) [10], histograms of oriented gradients (HOG) [11], local binary patterns (LBP) [12], discrete wavelet transform (DWT) [13], Weyl transform [14] and autoencoders [15].

FOS is the most simple feature extractor, it calculates the following statistical measures: the minimal, maximal and average pixel intensities, the variance of the pixel intensities and the skewness and kurtosis of the pixel intensities. The latter two are statistical measures that describe respectively the symmetry and the frequency of outliers.

GLCM represents an image in a matrix using the spatial relations between pixels. Using this matrix we can extract 5 features, namely: correlation, contrast, energy, homogeneity and entropy. These features are sometimes called the Haralick features named after the original author [16].

HOG is a technique that calculates the gradients present in a $m \times n$ cell in the image. The dominant gradient is then pooled from each cell and added to a histogram bin. This histogram can finally be defined as the resulting feature vector. HOG is a feature extractor often used in object detection.

LBP: transforms each pixel into a binary encoding. Pixels are compared to p pixels surrounding the central pixel with radius R. For each of these comparisons if the central pixel has a higher value 1 is appended to the encoding, otherwise zero is added. By adding these encodings into a histogram, a feature vector can be constructed.

Weyl Transform is a method that first concatenates all the pixels in a raster scanning fashion into a vector which is then transformed into a feature vector using the autocorrelations of the stacked patch.

An **Autoencoder** is an unsupervised deep learning technique [17], that learns to compress an image into a low-dimensional latent space, and reconstructing the image from this latent representation. Using this technique the autoencoder learns to transform an image into a low-dimensional feature vector containing the necessary features.

C. Clustering Algorithms

Clustering is an unsupervised machine learning method. This means that the data does not contain any labels that can be used for an artificial intelligence model to learn the correct classification. Clustering algorithms assign the data points based on similarity into groups called clusters [18]. In this thesis we evaluate three clustering algorithms, namely: k-means [18], Elastic Net Sparse Subspace (ENSSC) [19] and hierarchical clustering [20].

K-means clustering is an iterative algorithm that optimizes the resulting clusters by first randomly generating the cluster centers in the feature space. Then, it iteratively assigns the data points to the closest clusters, based on the euclidean distance. Finally, the algorithm regenerates the cluster centers as the average of the data points assigned to that cluster.

Elastic Net Sparse Subspace clustering is a relaxation of the sparse subspace clustering (SSC) algorithm. SSC is based on the principle that each particular data classes resides a in lower-dimensional subspace and the *self-expressiveness* property of the data, meaning that a data point can be represented as a linear combination of other data points. Finding a sparse linear combination for each data point can be formally expressed as

$$\min_{C} \|\mathbf{C}\|_0 \quad \text{s.t. } \mathbf{Y} = \mathbf{Y}\mathbf{C}, \quad \operatorname{diag}(\mathbf{C}) = \mathbf{0},$$

with $\mathbf{Y}$ the dataset and $\mathbf{C}$ the sparse coefficient matrix. A visualization of the sparse optimization problem is presented in Figure 1 Finding the most sparse solution, i.e., solving for the ℓ_0 norm is an NP-hard problem so a possible relaxation

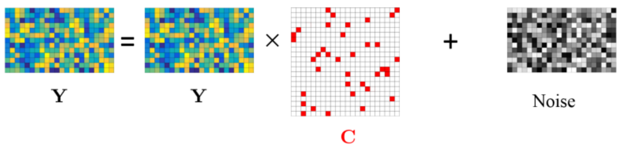


Fig. 1. A visual representation of the Sparse Optimization Problem. [22]

is the ENSSC algorithm where the optimization problem is reduced to

$$\mathbf{c}^*\left(\mathbf{x}_j, \mathbf{X}_{-j}\right) := \arg\min_{\mathbf{c}} f(\mathbf{c}; \mathbf{x}_j, \mathbf{X}_{-j}),$$

where f is defined as

$$f(\mathbf{c}; \mathbf{b}, \mathbf{A}) := \lambda\|\mathbf{c}\|_1 + \frac{1-\lambda}{2}\|\mathbf{c}\|_2^2 + \frac{\gamma}{2}\|\mathbf{b} - \mathbf{A}\mathbf{c}\|_2^2.$$

The optimization problem can be solved efficiently as f is a highly convex function. The next steps in the algorithms are to normalize the columns of the coefficient matrix C and construct the similarity graph as $\mathbf{W} = \frac{1}{2}(|\mathbf{C}| + |\mathbf{C}|^T)$. Then spectral clustering [21] can be applied using $\mathbf{W}$ as the similarity graph. Spectral clustering is an extension of the k-means algorithm that uses a similarity graph instead of the euclidean distances to compare datapoints.

Hierarchical clustering is similar to k-means clustering in the perspective that it uses the euclidean distance as a similarity measure, but the big difference is that, contrary to k-means that is a top-down approach, hierarchical clustering is a bottom-up approach. Hierarchical clustering starts with treating each data point as its own cluster and then constructing a merge tree or dendrogram to merge the most similar data points until a specified number of clusters is reached or a similarity threshold has been exceeded.

III. Problem Formulation

In this thesis, the objective is to develop a framework which can cluster microscopic images of fracture surfaces together into groups with similar metallurgical properties and more specifically the toughness property, using feature extraction techniques and unsupervised machine learning. Secondly the goal is to evaluate which feature extraction technique is capable of finding a low-dimensional representation for the microstructure of a metal sample and which clustering algorithm produces the best results using the resulting feature vectors.

IV. Proposed Methods

In this thesis, two methods were developed. The first method uses a dataset, which does not contain any data on the toughness of the metal samples included. The goal of this method is to find which combination of image magnification, feature extraction technique and clustering algorithm result in the highest quality clusters. For the optimization and evaluation of the clustering algorithms, the Davies-Bouldin index was used [1]. The Davies-Bouldin index is formally defined as

$$db = \frac{1}{n}\sum_i \max_{j\neq i}\left(\frac{\sigma_i + \sigma_j}{d\left(c_i, c_j\right)}\right),$$

where n is the number of clusters, c_i is the cluster center of cluster i, σ_i is the dispersion of a cluster and $d\left(c_i, c_j\right)$ is the distance between two cluster centers. Minimizing the Davies-Bouldin index means that the inter-cluster distances is maximized. This evaluation can both be used as optimization of the parameters and comparison between different combinations of feature extraction and clustering algorithm. The first method is shown in the gray section of Figure 2. The first step serves to crop the image, in order to remove additional textual information added at the bottom of the image, and then divide the image into (64×64) patches. These patches are then transformed into a lower-dimensional feature vector that serves as input for the clustering algorithm. Using the Davies-Bouldin index we can optimize the parameters of the feature extraction methods and the clustering algorithms.

The second proposed method is developed for a data set containing the toughness values of each sample in addition to the images that are included in the data set. This method aims to find classes of materials in the data set that are related in terms of their toughness property. The additional changes we made in comparison to the first method, are visualized outside of the gray area in Figure 2. The images are classified with the most frequent label of its patches. Using these labels we can then evaluate the performance of the model, by assessing the resulting distributions in the clusters. For this assessment, boxplots are used as they give a good summary of the distribution and allow for a good comparison of the separation of toughness distributions between clusters.

V. Experimental Results and Discussion

In this Section, the results obtained by applying the previously described methods are presented. First we optimized the discussed algorithms by applying the clustering on the first data set. Secondly we evaluated the capabilities of the algorithms to cluster the samples together based on their toughness similarity in an unsupervised manner.

A. *Clustering optimization*

The first objective of this part is to analyze which magnification level of the images works best with the feature extractors. For this reason we fixed the clustering algorithm to k-means and optimized the combinations of input magnification and feature extractions. The included image magnification levels where: $100\times$, $500\times$ and $1500\times$. These were chosen as they were all fairly well represented in the data set and were nicely spread in the different magnification levels. The resulting Davies-Bouldin scores are presented in Table I. The table shows that there are four feature extraction algorithms that consequently beat the FOS feature extraction, namely: GLCM, LBP, DWT and Weyl Transform. For these feature extractors, the $500\times$ magnification proves each time to be the best. In the

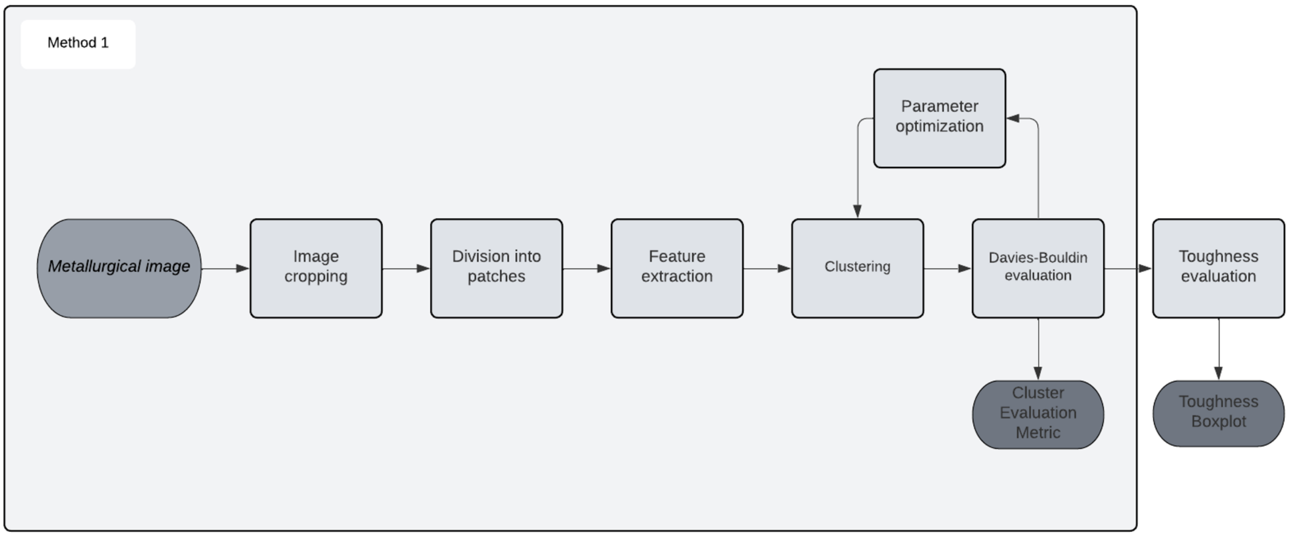


Fig. 2. The presented methods included in this thesis

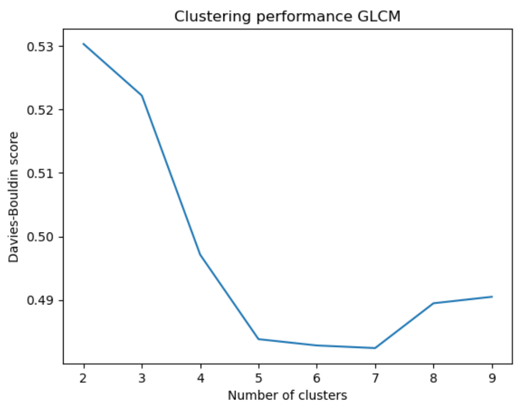


Fig. 3. The Davies-Bouldin score of GLCM with k-means in function of the number of clusters

following we will keep the magnification levels fixed to $500\times$. LBP produces the clusters with the lowest Davies-Bouldin index, but we consider the GLCM feature extraction to be the best here as the evaluation of GLCM is significantly lower for a higher number of numbers, this is illustrated in Figure 3. In this figure it is shown that the GLCM algorithm remains stable for a higher number of clusters while the quality of LBP rapidly decreases. A higher number of clusters is better as it allows for a more fine classification later.

Next, we combined the clustering algorithms earlier described, with the feature extraction techniques. These combinations where then compared using the Davies-Bouldin index, these results are presented in Table II. These results show us that GLCM and DWT can be slightly improved when using hierarchical clustering. Elastic Sparse Subspace clustering was never able to achieve a lower Davies-Bouldin index than the other clustering algorithms. From these results we can conclude that when only assessing the quality of the resulting clusters, LBP with k-means produces the lowest Davies-Bouldin scores, but when the number of clusters is taken into account GLCM with hierarchical clustering is better. It is important to note that a low Davies-Bouldin score does not necessarily mean that the algorithms are able to find meaningful clusters based on the toughness property and that the number of clusters with the lowest score will eventually result in the most meaningful clusters. When working with unsupervised techniques an evaluation of the clustering techniques is however the best that we can do as there are no labels available to evaluate the results.

B. *From clusters to Toughness Classification*

In the second phase of the experiments we aim to find the best feature extractor and clustering algorithm to cluster the samples based on their toughness property. The second data set contains the toughness values of the metal samples and the included samples are more closely related due to the fact that the test were executed multiple times for similar samples at different temperatures. For this reason we see a drop in the Davies-Bouldin scores for most of the feature extraction techniques. Using the most frequently appearing labels of the patches we can assign the full images to the clusters and evaluate the toughness distribution in each cluster. In the following paragraphs we will present the most promising results for the feature extraction techniques and clustering results.

GLCM is just like when using the Davies-Bouldin index a great feature extractor for clustering based on toughness. The clustering results using k-means with 9 clusters are shown in Figure 4. In this figure we see that there are two clusters that are more general and contain a variety of samples, but using the other clusters the data set can be divided into three

TABLE I

THE DAVIES-BOULDIN EVALUATION OF THE FEATURE EXTRACTION TECHNIQUES TOGETHER WITH K-MEANS, FOR EACH IMAGE MAGNIFICATION USING THE FIRST UNLABELED DATA SET.

Feature extraction	**100×**	**500×**	**1500×**
FOS	0.85	0.91	0.91
GLCM	0.50	0.50	0.50
HOG	1.25	2.86	3.03
LBP	0.53	**0.44**	0.82
DWT	0.76	0.75	0.76
Weyl	0.88	0.86	0.77
Auto-encoder	-	2.6	2.5

TABLE II

THE DAVIES-BOULDIN INDEX OF THE DIFFERENT CLUSTERING ALGORITHMS USING THE FIRST DATA SET, WITH AN IMAGE MAGNIFICATION LEVEL OF 500×.

Feature extraction	**K-means**	**Hierarchical**	**Sparse subspace clustering**
FOS	0.91	0.93	1.93
GLCM	0.50	0.48	0.69
HOG	2.86	1.25	1.60
LBP	**0.44**	0.62	1.12
DWT	0.75	0.73	2.0

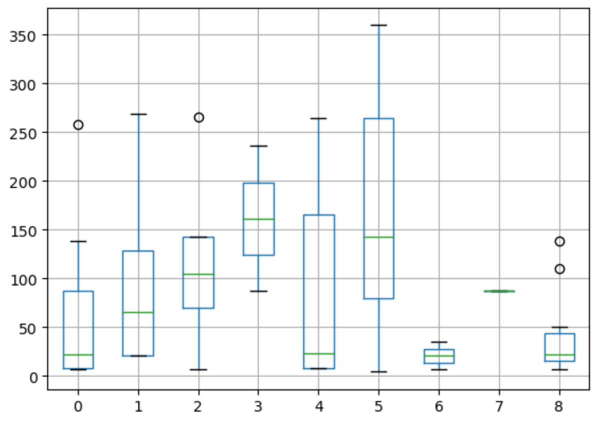


Fig. 4. The results of using GLCM feature extraction with k-means clustering for 9 clusters.

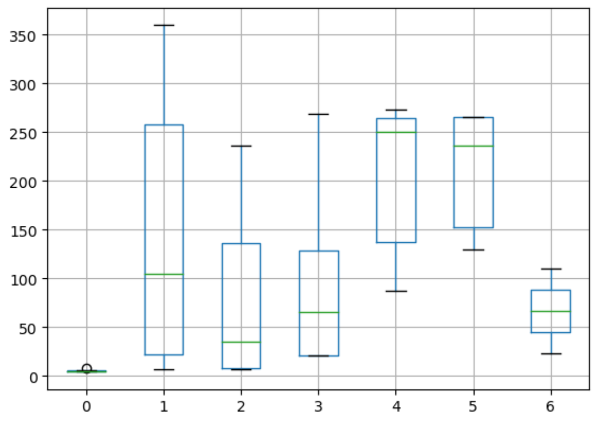


Fig. 5. The resulting clusters produced by hierarchical clustering with 7 clusters on the LBP feature vectors.

to four classes based on toughness. Note that we see here more clusters than final classes. The reason for this is that when fewer clusters are used, the results are less fine and more *general* clusters appear. Increasing the number makes the distributions found in the clusters narrower even though some clusters will represent the same class of materials.

LBP achieved the lowest Davies-Bouldin index in comparison to extraction techniques. It is thus able to divide the data into clusters that have the best quality based on inter-cluster distance and compactness. This should result in a more stable when combining the image patch cluster labels into an image label. It is however no guarantee for clusters that are meaningful with respect to toughness. The Hierarchical algorithm however, was able to separate the data for the most part in three classes fairly well separated classes. These results are presented in Figure 5. The downside was that there was still one cluster containing samples with both high and low toughness values. LBP does suffer from the compression artifacts introduced by the JPEG compression. So reducing the noise introduced by these artifacts could improve the results.

DWT reduces the blocking artifacts introduced by compression. By first extracting the HH representation of the discrete 2D-wavelet transform, and using the resulting image as input for the LBP algorithm, we can reduce the noise in the resulting feature vectors. This is shown in the results presented in Figure 6, where k-means clustering for 9 clusters is used and the algorithm is able to divide all data into three classes based on the toughness properties without the occurrence of a more

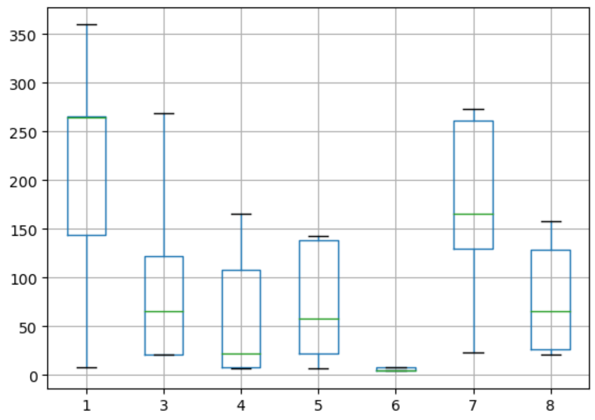


Fig. 6. The results of using first DWT and then LBP feature extraction with k-means clustering for 9 clusters.

general cluster. As the LBP pattern is a binary combination there are no real drawbacks of the first DWT step.

The other previously mentioned feature extraction techniques were not able to find meaningful clusters based on toughness with a similar quality as with LBP. Our baseline FOS resulted a few clusters that all had a similar distribution. HOG and DWT were able to group the data somewhat together based on the toughness, but significantly worse than GLCM or DWT-LBP. The Weyl features showed in the k-means experiment the capability to separate the data quite well into 3 classes, but a few clusters contained very broad distributions making the classification not always meaningful. Finally the autoencoder was not retrained on the new data set, because of the high Davies-Bouldin score.

The other feature extraction techniques were mostly able to find some toughness separation in the data, but most clustering results contained many clusters with both high and low ductility measurements making most of these classifications not always informative.

We can conclude that GLCM and a combination of DWT and LBP results in the clusters when considering the toughness property of the samples. These both show that the data can be separated into three to four classes. In future research it may be beneficial to extend on this work using supervised or semi-supervised techniques as these are able to learn from the given labels.

VI. Conclusion

In this thesis we developed a method to evaluate feature extraction techniques and clustering algorithms based on their ability to assign the dataset into well separated and compact clusters. We showed that LBP and GLCM were the feature extraction techniques producing the best results based on the Davies-Bouldin index. The results of GLCM showed that this technique did not only produced high quality clusters, but that it was capable of separating the data based on the toughness property by using only microscopic images. The second great technique was the combination of first executing DWT and using the results as input for the LBP algorithm. The DWT step serves as a transformation to reduce the noise present in the image and LBP performs the final feature transformation.

References

[1] M. Mughnyanti, S. Efendi, and M. Zarlis, "Analysis of determining centroid clustering x-means algorithm with davies-bouldin index evaluation," in *IOP Conference Series: Materials Science and Engineering*, vol. 725, no. 1. IOP Publishing, 2020, p. 012128.

[2] W. D. Callister, D. G. Rethwisch *et al.*, *Materials science and engineering: an introduction*. Wiley New York, 2018, vol. 9.

[3] M. Odendaal, F. Vermaak, and E. du Toit, "Cost estimation and management over the life cycle of metallurgical research projects," *Southern African Business Review*, vol. 19, 01 2015.

[4] M. E. Launey and R. O. Ritchie, "On the fracture toughness of advanced materials," *Advanced Materials*, vol. 21, no. 20, pp. 2103–2110, 2009.

[5] L. Dobrzański, M. Kowalski, and J. Madejski, "Methodology of the mechanical properties prediction for the metallurgical products from the engineering steels using the artificial intelligence methods," *Journal of Materials Processing Technology*, vol. 164, pp. 1500–1509, 2005.

[6] M. Larmuseau, M. Sluydts, K. Theuwissen, L. Duprez *et al.*, "Race against the machine: can deep learning recognize microstructures as well as the trained human eye?" *Scripta Materialia*, vol. 193, pp. 33–37, 2021.

[7] S. Zhang, D. Sun, Y. Fu, and H. Du, "Toughness measurement of thin films: a critical review," *Surface and Coatings Technology*, vol. 198, no. 1, pp. 74–84, 2005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0257897204010084

[8] R. Moskovic and P. Flewitt, "An overview of the principles of modeling charpy impact energy data using statistical analyses," *Metallurgical and Materials Transactions A*, vol. 28, no. 12, pp. 2609–2623, 1997.

[9] J. Xu, P. Ye, Q. Li, H. Du *et al.*, "Blind image quality assessment based on high order statistics aggregation," *IEEE Transactions on Image Processing*, vol. 25, no. 9, pp. 4444–4457, 2016.

[10] N. Zulpe and V. Pawar, "Glcm textural features for brain tumor classification," *International Journal of Computer Science Issues (IJCSI)*, vol. 9, no. 3, p. 354, 2012.

[11] C. Tomasi, "Histograms of oriented gradients," *Computer Vision Sampler*, pp. 1–6, 2012.

[12] Z. Guo, L. Zhang, and D. Zhang, "A completed modeling of local binary pattern operator for texture classification," *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1657–1663, 2010.

[13] G. Othman and D. Q. Zeebaree, "The applications of discrete wavelet transform in image processing: A review," *Journal of Soft Computing and Data Mining*, vol. 1, no. 2, pp. 31–43, 2020.

[14] T. Zhao, G. Montereale Gavazzi, S. Lazendić, Y. Zhao *et al.*, "Acoustic seafloor classification using the weyl transform of multibeam echosounder backscatter mosaic," *Remote Sensing*, vol. 13, no. 9, p. 1760, 2021.

[15] D. Bank, N. Koenigstein, and R. Giryes, "Autoencoders," 2021.

[16] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, no. 6, pp. 610–621, 1973.

[17] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[18] K. P. Sinaga and M.-S. Yang, "Unsupervised k-means clustering algorithm," *IEEE Access*, vol. 8, pp. 80 716–80 727, 2020.

[19] C. You, C.-G. Li, D. P. Robinson, and R. Vidal, "Oracle based active set algorithm for scalable elastic net subspace clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3928–3937.

[20] F. Nielsen and F. Nielsen, "Hierarchical clustering," *Introduction to HPC with MPI for Data Science*, pp. 195–211, 2016.

[21] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, pp. 395–416, 2007.

[22] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 11, pp. 2765–2781, 2013.

Contents

List of Figures

List of Tables

1 Introduction

Metals have already been used for millennia in human history. Copper, for example, has already been used for more than seven thousand years [1], but the use of metals is not limited to copper. In practice a wide variety of metals and alloys are used for numerous applications, from [2–4]. The reason that metals appear in this many applications, is due to the fact that there is a wide variety of different metals available that each have their own specific properties that suit different applications. The relation between these different kinds of metals, is their high tensile strength and durability. When a choosing a metal, the key properties to take into account are: stiffness, strength, hardness and toughness [5].

The applications of metal range from construction to aerospace industry. It is possible to use metals for these different applications, due to the fact that its metallurgical material can be to engineered to have the properties that exactly suit the application. The engineering process consists of different sequential steps. A chemical material composition must be chosen for the metal or alloy. Then using different heated processes, the microstructure of the final metal can be influenced and adapted to the desired application. Using cold processing techniques, the microstructure can be deformed with the aim to change the mechanical properties. All these steps are time intensive and each process has a wide variety of possible parameters, making it very difficult to find the perfect production process for a metal with certain properties. To reduce the time cost of metallurgical development [6], it is important to have a very good understanding of the influence of all aspects present in the microstructures of metals. With a better understanding, it is possible to make more informed decisions to design the production process, thus requiring less development cycles [7]. In this thesis we will specifically focus on the toughness property of materials. The toughness property of a material can be summarized as the resistance to fractures of that material. In some applications like nuclear reactors, aerospace applications and gas pipelines, fractures can have catastrophic consequences. For this reason, depending on the application, the toughness property of the used materials is a crucial factor to consider [8].

Using microscopic images from fracture surfaces, experts can currently identify the features that indicate the metallurgical properties found in the tests. A quantitative evaluation of the feature influences to predict the metallurgical and more specific the toughness property is not possible yet. In this part, artificial intelligence and feature extraction techniques in particular can be utilized for assessing the microstructures of materials. When pattern recognition algorithms combined with classification models can be used as a tool for pattern analysis in the microscopic images, then a deeper evaluation of the microstructural characteristics is possible and the relation to the metallurgical properties of a given metal sample can be found. For researchers this could be useful to improve the understanding on the quantitative effect the features in an image have on the resulting metallurgical properties.

Artificial intelligence and pattern recognition are, by no means, recent subjects in metallurgical research. Evolutionary methods have already been developed to optimize the process parameters in the production of metals [9]. Feature extraction techniques and neural networks have already been used to find compact representations of these microstructures and classify them by their chemical composition [10, 11]. However, to the best of our knowledge, no models have yet been developed as a tool for metallurgical property analysis. In this thesis we will develop unsupervised clustering models, that can find a natural classification of metal samples based on the toughness properties of these samples.

We would like to thank OCAS, Lode Duprez, and John Vande Voorde for composing the data sets and provide the necessary guidance concerning the metallurgical part of this thesis. Their knowledge and help was essential for the completion in the best manner possible.

1.1 Objective and Research Methodology

This work focuses on discovering patterns in microscopic metallurgical images, from fracture surfaces, that are related to the toughness property of the metal samples and can thus be used to cluster the samples together in clusters of related toughness properties. Due to the fact that available data is unlabeled, i.e., the data does not contain any classes in which the samples should be classified. Thus, in this manuscript we will focus on unsupervised machine learning techniques as the goal is not to classify the samples into an existing set of classes, but to find a natural set of groups, with related toughness, residing in the data.

The high-dimensional data is not truly high-dimensional. The features usually reside in a lower dimensional space, so the intrinsic features can typically be represented in a lower dimensional space. Using feature extraction techniques, we aim to obtain such a lower-dimensional representation. The utilized feature extraction techniques already exist for a very long time and found their applications in many different fields of science and engineering and they represent a well researched topic. We will base ourselves on the existing feature extraction techniques and we will compose a set of feature extractors that are used in similar applications and adapt these for the use on metallurgical images. We will then evaluate which algorithms are able to remove the additional dimensions and reduce some of the noise, but still capture the underlying information that can be used to efficiently cluster the samples, based on their metallurgical properties. In the following we will examine which feature extraction technique can find the most representative lower-dimensional representations of the metallurgical images.

The toughness property of a material is a continuous physical quantity and is not suitable for the purpose of classification. The values can be divided into buckets to create discrete labels, but then the number of classes and optimal extreme values of each bucket must be known. As these are unknown at the moment, a supervised classification algorithm cannot be trained on the data and we must utilize unsupervised methods to find the classes naturally residing in the data, even when the toughness labels of the metal samples are available. We can evaluate the unsupervised classification using the provided labels.

Clustering is such an unsupervised machine learning method that groups similar data points together into groups called clusters. In this work we will research which clustering algorithms can be combined to the feature extraction methods to assign the images of metal samples into groups of a similar toughness property. We will investigate which techniques provide the most meaningful results and which are the most suitable to divide the available data in to groups related to their toughness.

1.2 Main Contributions

The application of artificial intelligence and more specifically clustering algorithms on microscopical metallurgical images of fracture surfaces has, to the best of our knowledge, not been researched before. Research on classifying different kinds of metals has already been published [10, 11], but not yet on finding a relation between the metallurgical properties and the microstructure of metals.

The main contributions of this work can be summarized as follows:

- Feature extraction techniques for pattern recognition have already been used for numerous applications, but not yet for classifying metals based on their metallurgical properties. In this work we investigate the state-of-the-art feature extraction techniques on their ability to find feature vectors that can be used to assign the metallurgical microscopical images of fracture surfaces into well separated clusters. Secondly, we compare the features in their ability to extract informative features relating to the toughness property of the given microstructure.

- In this work unsupervised clustering methods are utilized to find latent data structures present in the resulting feature vectors of a feature extraction technique. Using cluster evaluation metrics, such as the Davies-Bouldin index or silhouette score, we determine for each feature extractor the most suited clustering algorithm to find well separated clusters in the data. The algorithms are also compared in their ability to assign the given feature vectors into clusters of samples that are closely related based on their toughness property.
- We propose a general pipeline for classifying metal samples based on their toughness property. Metal classification algorithms based on microscopic images have already been developed [10, 11], but determining the relations of metallurgical properties between the given samples, was not in the scope of their research. In this work, we develop a framework that can be used for the analysis of the metal samples, based on their toughness property.

Furthermore, we made some original visualization that serve for a better understanding of the presented models and their presented results. We also have written a well organized Python code base that can be used by metallurgical experts for their research in the field of metal properties or as a baseline for future developments on the subject.

1.3 Thesis Outline

This thesis is organised as follows:

- *Chapter 2: Background and Related Work* presents the necessary metallurgical knowledge used in this work. The characteristics of the toughness property and the difference between brittle and ductile materials are explained. The Charpy test for toughness measurement is covered and the most important features, relating to toughness, in fracture images are listed. Furthermore the feature extraction algorithms and their use in pattern recognition are researched. Finally both classical and more advanced clustering algorithms are shown.
- *Chapter 3: Proposed Methods* presents the different methods that were developed in this work. We present a method to optimize the quality of clusters resulting from feature extraction and clustering algorithms. Secondly we present a framework to classify metal samples into groups that are related in terms of their toughness property.
- *Chapter 4: Experimental Results and Discussion* discusses the experimental results by applying the previously mentioned methods to two data sets. We start by analyzing the influence of inputs, feature extraction and the clustering algoritms on the resulting cluster quality metrics. Afterwards, we compare the ability of different feature extractors combined with clustering algorithms to find metallurgical samples related in toughness property.

- *Chapter 5: Conclusion and Future Work* gives a summary of the contributions and results presented in this thesis and lists the possible future research directions for which our work could serve as a baseline.

2 Background and Related Work

In this Chapter, we will provide an overview of the fundamental metallurgical knowledge required for this work, as outlined in Section 2.1. We will delve into the concept of toughness as a critical material property, analyzing its significance. Additionally, we will examine the key characteristics present in fracture images that serve as indicators of toughness. Moreover, Sections 2.2 and 2.3 will focus on the various algorithms employed in this thesis. These algorithms are divided into two main categories. Firstly, we will discuss the feature extraction algorithms, which facilitate the transformation of an image into a low-dimensional feature vector. We will explain their functionality and discuss their relevance in the context of metallurgical images. Secondly, we will explore clustering algorithms, encompassing both traditional and more recent approaches.

2.1 Metallurgical Background

Metals possess various properties that define their behavior in the world, such as density, hardness, yield strength, and toughness. For the purpose of this thesis, our focus will be solely on exploring one specific metallurgical property: toughness. Toughness refers to the ability of a material to absorb energy before fracturing [5]. Fracture toughness is a related property, as it determines the amount of energy required for a preexisting fracture to propagate [12]. Given that metals often encounter high stresses, the toughness property becomes crucial in selecting the appropriate metal for a particular application. For instance, in an oil pipeline, it is vital to ensure that if a failure occurs and a crack appears, the crack does not propagate due to the pressure exerted by the oil. This prevents further leaks and damages. Numerous similar examples highlight the significance of toughness in various scenarios.

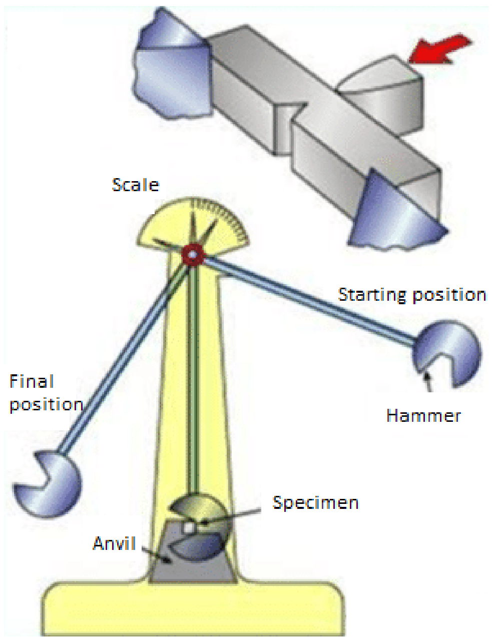


Figure 2.1: The charpy set up for toughness measurement [14].

2.1.1 The Charpy Impact Test

The toughness property of a metal sample, can be measured using a Charpy impact test [13]. In this test a notch is drilled into a metal sample The notch is 2mm deep and as an angle of 45 degrees. Then, a hammer fixed to a pendulum is dropped on the sample in order to break it. After breaking the metal, the toughness can be found, by measuring the remaining mechanical energy in the hammer, which can be calculated by measuring the maximum height the hammer reaches in the second part of the swing, after breaking the sample. By subtracting the second measurement from the initial energy, we can calculate the energy absorbed by the material sample. An illustration of the test setup is given in Figure 2.1. It shows how a sample is placed and how the hammer swings to break the sample. Finally, it can be seen how the measurement of maximum height reached by the remainder of the swing, can be obtained. Due to the influence of temperature on the toughness property, the Charpy test is often repeated multiple times at different temperatures. It should be noted that other tests for toughness measurement exist, but in this work only the Charpy impact test is considered, as the data sets used only contain Charpy test samples.

2.1.2 Classification Based on Toughness

Materials can be divided into two general groups based on their toughness. The first group is the brittle group, these materials absorb very little energy and barely deform while breaking. Glass

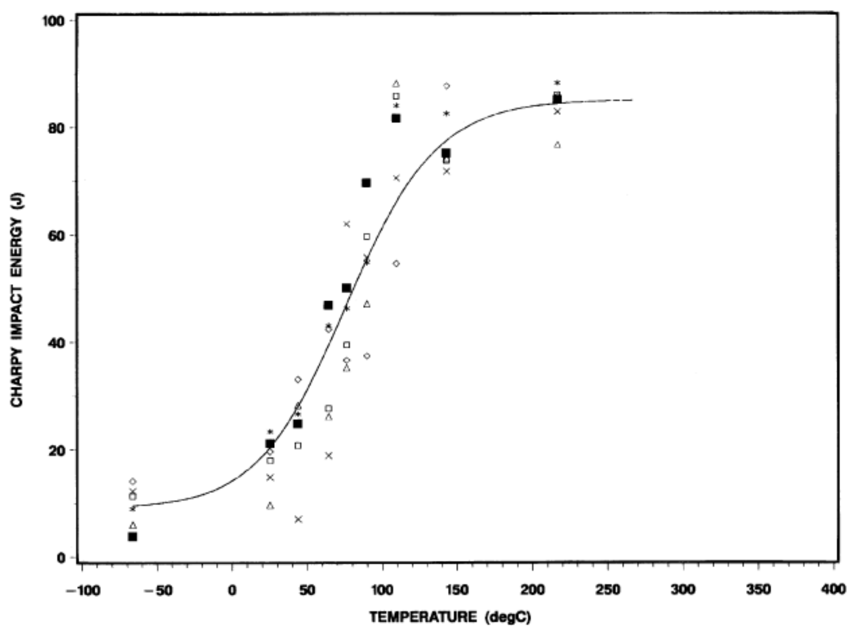


Figure 2.2: The Charpy impact energy depending on the temperature. [13]

is an ideal example for a brittle material, at cold temperatures it does not deform, meaning permanently, but rather breaks instead. The other category of materials is the ductile group, these absorb a lot of energy by deforming before actually fracturing. The deformation mentioned is plastic deformation, meaning permanent. When the stress is removed the deformation stays. A good example for a ductile material is a metal wire, which you can deform into most shapes without a fracture appearing.

As mentioned in the previous paragraph, the amount of deformation and thus absorbed energy is dependent on the temperature. Hot glass will absorb energy instead of breaking, but cold glass shatters instead of deforming. For this reason a Charpy impact test must be performed at different temperatures to gather all information about the toughness property of the material. A graph can be constructed when multiple Charpy tests have been performed, such a graph is shown in Figure 2.2. The shape of this curve is one that is very similar for most materials. Often it is crucial to investigate how the toughness changes for different temperatures as the material may require to be able to withstand different temperatures. A plane, for example, must be able to withstand the same stresses close to the ground while landing or taking of as when encountering turbulence in a freezing cloud. To summarize, knowledge of the temperature curve is crucial for the suitable applications of the material.

After a Charpy impact test a broken metal sample remains. These fractures scraps of metal can provide crucial insights into the microstructure of the material. We can examine the fracture surface of the sample with a scanning electron microscope (SEM) and identify features that contribute to the measured toughness value. The following patterns in an image give hints about the toughness value:

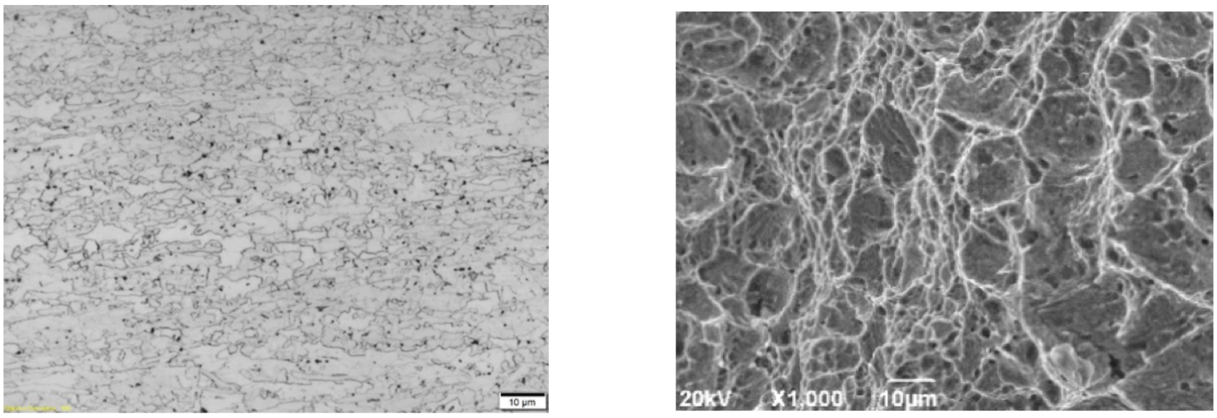
- The **microstructure** of a material gives important insights into the properties of a material. For metals, characteristics such as grain size, and alignment that have a big impact on the behavior of the material. Often the effects of the different processing steps can be analyzed, making the evaluation of the process possible.
- **Pre-existing cracks** have a major influence on the toughness of a material. Some materials have due to their production more pre-existing cracks present in their microstructure, reducing the toughness of that material. The cracks act as a weaker point than the rest of a metal where the fracture can initiate.
- An **inclusion** is the incapsulation of a different material or metal phase in the sample. Inclusions often do not have the same strong bonds at their grain borders as grains of the same materials have. Inclusions can thus give an indication that the material will be weaker and show a more brittle behavior.
- **Dimple** structures are often found in images of fracture surfaces. They are caused by local shear planes that formed a cluster of grains, that were unable to sustain plastic deformation, due to their misorientation. Dimples are considered to be a sign of a ductile fracture [15].

The feature described above can be used to analyze the toughness property of a given metal sample. They can also be used to find similarities in toughness between different samples and group them together to form clusters of similar materials.

2.1.3 Microscopic fracture surface images

The fracture surfaces of the metallurgical samples are captured using Scanning Electron microscopy (SEM). in contrast to traditional microscopes. SEM is based on the electron emission instead of light, the electrons are accelerated to the sample with an acceleration voltage. The reason to use SEM is that very high magnifications are possible using these microscopes. In contrast to optical or light microscopy, where the wavelength characteristics limit the maximum magnification [16]. The microscope is used to capture images of fracture surfaces. The reason to use fracture surfaces is due to the fact that fracture surfaces show characteristics of the behavior of the material while fracturing. The difference between normal images of the microstructure of a metal and a fracture surface is shown in Figure 2.3. The image of the fracture surface is more complicated, but contains more information about the metallurgical properties of the material.

The datasets of images used in this work, contain compressed JPEG images [17]. The JPEG



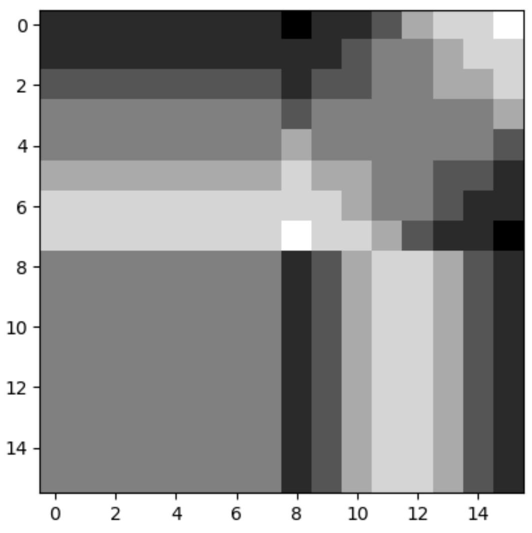
(a) A SEM image of a metal microstructure. (b) A SEM image of a fracture surface.

Figure 2.3: A comparison between a SEM image of a metal microstructure and a fracture surface.

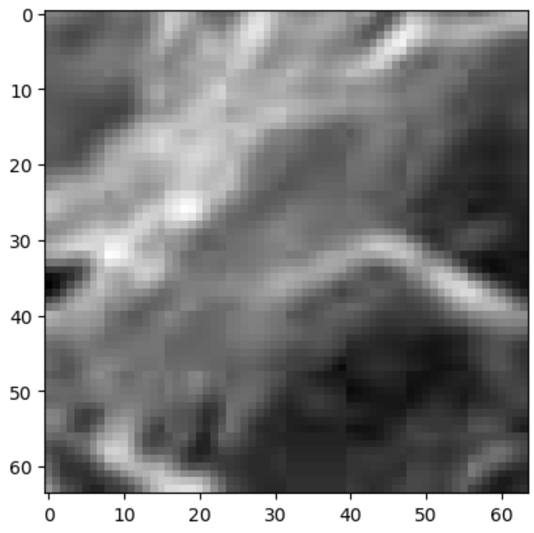
compression algorithm allows to reduce the size of images using quantization and by exploiting spatial correlations between pixels intensities with pixel predictions. The downside of the algorithm is that due to quantization, it is not lossless and the image quality will be reduced when the image is reconstructed. This information loss expresses itself in certain visual errors called compression artifacts. The two compression artifacts mentioned in this work are banding and blocking artifacts. The first is visible when neighboring pixels, with small pixel intensity differences, are reconstructed as the same pixel after compression, thus creating a band of the same pixels intensity.

The latter is due to the predictive algorithm in JPEG. JPEG divides an image into 8×8 blocks and applies quantization on each block separately. Inside the block a predictive algorithm and compression is then used to represent the block in a more compact way. The combination of the quantization and predictive algorithm makes that sometimes the block borders are visible. The two artifacts are shown in Figure 2.4. The first image shows the banding is shown as multiple pixels have the same values and the second presents the visible block borders in the image.

These artifacts are mentioned due to the fact that they have an impact on the feature extraction techniques mentioned below by introducing new features that have no relation to the actual material characteristics. Some of the following methods may drop in performance because of the presence of these artifacts.



(a) A banding artifact.



(b) A blocking artifact.

Figure 2.4: Two compression artifacts.

2.2 Feature Extraction Techniques

As previously mentioned, the necessary features often reside in a lower-dimensional space than the full image. Using the raw image directly as input for artificial intelligence algorithms will increase the computational complexity. Secondly, an increased dimensionality is correlated with a higher amount of noise present and an increased chance of overfitting. To alleviate these problems, we use feature extraction techniques to transform the input data into low-dimensional feature vectors [18].

2.2.1 First Order Statistics

The simplest feature extraction technique consists of taking the first order statistical measures of a grayscale image. These statistical measures are:

- The average pixel intensity.
- The variance of the pixel values.
- The maximum pixel intensity.
- The minimum pixel intensity.
- The skewness of the gray value distribution. Skewness is a statistical measure that indicates the symmetry of the distribution. A high skewness value means that the distribution has a long tail to the right. While a very small skewness indicates a long tail on the left. A skewness value close to zero indicates a symmetrical distribution.

- The kurtosis of the gray value distribution. Kurtosis is a statistical measure that indicates how often outliers occur. A low kurtosis means that the variance of the distribution is caused by a few extreme outliers, while a higher kurtosis means that the variance is caused by more less extreme values.

While this technique is attractive because it is very simple to implement and has a low computational complexity, it is often unable to capture more complex patterns in the image. Another disadvantage is that lighting circumstances of the image has a big impact on the some of the measures that are used in this technique.

2.2.2 Histograms of Oriented Gradients

Histograms of oriented gradients (HOG) [19] is a widely used feature extraction technique for images. The technique is most known for human detection in images. In the algorithm three steps have to be executed. The first step is an optional normalization step. Normalizing the input values reduces the influence of variances in illumination. Then, the second step consists of computing the first order image gradients. Here the gradients of contours and textures are computed in cells of size $m \times n$. The computation of the gradients in a cell is defined as

$$G_x(r,c) = I(r,c+1) - I(r,c-1), \qquad G_y(r,c) = I(r-1,c) - I(r+1,c),$$

where r and c are respectively the row and column indices and G_x and G_y are the horizontal and vertical component of the gradient, respectively. These two components can now be used to calculate the magnitude and orientation of the gradient using

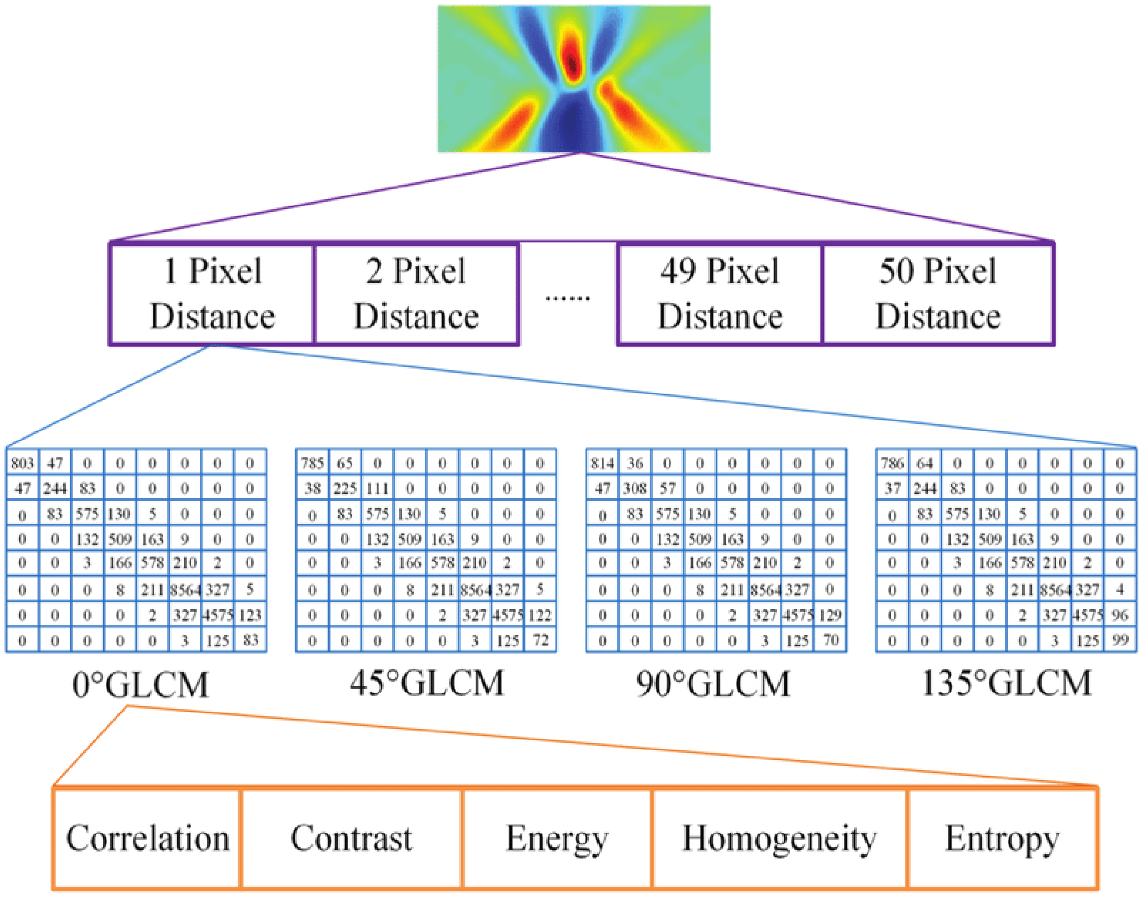
$$\mu = \sqrt{I_x^2 + I_y^2} \qquad \text{and} \qquad \theta = \frac{180}{\pi}\left(\tan_2^{-1}(I_y, I_x) \mod \pi\right),$$

where μ and θ are the magnitude and orientation of the gradient and $\tan_2^{-1}$ is the four-quadrant inverse tangent yielding values between $-\pi$ and π. The third step pools the gradients from image patches. The dominant gradient, i.e. the gradient with the highest μ, of each cell is then matched to a gradient bin. These bins can then be used as a 1-dimensional feature vector. In this work the implementation of the python library Scikit-Image is used [20].

2.2.3 Grey Level Co-Occurrence Matrix

Texture characterization can be done using the Grey-Level Co-occurence Matrix (GLCM). GLCM is a second-order histogram based approach. This means that it is based on correlations among pixels [24]. The GLCM of an $(M \times N)$ image, is constructed as

$$M(i,j) = \sum_{x=1}^{M}\sum_{y=1}^{N}\begin{cases}1, \ \texttt{if} \ I(x,y) = i \ \texttt{and} \ I(x+\Delta x, y+\Delta y) = j, \\ 0, \ \text{Otherwise,}\end{cases}$$



0°GLCM

803	47	0	0	0	0	0	0
47	244	83	0	0	0	0	0
0	83	575	130	5	0	0	0
0	0	132	509	163	9	0	0
0	0	3	166	578	210	2	0
0	0	0	8	211	8564	327	5
0	0	0	0	2	327	4575	123
0	0	0	0	0	3	125	83

45°GLCM

785	65	0	0	0	0	0	0
38	225	111	0	0	0	0	0
0	83	575	130	5	0	0	0
0	0	132	509	163	9	0	0
0	0	3	166	578	210	2	0
0	0	0	8	211	8564	327	5
0	0	0	0	2	327	4575	122
0	0	0	0	0	3	125	72

90°GLCM

814	36	0	0	0	0	0	0
47	308	57	0	0	0	0	0
0	83	575	130	5	0	0	0
0	0	132	509	163	9	0	0
0	0	3	166	578	210	2	0
0	0	0	8	211	8564	327	0
0	0	0	0	2	327	4575	129
0	0	0	0	0	3	125	70

135°GLCM

786	64	0	0	0	0	0	0
37	244	83	0	0	0	0	0
0	83	575	130	5	0	0	0
0	0	132	509	163	9	0	0
0	0	3	166	578	210	2	0
0	0	0	8	211	8564	327	4
0	0	0	0	2	327	4575	96
0	0	0	0	0	3	125	99

Figure 2.5: The GLCM feature extraction process, where first a GLCM matrix is calculated and then 5 feature are extracted from the matrix. [26]

where $(\Delta x, \Delta y)$ is a specified displacement vector and $I(x, y)$ is the pixel value of the image at row x and column y. The construction of such histogram in four directions with a displacement of 1 pixel is shown in Figure 2.5. From the resulting GLCM matrix, 5 features are extracted: correlation, contrast, energy, homogeneity and entropy. These features are sometimes called the Haralick features, named after the original author [25].

2.2.4 Local Binary Pattern

Local Binary Pattern (LBP) is a very popular method for texture analysis [27]. Each pixel gets binary encoded by a comparison with the neighboring pixels on a circle. For each comparison, if the central pixel is greater of equal than the compared pixel, a one is encoded, else a zero gets encoded. By combining these binary figures for a specified number of pixel comparisons, the

LBP encoding is achieved. The LBP encoding is formally defined as

$$\mathbf{LBP}_{P,R} = \sum_{p=0}^{P} s\left(g_p - p_c\right) 2^p, \tag{2.1}$$

where P is the number of pixels that are compared in the LBP encoding, R is the radius on which the comparison pixels are located, g_c is the central pixel and g_p is the comparison pixel. $s(x)$ is the comparison function and is defined as

$$s(x) = \begin{cases} 1, & x \geq 0, \\ 0, & x < 0. \end{cases} \tag{2.2}$$

A histogram can be constructed using pixel encodings to represent the entire image. By constructing an LBP histogram, patterns within the image can be identified and the frequency of occurrence of these patterns can be quantified. The LBP feature extraction technique is known for its simplicity of implementation and computational efficiency. In this thesis the implementation of Scikit-Image [20] is used for LBP feature extraction.

2.2.5 Discrete Wavelet Coefficients

The discrete wavelet transform (DWT) is a technique used in signal analysis [21], but it can also be expanded to images. The rows and columns of pixels are treated as 1D signals. The wavelet transform for image analysis consists thus of a horizontal and a vertical transformation. Each time the transform in a certain direction is executed, the dimension in that direction of the image halves. The iterative transformation process is visualized in Figure 2.6. There are different versions of the wavelet transform, but for edge detection, often the Haar wavelets are used, which can be calculated by multiplying the input to the Haar transformation matrix. In this work we will also use this transformation as it can be efficiently calculated using matrix multiplications and it is often used in feature extraction for classification tasks. The Discrete Wavelet transform, with both the vertical and horizontal transformation, consists of the following steps, with an input $N \times N$ input image:

- Divide the image into neighborhoods, which are non-overlapping blocks of size 2×2.
- Apply the Haar wavelet transform on each block individually.
- From these coefficients, four approximations of size $\frac{N}{2} \times \frac{N}{2}$ can be calculated:
  - LL which is an approximation of the original image and is calculated by taking the average of each pixel.
  - HL is the vertical detail and is calculated by subtracting the average of the 2 pixels from the average of the highest pixels in a block.
  - LH is the horizontal detail, calculated by subtracting the average of the left pixels from the average of the right pixels.

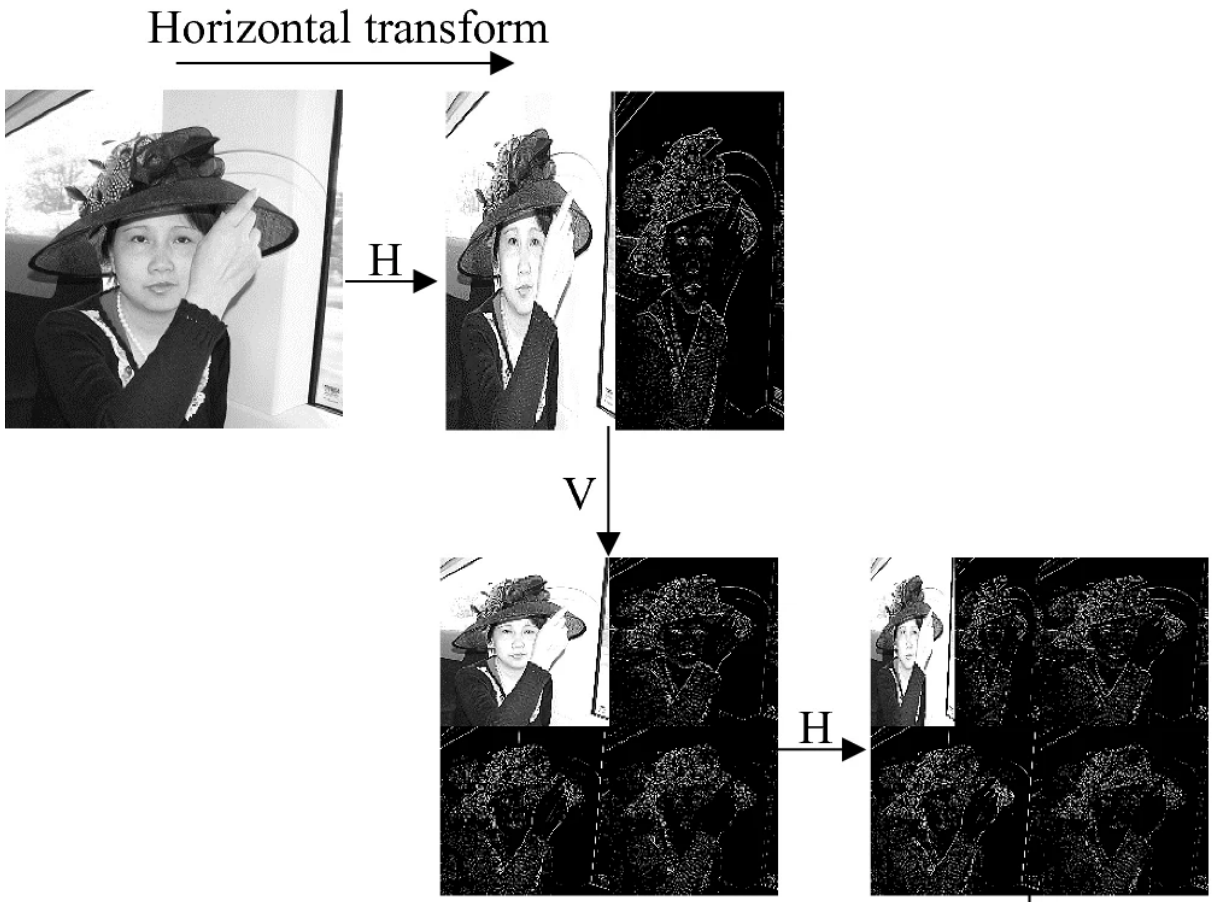


Figure 2.6: The Process of the DWT transformation on an image, with the horizontal and vertical transform alternately.[23]

- HH is the diagonal detail and is calculated by a subtraction of the averages of top-left and bottom-right.

From the DWT images the features must still be extracted as these are still an matrices. In this work we extract the average and standard deviation of the pixel intensities from each quadrant achieved after doing one horizontal and vertical transformation. The used implementation is from a the library Pywavelets [22].

2.2.6 Weyl Transform

Weyl transform is a statistical feature extraction method used in pattern recognition as a low dimensional data representation. The transform uses a mapping from signal to its autocorrelation coefficients based on the Heisenberg-Weyl group [28].

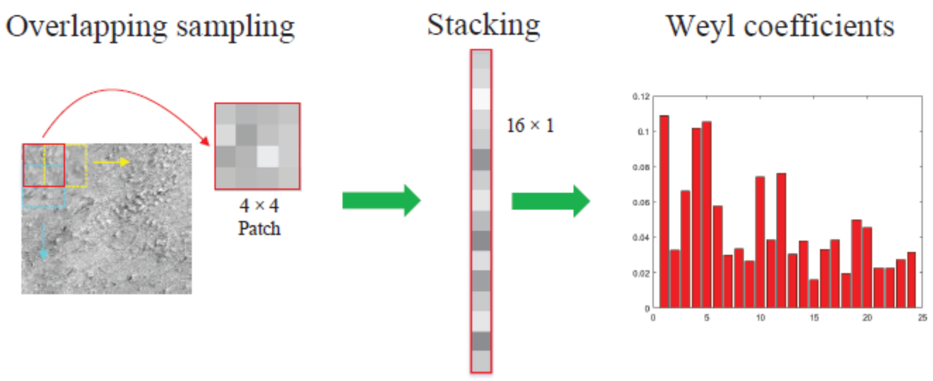


Figure 2.7: The process of extracting the Weyl coefficients [29].

Let $a = (a_0, ..., a_{m-1}) \in \mathbb{Z}_2^m$ and $b = (b_0, ..., b_{m-1}) \in \mathbb{Z}$, and let

$$\mathbf{x} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \mathbf{z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

For

$$\mathbf{D}(a,0) = \mathbf{x}^{a_{m-1}} \otimes \cdots \otimes \mathbf{x}^{a_0}$$

and

$$\mathbf{D}(0,b) = \mathbf{z}^{b_{m-1}} \otimes \cdots \otimes \mathbf{z}^{b_0},$$

where $\otimes$ is the Kronecker product, we can now define the signed permutation matrices from the binary Heisenberg-Weyl group as

$$\mathbf{D}(a,b) = \mathbf{D}(a,0)\mathbf{D}(0,b).$$

Given real symmetric matrices $\mathbf{R}, \mathbf{S} \in \mathbf{V}_{Sym}$, we can associate with the vector space $\mathbf{V}_{Sym}$ the inner product

$$\langle \mathbf{R}, \mathbf{S} \rangle := \mathrm{Tr}(\mathbf{R}^T \mathbf{S}),$$

which subsequently induces the Frobenius norm

$$\|\mathbf{S}\| = \mathrm{Tr}(\mathbf{S}^T \mathbf{S})^{1/2}.$$

The matrices $\mathbf{D}(a,b)$ with $ab^T = 0$ form an orthonormal basis of the vector space $\mathbf{V}_{Sym}$. Thus, each symmetric matrix $\mathbf{R} \in \mathbb{R}^{2^m \times 2^m}$ can be expanded as

$$\mathbf{R} = \sum_{\substack{a,b \in \mathbb{Z}_2^m \\ ab^T = 0}} \left\{ \frac{1}{2^{m/2}} \mathrm{Tr}\left[\mathbf{R} \cdot \mathbf{D}(a,b)\right] \right\} \frac{1}{2^{m/2}} \mathbf{D}(a,b)$$

We can thus expand the correlation matrix yy^T of a vectorized signal $y \in \mathbb{R}^{2^m}$ as

$$\begin{aligned} yy^T &= \sum_{\substack{a,b \in \mathbb{Z}_2^m \\ ab^T=0}} \left\{ \frac{1}{2^{m/2}} \mathrm{Tr}\left[yy^T \cdot \mathbf{D}(a,b) \right] \right\} \frac{1}{2^{m/2}} \mathbf{D}(a,b) \\ &= \sum_{\substack{a,b \in \mathbb{Z}_2^m \\ ab^T=0}} \omega_{a,b}(y) \frac{1}{2^{m/2}} \mathbf{D}(a,b) \end{aligned}$$

The coefficients $\omega_{a,b}(y)$ are known as *Weyl coefficients* of y. The mapping

$$yy^T \mapsto \omega_{a,b}(y)$$

is known as *the Weyl transform*. In practice this means that an image patch of shape (4,4) gets vectorized to a vector of length 16. Here the autocorrelation mapping can be executed to achieve the Weyl representation of that patch. The process is shown in Figure 2.7. This figure shows the steps required to achieve the Weyl representation of an image. First overlapping 4×4 patches are sampled from the image. These patches are then stacked into a 16×1 vector and on that vector, the Weyl transform is performed.

The Weyl transform has very good results in pattern recognition for cloth fabrics in [28] and also in seafloor classifications [29].

2.2.7 Autoencoders

To delve into the topic of autoencoders, it is essential to first explore the principles and foundations of deep learning. Deep learning is a very powerful technique in machine learning often outperforming most other techniques if enough data is available. The result is a neural network that is optimized to give the desired output given an input [30]. The most simple neural networks are often called the multilayer perceptron, which consists of perceptrons, often called neurons, organized in a layered formation. An example of a multilayer perceptron is shown in Figure 2.8. These neurons are universal approximators, meaning that given an a vector of inputs an a bias, the perceptron will learn a function. Perceptrons can be connected to each other to repeat this process a number of times until the final output is reached. It is said that any function can be approximated by a multilayer perceptron that has two hidden layers, where a hidden layer is not the input layer and not the output, but hidden from the outside world. Deep learning uses as the name says deeper neural networks, because a deeper model will need less neurons and learn the desired function faster. The loss function evaluates the quality of the outputs from the neural network and using this loss function and an algorithm called backpropagation [31], the model is optimized to generate outputs with the lowest possible errors.

When using these techniques on images every pixel is fixed to an input in the neural network. Meaning that the neural network has to learn every combination of pixels to learn what is inside

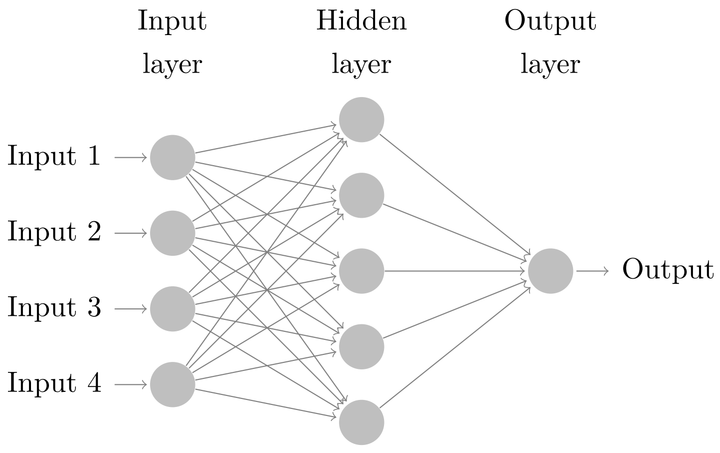


Figure 2.8: An example of the multilayer perceptron architecture with one hidden layer.

the image. This is not very efficient, because if it tries to recognize a car for example, when the car moves a little bit to the left, the inputs have not changed a lot, but it seems like a completely different input to the model. Convolutional layers contain filter matrices. The convolutional multiplies small patches of the image with filters matrices. This way very localized features can be detected by the neural network. When multiple convolutional layers are used serially, the deeper the layer the more complex feature it will be able to recognize. [32]

An autoencoder is an unsupervised deep learning technique that first encodes the input data into a lower dimensional latent space and then tries to reconstruct the original input [33]. The model architecture consists of two components: the encoder and decoder. The autoencoder technique can be applied to images, where the decoder is optimized to find a low-dimensional latent representation for the image and the decoder then learns to reconstruct the images using the latent spaces. For image applications, often a combination of convolutional and fully connected layers is used for optimal results.

The model is trained by combining encoder and decoder into one neural network and using a reconstruction error as a loss function. A loss function shows the error the neural network is still making. For autoencoders the mean squared error (MSE) is often used as a loss function. Using the loss function and backpropagation, the neural network then learns to minimize the differences between the original image and the reconstructed image. Training results can be evaluated using unseen data, by calculating MSE and visually checking the reconstruction of the images. When the algorithm is fully trained the encoder has learned an effective way to compress an image with a minimal loss of information in the compressed representation. Then the decoder learns how to reconstruct the image. We can then use both of these neural networks individually. The encoder can be used to compress the images while the encoder can transform low-dimensional vectors into an image.

In this work an autoencoder will be used for feature extraction. During training the autoencoder learns to represent the images into a low-dimensional space. Due to the fact that the decoder is able to reconstruct the image using the latent space, the latent space contains all crucial information on the microstructure. The transformation from image to latent space is thus a feature extraction method, where the transformation is learned from the available training data.

2.3 Clustering

Most of the machine learning approaches can be categorized as supervised learning, unsupervised learning or a combination of the two. Other machine learning paradigms, like reinforcement learning, are outside the scope of this work. The most commonly known is supervised learning, where a model is optimized to approximate a given function. In many cases labeled samples are not available and then we turn to unsupervised learning, the model is used to analyze datasets without requiring labels as in supervised learning. The goal of unsupervised learning models is to hidden patterns in the data. The last part is called semi-supervised learning and is a combination of supervised and unsupervised learning. A small part of the data contains labels and the majority is unlabeled, reducing the cost of labeling the complete data set. [34]

Clustering is a part of unsupervised machine learning. The algorithms group data points together into a cluster, based on a certain similarity function. The unsupervised nature of the algorithms mean that there are no labels available to evaluate the clustering performance. The algorithms can only be optimized using metrics taking into account characteristics like shape and distance between clusters. Many clustering algorithms exist, but in this work only k-means, spectral clustering, sparse subspace clustering and hierarchical clustering algorithms are covered.

2.3.1 K-Means Clustering

K-means clustering is the most used yet, most simple clustering technique. The algorithm searches in incremental manner for the best cluster solution, minimizing the euclidean distances inside the clusters. The basic steps, given the amount of clusters n, for the algorithm are the following:

- Randomly initialize n cluster centers in the N-dimensional euclidean space, with N the dimension of the feature vectors.
- Iteratively execute the following steps for a previously defined amount of steps:
  - Assign every data point to the nearest cluster center using the euclidean distance of the feature vectors.

- Compute the new cluster centers by taking the average of all the points assigned to a cluster.

The downside of this algorithm is that it falls into local minima. To alleviate this problem the base k-means clustering algorithm is often executed multiple times. Because of the random initialization of the cluster centers the algorithm may optimize to a new local minima or in the best case scenario a global minimum. From the different initializations the best clustering is chosen minimizing the inertia or the sum of squared distances of all samples to their closest cluster centers. In this work we will use the Sci-Kit learn implementation for k-means clustering [35].

When using k-means there are some factors we need to consider. Firstly, because the algorithm is based on euclidean distances of feature vectors it is important that every feature in the vector is scaled to the same scale, otherwise a few features with a high differences may dominate the clustering decisions while others have little influence. Secondly because the distance to one point, the cluster center is minimized, the cluster shapes will always be round. If the relation in the euclidean space has another shape, this algorithm will not always be able to capture that relation.

2.3.2 Spectral Clustering

The spectral clustering algorithm optimizes the clusters based on a similarity graph $G = (V, E)$ [36]. The vertices represent the data points. In the graph, two data points, y_i and y_j, are connected if their similarity s_{ij} exceeds a threshold. A similarity graph is an undirected weighted graph, this means that all edges are weighted by s_{ij}. After a similarity graph is constructed the clustering problem is now reduced to finding the partition of $\mathbf{G}$ that minimizes the weight of the edges going between the different clusters. In this algorithm a graph has two matrices linked to it. The first matrix is the weight matrix $\mathbf{W}$ with W_{ij} is the weight between x_i and x_j. The second matrix is the diagonal degree matrix $\mathbf{D}$. The degree of an undirected weighted matrix is defined as $d_i = \sum_{j=1}^{n} w_{ij}$ or the sum of all weights of connected edges. $\mathbf{D}$ is thus given by $D_{ii} = \sum_j W_{ij}$. In the following we list the individual steps of the spectral clustering algorithm for a dataset with N datapoints and n clusters [37].

- If no similarity matrix is given calculate the weight matrix $\mathbf{W} = [W_{ij}]_{i,j=1}^{N}$, where

$$W_{ij} = \begin{cases} \exp\left(\frac{-||y_i - y_j||^2}{2\sigma^2}\right), & i \neq j, \\ 0, & i = j. \end{cases}$$

- Now the normalized Laplacian $\mathbf{L}$ can be calculated using $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$, with $\mathbf{W}$ the weight matrix and $\mathbf{D}$ the degree matrix of the similarity graph.

- Take the n smallest eigenvalues from $\mathbf{L}$ and determine the corresponding eigenvectors. By stacking these eigenvectors as column vectors we can construct the matrix $\mathbf{B} \in \mathbb{R}^{N \times n}$.
- Next we can construct $\mathbf{E}$ by normalizing each row of $\mathbf{B}$.
- Finally, using a clustering algorithm, we can cluster the rows of $\mathbf{E}$ as datapoints in $\mathbb{R}^n$. For example k-means could be used.

Spectral clustering often outperforms k-means clustering, as it is able to capture more complex relations in the data. The downside is that the calculation of the eigenvectors is computationally expensive for large data sets.

2.3.3 Sparse Subspace Clustering

High-dimensional data belonging to a specific class or category can be effectively represented using a lower-dimensional subspace within the larger high-dimensional space. As a result, the data associated with these classes can be found within the combined set of these lower-dimensional subspaces. Subspace clustering algorithms tackle the clustering task by partitioning the data based on the underlying subspaces they inhabit [38].

Sparse subspace clustering (SCC), uses the *self-expressiveness* of the data. This means that every data point in a union of subspaces can be represented as a linear or affine combination of other points. A data point can generally be represented using different combinations, but ideally a sparse representation uses some points from its own subspace. Finding these representations using a sparse optimization program, will lead to a few other points chosen from the same subspace, but these do not necessarily have to be close [38].

The sparse subspace clustering algorithm is defined as follows: Let $\{S_l\}_{l=1}^n$ be an arrangement of n linear subspaces of $\mathbb{R}^D$ of dimension $\{d_l\}_{l=1}^n$. Consider a given collection of N data points that lie in the union of the n subspaces. All the data points can be represented as a matrix $\mathbf{Y} = [\,\mathbf{y}_1\ \mathbf{y}_2\ ...\ \mathbf{y}_N]$. Following the previously described *self-expressiveness* property , we can assume that each vector $\mathbf{y}_i \in \mathbb{R}^D$, for $i = 1, ..., N$ can be represented as

$$\mathbf{y}_i = \mathbf{Y}\mathbf{c}_i, \quad c_{ii} = 0 \tag{2.3}$$

With $\mathbf{C}_i = [\mathbf{c}_{i1}\ \mathbf{c}_{i2}\ ...\ \mathbf{c}_{iN}]^T$. The constraint $c_{ii} = 0$ ensures that the data point itself is not used in alternative representation of that data point, thus eliminating the trivial solution for the linear combination. A visual representation of the *self-expressiveness* property is shown in Figure 2.9. The most sparse solution, i.e. the solution with the least non-zero elements can be found by minimizing the ℓ_0 norm of the solution vector $\mathbf{C}$. If we rewrite Equation 2.3 for all data points, we get:

$$\min_C \|\mathbf{C}\|_0 \text{ s.t. } \mathbf{Y} = \mathbf{Y}\mathbf{C}, \quad \text{diag}(\mathbf{C}) = 0 \tag{2.4}$$

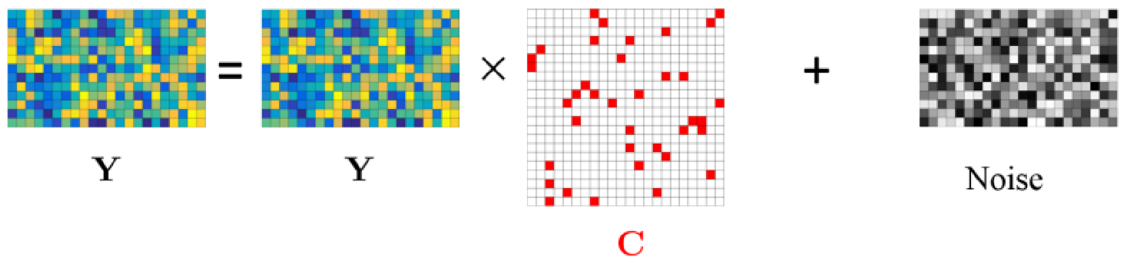


Figure 2.9: A visual representation of the *self-expressiveness* property in sparse subspace clustering [40]

.

Minimizing the ℓ_0-norm is the NP-hard problem of finding the most sparse representation. To improve efficiency, the tightest convex relaxation of the ℓ_0-norm is used, namely the ℓ_1-norm. If we rewrite the optimization problem for all data points:

$$\min_{C} \|\mathbf{C}\|_1 \quad \text{s.t. } \mathbf{Y} = \mathbf{YC}, \quad \text{diag}(\mathbf{C}) = 0 \tag{2.5}$$

The sparse subspace clustering algorithm can be summarized as follows:

- Solve the sparse optimization problem.
- Normalize the columns of the coefficient matrix $\mathbf{C}$.
- Construct the similarity graph as: $\mathbf{W} = \frac{1}{2}(|\mathbf{C}| + |\mathbf{C}|^T)$.
- Apply spectral clustering using $\mathbf{W}$ as the similarity graph.

2.3.3.1 Scalable Elastic Net Sparse Subspace Clustering

Instead of finding, for each data point, a linear combination of as few other data points as possible. The optimal solution can be found by solving Equation 2.4. Because of efficiency reasons the optimization problem is often relaxed to Equation 2.5. Using the ℓ_1 regularization, the solution guarantees to have only connections between data points from the same subspace, i.e. the solution is *subspace-preserving*, but may not be connected. The ℓ_2 regularization optimization solves this problem, but does not guarantee subspace-preservation. Using a mixture between ℓ_1 and ℓ_2 regularization, also called elasticnet regularization a balance between the two properties can be found [39]. The elastic optimization function is defined as

$$f(\mathbf{c}; \mathbf{b}, \mathbf{A}) := \lambda\|\mathbf{c}\|_1 + \frac{1-\lambda}{2}\|\mathbf{c}\|_2^2 + \frac{\gamma}{2}\|\mathbf{b} - \mathbf{Ac}\|_2^2, \tag{2.6}$$

where $\mathbf{b} \in \mathbb{R}^D$, $\mathbf{A} = [\mathbf{a}_1, ..., \mathbf{a}_N] \in \mathbb{R}^{D\times N}$, $\gamma > 0$ and $\lambda \in [0, 1)$. We assume that $\mathbf{b} and \mathbf{a}$ are normalized using the ℓ_2-norm. The elastic model then computes

$$\mathbf{c}^* (\mathbf{b}, \mathbf{A}) := \arg\min_{\mathbf{c}} f(\mathbf{c}; \mathbf{b}, \mathbf{A}). \tag{2.7}$$

Due to the fact that $f(\mathbf{c}; \mathbf{b}, A)$ is a stronly convex function, $\mathbf{c}^* (\mathbf{b}, \mathbf{A})$ is a unique solution. This optimization technique can be applied to the sparse subspace clustering algorithm. Consider the data set $X = [\mathbf{x}_1, ..., \mathbf{x}_N]$, where each $\mathbf{x}_j$ is a different data entry. The Elastic Net model computes

$$\mathbf{c}^* (\mathbf{x}_j, \mathbf{X}_{-j}) := \arg\min_{\mathbf{c}} f(\mathbf{c}; \mathbf{x}_j, \mathbf{X}_{-j}). \tag{2.8}$$

where $\mathbf{X}_{-j}$ is $\mathbf{X}$ without the j-th column. When using this technique to construct the matrix C, we can balance the conectiveness of the clusters and the subspace preserving property while still having a lower computational complexity than the ℓ_0-norm optimization. In this work we use the implementation from You et al. [39].

2.3.4 Hierarchical Clustering

The final clustering technique that will be covered in this work is hierarchical cluster, sometimes also called agglomerative clustering [41]. The hierarchical clustering algorithm consists of building a binary merge tree, called a dendrogram. The leaves of this tree are all the data points in the data set. Every level going up through the tree merges the two most similar clusters. Such a dendrogram is shown in Figure 2.10. In other words, hierarchical clustering is a bottom up method for the clustering problem. The individual steps in the hierarchical clustering algorithm are described as follows:

- Treat each individual data point as its own cluster.
- Calculate the similarities between clusters. The default affinity metric between clusters is the euclidean distance: $d = \sqrt{\sum_i (y_{i1} - y_{i2})^2}$.
- Merge the clusters with the closest cluster centers.
- Repeat until desired amount of clusters is reached or until the smallest similarity is above a certain threshold.

Some advantages of the hierarchical clustering algorithm are a more intuitive relation between data points and clusters, visualized in a dendrogram. It is also possible to let the algorithm stop at an ideal amount of clusters during the clustering process. When a similarity threshold is specified it is possible to stop merging clusters when their distance is above that threshold. The biggest downside of hierarchical clustering is the computational complexity for large datasets or high feature dimensions. The hierarchical clustering algorithm used in this work is the agglomerative clustering from Scikit-Learn [35].

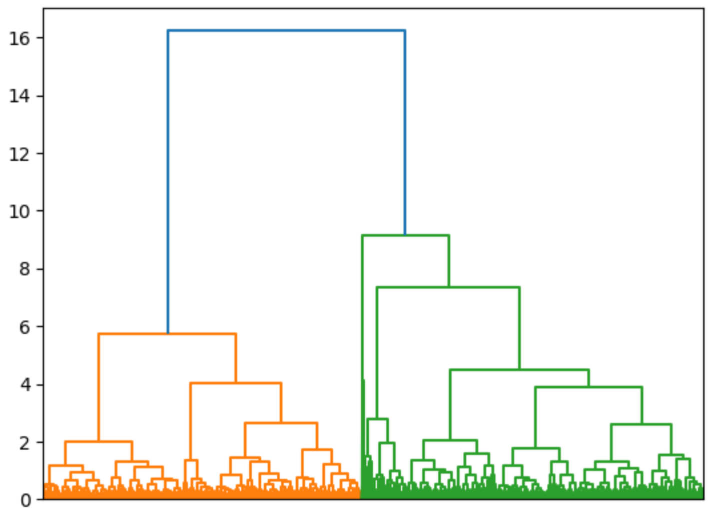


Figure 2.10: A dendrogram built using the hierarchical clustering algorithm on a metallurgical dataset. This figure shows information of when the different clusters were merged.

2.4 Conclusion

In this chapter the knowledge required for this thesis was presented. The necessary metallurgical background was provided to understand how the toughness property of a metal is defined and how it can be measured with the Charpy test. The microstructural characteristics visible on microscopic images that influence the toughness were presented and discussed. Then different techniques to transform images into a low-dimensional representation were analyzed. Finally an overview on four clustering methods was given with the goal to gain insight in their base principles.

These three different subjects were researched to gain all knowledge required to combine them into one framework. In the following sections we will analyze the ability of the given feature extraction techniques to compress the dimensions of microscopic metallurgical images, but still retain the necessary information relating to the toughness property of given material. Then we will cluster the resulting feature vectors and compare the different clustering algorithms based on the resulting cluster quality metrics and the distribution of the toughness for each cluster.

3 Proposed Methods

This Chapter presents our contribution to pattern recognition in microscopic images from metallurgical research. The main focus of our work is to utilize and assess the potential of different feature extraction and clustering techniques for analyzing metallurgical images. These techniques have been sourced from relevant literature and previous applications with similar objectives. We explore various combinations of feature extraction and clustering techniques to classify microscopic images based on the toughness property of the metal samples. To the best of our knowledge there have been no reported works that analyze the metallurgical properties of materials from fracture images. So the goal of these methods is to develop a framework that is capable of evaluating the properties from fracture images and analyzing which techniques can achieve the best results in this context.

Our evaluation process consists of two steps. Firstly, we employ a dataset comprising randomly selected metal sample images to evaluate the techniques in an unsupervised manner. The primary objective is to maximize the quality of clustering. To achieve this, we assess different combinations of algorithms and fine-tune their parameters to identify the optimal configuration yielding the highest quality clusters. In the second step, we utilize a second dataset is composed with metal samples chosen specifically for their diverse toughness properties. This dataset is employed to evaluate the unsupervised algorithms based on their ability to effectively group the samples according to similar properties. By employing these evaluation steps, we aim to contribute to the advancement of pattern recognition in metallurgical research, particularly in the analysis of microscopic images.

3.1 Evaluating Clustering of the First Dataset

The following method is developed to cluster microscopic images of metallurgical fracture surfaces. It is an unsupervised method so there is no ground truth or information about the toughness properties of the metal samples available. The only possible evaluation, is the cluster quality. The method consists of the following steps: dividing the image into patches, feature extraction, clustering and finally cluster evaluation. The goal of this method is to optimize and compare the different combinations of feature extractors and clustering techniques and to find what the optimal number of clusters is for these techniques.

3.1.1 Division into Patches

The fracture surfaces that are analyzed are obtained using a Charpy test, described in Chapter 2. During the breaking process of the metal sample, the actual stresses on the fracture line may vary. This is caused by: imperfections in the material, deformation and other factors during the crack propagation. For this reason a material may behave more ductile or brittle in certain areas of the fracture surface. There is thus a need to be able to differentiate between different parts of the image in order to classify the differences in behavior. The final clustering classification of a metal sample can than be decided using a majority vote between the clustered patches.

By dividing the images into patches the analysis becomes localized instead of the whole image which may contain sections with a different behavior, the analysis becomes localized and isolates these areas. There are other additional benefits such as increased computational complexity and an increased amount of data points for the unsupervised machine learning algorithms later in the process.

When dividing images into patches it is important to choose the right size. Take too small patches and they will not contain enough information, to big and the benefits of working with patches decrease. In this work patches of shape (64×64) pixels where chosen, as an optimal compromise to contain enough information, but still be of a small enough size. An added benefit of taking this patch size, is that the JPEG block size is often a divisor of 64, minimizing the amount of blocking artifacts in the input images.

3.1.2 Feature extraction

Microscopic metallurgical images have a high dimensionality, the images used have a shape of (832×1280) and have does a feature vector of 1 064 960 dimensions. These contain extra information that is not relevant for toughness classification. This is still the case after dividing the images into patches of shape (64×64) as described in the previous part. By reducing the dimensions of the data we can eliminate unnecessary data and retain the useful information. The reason we want to decrease the amount of features is the curse of dimensionality, meaning that the algorithms will need more time for training and classification. A high dimensionality also correlates to a higher chance of the algorithms overfitting and to an increased amount of noise present in the data, as each feature will always introduce some noise. The necessary information relating to toughness, resides in a lower-dimensional space than the full image dimensions. In summary the goal of the feature extraction step consists of creating a compact representation of the image patch that still contains all necessary features to classify the images into meaningful clusters.

To create these compact data representations feature extraction techniques are used. These aim to extract the features that reflect the intrinsic contents of the images that relate to the application at hand [18]. This work uses features often used in pattern recognition, to try and identify the different patterns that appear based on the toughness property of the material. The algorithms must thus be able to process the form of the microstructure and and the structures present in the image that influence the toughness of a material. The algorithms used in this work, are covered in detail in Chapter 2, but for completeness we will list them here:

- First Order Statistics (FOS),
- Histograms of Oriented Gradients (HOG),
- Grey Level Co-occurence Matrix (GLCM),
- Local Binary Pattern (LBP),
- Discrete Wavelet Transform (DWT),
- Weyl Transform,
- Autoencoders.

These feature extraction algorithms will each be optimized and evaluated in combination of a cluster to select the best performing algorithm for toughness classification.

3.1.3 Clustering

After feature extraction, unsupervised classification is performed. The reason for the choice of unsupervised techniques is twofold. First, due to the lack of previous work on the same subject it is unclear how many classes and what kind of classes can be classified based on the toughness property. Second, labeling all the images into their respective classes is a difficult and time consuming process. Using clustering techniques, it is possible to find a classification based on the latent structures in the data. Because the unsupervised classification does not try and replicate a currently existing classification method, the results may also be interesting for experts in the metallurgical field.

The clustering algorithms used are also described in Chapter 2, but for completeness they are listed below:

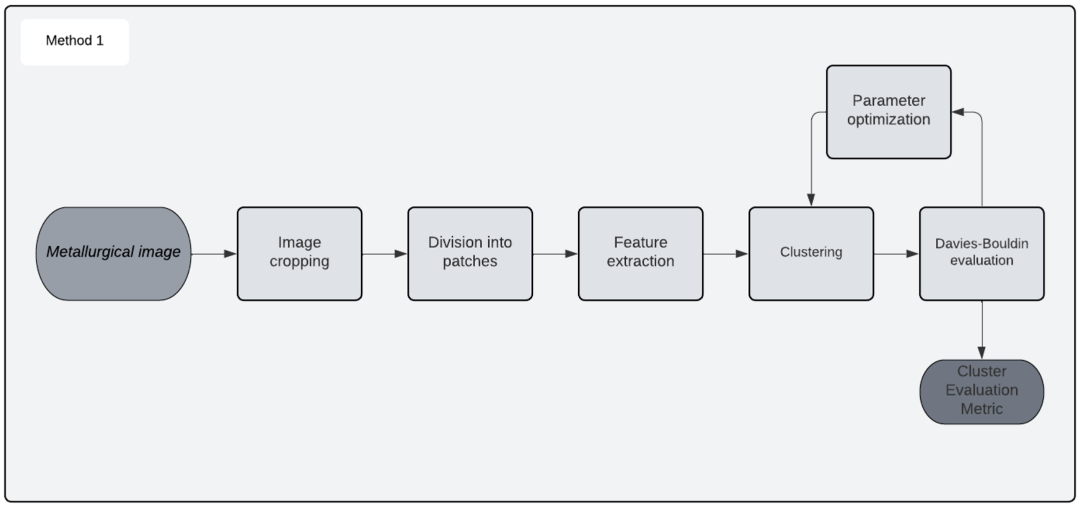


Figure 3.1: An overview of the first method we developed.

- K-means clustering,
- Sparse subspace clustering,
- Hierarchical clustering.

The reason for comparing all these algorithms is the fact that each of these algorithms uses different methods to express the similarity between data points. The clustering results may differ and some algorithms may find a better relation to the toughness property than others. Beforehand there is no real indication to which one will perform better.

3.1.4 Clustering evaluation

The clustering algorithms must be optimized to find their best solution and the resulting solutions must be compared to find the best algorithm. Due to the fact that clustering is an unsupervised machine learning method, there are no labels to calculate the accuracy of the clustering algorithms. For this reason, other evaluation methods must be used. The two most used metrics for cluster evaluation are the Silhouette score [42] and the Davies-Bouldin index [43]. The silhouette score is defined as

$$s = \frac{b-a}{\max(a,b)},$$

where a is the mean intra-cluster distance and b the mean nearest cluster distance. The default metric used for these distances, is the euclidean distance. The downside of the silhouette score is its high time complexity for big data sets, making the optimization process for all algorithms very time consuming. The second evaluation metric is the Davies-Bouldin index [43]. The lower the index, the higher the inter-cluster distance is. The Davies-Bouldin index is formally defined as

$$db = \frac{1}{n} \sum_{i} \max_{j \neq i} \left(\frac{\sigma_i + \sigma_j}{d\left(c_i, c_j\right)} \right),$$

where n is the number of clusters, c_i is the cluster center of cluster i, σ_i is the dispersion of a cluster and $d\left(c_i, c_j\right)$ is the distance between two cluster centers. The advantage of using the Davies-Bouldin index, is that the time complexity is significantly lower than the silhouette score, while the results are often similar, as reported in literature [44]. In this work the Davies-Bouldin index is used for the improved time complexity. Using the index it is possible to determine an optimal amount of clusters and compare the clustering quality to other feature extraction or clustering methods. However, it must be noted that these evaluation metrics optimize the separation and compactness of the resulting clusters, it has no direct relation with the toughness property of the metals. We can thus use these to optimize the algorithms and compare quality of the resulting clusters, but a worse evaluation does not necessarily mean that the combination of feature extractor and clustering algorithm has worse results relating to the toughness property. A summary of the first method is presented by Figure 3.1. In this figure an overview is given of all necessary steps in this method and where the optimization process is located.

3.2 From Clusters to Toughness Classification

The goal of the previous method was to optimize the different feature extraction and clustering algorithms. In this method a second data set is used where the labels of toughness per metal sample is provided. A second difference for this data set is, that it is composed of metals chosen for their toughness properties and contain the same material multiple times, but tested at different temperatures. For this reason we will perform the previous method again on the data set, to compare the differences between feature extraction algorithms when the data contains more samples that are closely related and for which combinations of algorithms this impacts the clustering quality.

As mentioned in the previous method, the images will be divided into (64×64) shaped image patches. These patches will undergo the feature extraction algorithms to transform them into compact feature vectors. Subsequently, we will cluster these low-dimensional representations. For each combination of algorithms, we will optimize the number of clusters to achieve the lowest Davies-Bouldin index. By comparing these results with the other algorithms on the dataset and outcomes from the previous test, we can determine which algorithms are capable of achieving a distinct separation between the samples.

3.2.1 Image Classes

When the clustering process is complete, it produces numerous image patches with cluster labels. However, evaluating these results solely based on the toughness properties present in each cluster of patches, is not directly possible since the objective is to group the images and not the individual patches. To assign a label to each image, the patch results need to be combined in a manner that considers the majority of the patches. A disadvantage of applying this method is that when a cluster only contains a small amount of sample, that cluster may not appear in any of the final image clusters.

To achieve this, a majority vote approach is employed. All labels present in the patches of an image are counted, and the label that occurs most frequently is considered the final clustering label for that image. Other methods are often not viable since there is no known relationship between the clusters. However, it is possible to calculate the algorithm's certainty for each cluster label. The certainty can be measured by determining the percentage of patches with the most frequent label. If the most occurring label has only a slight majority, it indicates that a significant number of patches were assigned to different clusters, resulting in less confidence compared to a scenario where all patches were assigned to the same cluster.

In summary, the majority vote method is employed to combine patch results and assign labels to images. The certainty of the algorithm can be assessed by analyzing the percentage of patches with the most frequent label, providing insight into the algorithm's confidence level. A small margin in the most occurring label suggests less certainty due to patches being assigned to different clusters, while a larger majority indicates higher confidence when all patches belong to the same cluster.

3.2.2 Toughness evaluation methods

The evaluation of the toughness properties of the samples in each cluster is not as straightforward as it seems. One of the goals of this thesis is to assess how many classes can be found in the data set, based on the toughness property. These classes are thus not predefined and a metric like accuracy measurement is not an option. In addition, the average toughness value measured in each cluster does not contain enough information to actually evaluate if the cluster contains samples that are related in respect to toughness. To enable a good evaluation of the resulting clusters, the distributions of the toughness properties per cluster should be compared.

A boxplot is an excellent solution to compare the toughness distributions per cluster. It shows the average, maximum and median value, while also showing the first and third quartile. The information displayed by a boxplot allows us to evaluate if there is a good separation between

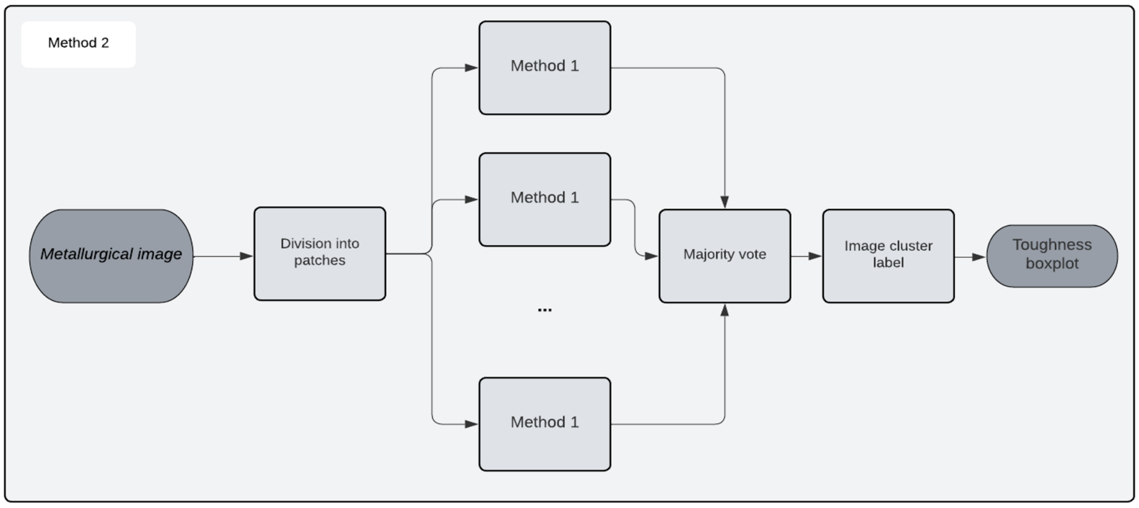


Figure 3.2: An visualization of the second method developed.

the distributions and how narrow the window of toughness values is per distribution. These boxplots can be generated for the different algorithm combinations and numbers of clusters, to evaluate which models can make the best separation between the different sample based on their toughness. Finally using these plots enables for an assessment of which pipelines are able to capture the toughness data from the microscopic fracture surface images and grouped these samples then correctly together. The certainty measure is an extra metric that shows us how stable the clustering of the model is. A visualization of the second method is shown in Figure 3.2. The figure shows the developed steps for this method and visualize the majority voting process that leads to the cluster label for one image instead of each of the patches.

3.3 Conclusion

The described methods share a common objective: to identify the optimal combination of feature extraction and clustering algorithms that can effectively group metal samples with similar toughness properties. The first method focuses on an unlabeled dataset, aiming to optimize the algorithms and achieve the highest possible cluster quality given this dataset. The second method involves evaluating the cluster labels in relation to the toughness property of each image.

In the second method, the distributions of toughness for each cluster are visualized using a boxplot, allowing for a comparison between clusters. This comparison enables the assessment of how well the clustering algorithm has grouped metal samples with similar toughness properties.

By utilizing the first method for parameter optimization and the second method for toughness evaluation, both the compact data representation obtained through feature extraction and the clustering algorithm itself can be evaluated in the context of the given subject. This comprehensive evaluation approach facilitates the selection of the best combinations of feature extraction techniques and clustering algorithms, providing insights into their effectiveness in grouping metal samples based on toughness properties.

4 Experimental Results and Discussion

Here we present a detailed account of the experimental results obtained from applying the developed methods to metallurgical images. The experiments were conducted with dual objectives. Firstly, the aim was to optimize the proposed method by identifying the ideal combination of feature extraction techniques and clustering algorithms, utilizing optimally tuned parameters. This optimization process ensures that the method performs at its best capability.

Secondly, the goal was to establish a framework that represents an advancement in the field of metallurgical research. To evaluate the efficacy of the framework, the relationship between the clustering classification and the actual properties of the metal samples from the Charpy tests was assessed. By examining this relationship, the framework's ability to accurately capture the real properties of the metal samples could be gauged.

In this Chapter, the techniques were fine-tuned and optimized to enhance their performance, taking into account the feedback received from the experimental results. The overall objective was to develop a robust framework that improves upon existing methodologies in the field of metallurgical research.

4.1 Experimental Setup

In this section, the experimental setup, used in this work, is covered. The goal of the experiments is to evaluate which combination of input data, feature extraction techniques and clustering algorithms is able to separate the metal samples based on their metallurgical properties, specifically the toughness property. Firstly, the data sets are discussed and their particular characteristics. Secondly, the implementation of the feature extraction techniques on the data set is discussed. Furthermore, the clustering algorithms and the parameter tuning of these algorithms are presented. Finally, the evaluation process of the previous algorithms is covered.

Parameter	Definition
Acceleration voltage	The voltage at which electrons are accelerated.
Spot Size	The diameter of the spot that can be captured.
Magnification level	Amount of times the image is magnified.
Working distance	The distance between the aperture of the microscope and the specimen.

Table 4.1: The main parameters and their definition of a scanning electron microscope.

4.1.1 Data Sets

In this section we present the experimental setup that will be used while conducting the experiments described in this work. In this work we define a model or a pipeline the process that divides the image into patches, extracts features, and clusters these feature vectors. As a baseline model we consider first order statistics with k-means clustering. These are the two most simple algorithms and are expected to be outperformed by most other algorithms. First we will use all feature extraction methods and image magnification levels in combination with k-means. This gives us already an insight in how the feature extraction methods compare to each other and what magnification level should be used going forward. Then the other clustering techniques are evaluated to get the complete results.

4.1.1.1 The Unlabeled Data Set

We refer to this data set as the unlabeled data set as no information about the metal samples is provided. This data set contains SEM microscopic images from fracture surfaces of metal samples. The only metadata provided, for each image, are the ID of the metal sample and the settings of the SEM. The parameters provided in the scanning electron microscope are shown in Table 4.1. Increasing the acceleration voltage reduces the spot size. A small spot size captures a smaller region of the specimen, but achieves a higher signal to noise ratio. For this reason a small spot size is necessary for large magnification levels. The working distance parameter is related to the depth-of-field.

The data set contains 784 images, from 53 different metal samples. Statistics of the number of images per sample and the previously described parameters can be found in Table 4.2. For the clustering process the most important parameter is the magnification level, as images from the same sample, but with different magnifications, will not seem similar in images for the clustering algorithms. For this reason we split the data sets into sets of the same magnification level. We chose to use three magnification levels in the experiments. A very high magnification level, one relatively small level and something in between. Based on the most occurring magnification levels shown in Figure 4.1. We chose the levels: 100, 500 and 1500. One of each magnification

Parameter	Minimum	Average	Maximum
Images per sample	6	14.8	29
Acceleration Voltage	20	20	20
Spot size	50	52.9	75
Magnification level	30	480.8	5000
Working distance	9	11.1	14

Table 4.2: Statistics on the SEM settings in the unlabeled data set.

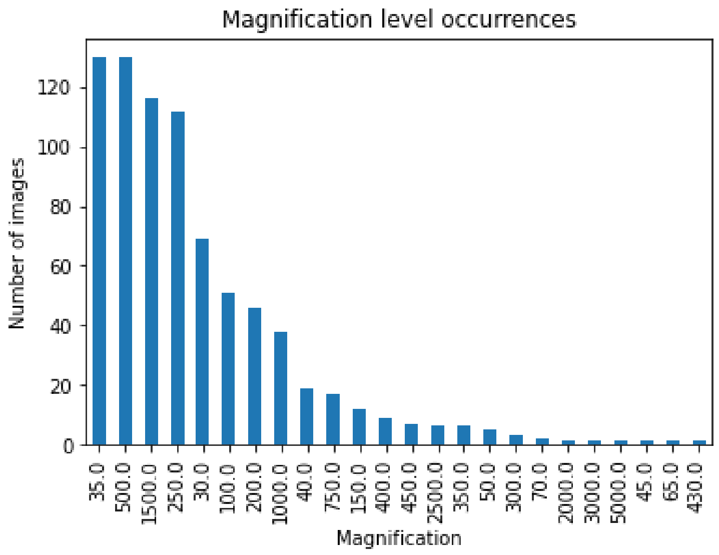
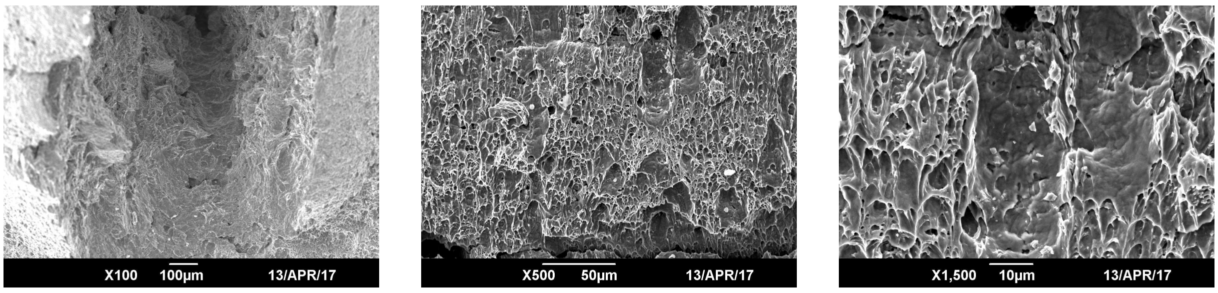


Figure 4.1: The occurrences of magnification levels in the unlabeled data set.



(a) Metal sample with magnification level 100×. (b) Metal sample with magnification level 500×. (c) Metal sample with magnification level 1500×.

Figure 4.2: The same metal sample with magnification 100×, 500× and 1500×.

levels is shown in Figure 4.2.

Parameter	Description
Absorbed Energy	The energy absorbed by the sample during the Charpy impact test.
Specific Absorbed Energy	The absorbed energy per unit mass of crushed material.
Temperature	The temperature of the sample during the test.

Table 4.3: The extra data provided in the labeled data set with their description.

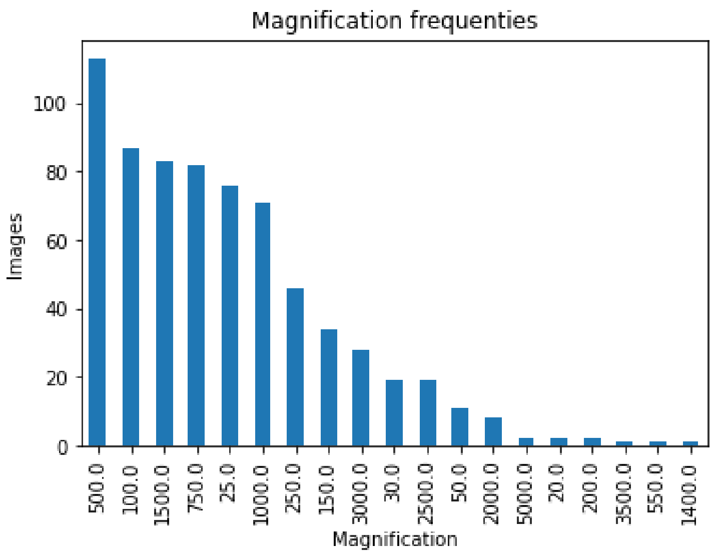


Figure 4.3: The occurrences of magnification levels in the unlabeled data set.

4.1.1.2 The Labeled Data Set

The second data set is very similar to the first unlabeled set. A collection of microscopic images of metal samples broken in a Charpy impact test. The difference is that here information about the metallurgical images is provided together with the images. The extra data and their description is listed in Table 4.3.

Another notable difference is that this data set, contains multiple samples of the same material, that were tested at different temperatures. There is less variety in materials available in this set.

The labeled data set contains less images than unlabeled set, but there is again a good amount of images with magnification levels 100, 500 and 1500. The magnification level occurrences are shown in Figure 4.3. Therefore the same magnification levels were used in this data set.

To compare the toughness properties of each cluster, the specific absorbed energy in the Charpy impact test, is used. The specific absorbed energy as this value takes into account variations in sample size and mass.

4.1.2 Data Preprocessing

For this data, there is only two steps necessary in preprocessing. Pixel values range from zero to 255, but these were not rescaled, as some of the feature extraction techniques use this range as standard input. The first necessary step was the removal of the textual data at the bottom of each image. This was achieved, by cropping the bottom 128 pixels from each image. Secondly each image was divided into image patches of shape (64×64). For each image 260 patches are obtained.

4.1.3 Feature extraction

The feature extraction techniques used in this work are described in 2.2, but we will show the implementation of these algorithms here.

- **FOS**: the baseline feature extraction technique. Extracts 6 first order statistical features from each image patch. It is a simple technique that should be outperformed by the other feature extraction methods as it is often not capable of capturing microstructural features.
- **HOG**: transforms an image patch by extracting the gradients in cells of (16×16) pixels. A visualization of the gradient extraction on a full image is shown in Figure 4.4. These gradients are then collected into a histogram to create the feature vector. The reason HOG could work well in this instance, is that the microstructural features all have very harsh gradients making them easy to pick up for this feature extraction algorithm.
- **GLCM**: extracts second order statistics of the image. GLCM maps spatial relations between pixels. From this matrix 5 features are extracted, namely, correlation, contrast, energy, homogeneity and entropy. These features have a very low dimension, but give a good overview of the geometry in the microstructure.
- **LBP**: consists making an encoding for each pixel in de image using a binary comparison with p points evenly spread on a circle with radius r. In this work, the best parameters were found to be $p = 8$ and $r = 1$. All the frequency of each pixel encoding is counted and combined into a histogram that can then subsequently be used as a feature vector.
- **DWT**: Uses Haar wavelet transform and filters to create an image only containing certain frequencies. From these images the mean and standard deviation are extracted to form the feature vector. A visualization can be seen in Figure 4.5. The Haar transform is used for edge detection, so all the edges in the microstructure are highlighted while other noise is removed.

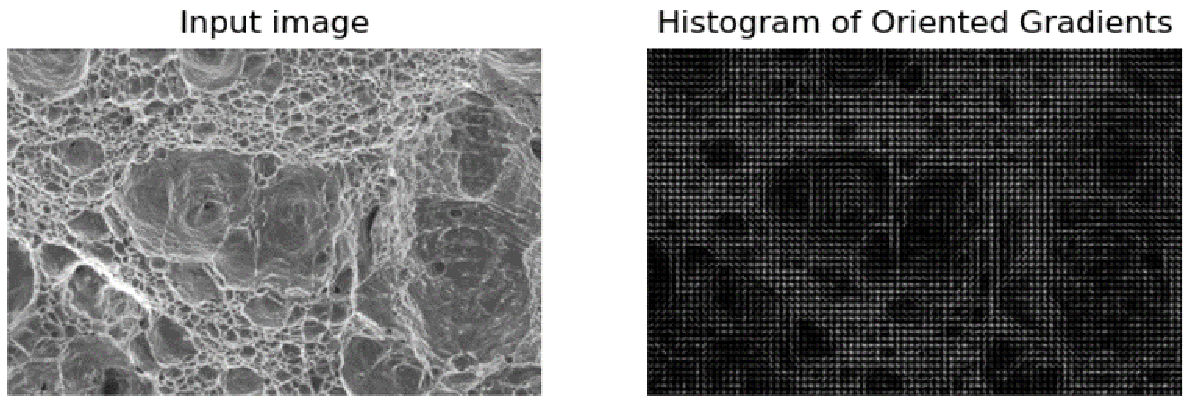


Figure 4.4: A visualization of the gradients obtained by HOG feature extraction.

- **Weyl Transform**: Creates a feature vector of an image patch based on the autocorrelations between pixels. This feature extraction technique was implemented in python for this work, so that the full pipeline can be executed with python only. For the Weyl Transform the results of overlapping patches can be averaged for increased accuracy. For this reason the Weyl Transform is the only feature extractor with overlapping patches.
- **Convolutional Autoencoder**: Learns a feature vector for the data set, by learning how to reconstruct the image after passing it through a compression bottleneck. The architecture of the autoencoder is shown in Figure 4.6 and an input and reconstructed image is shown in Figure 4.7. The reconstruction is a bit more blurred, but the microstructure from the first image is clearly the same. We can thus conclude that this autoencoder is able to compress the input without loss of microstructural information.

The aim of using these feature extraction techniques, is to create a noise free and compact representation of the image patches. The sizes of the resulting feature vectors in this work are shown in Table 4.4. These vectors are a huge reduction in dimensions as the original feature vector size of an image patch is 4096 and the biggest feature vector of the autoencoder is still a reduction of almost 94%.

4.1.4 Training process

In training it is necessary to tune the parameters of the algorithms, in order to find the optimal amount of clusters, used in the clustering process. To find the optimal amount, the algorithms are trained multiple times, with the number of clusters ranging from two to 9. After each training run the Davies-Bouldin index is calculated to compare with other training runs. The number of clusters that achieves the lowest Davies-Bouldin index is considered the best model.

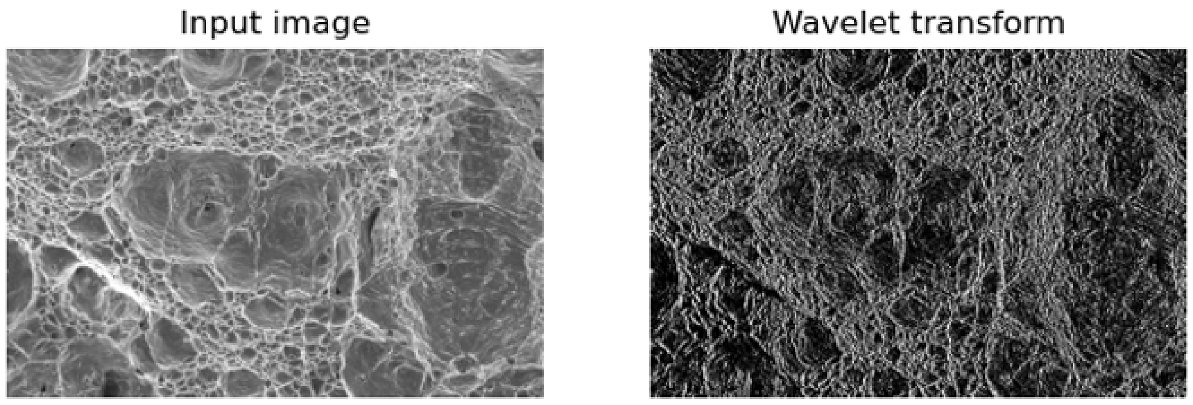


Figure 4.5: A visualization of the haar wavelet transform image in the HH quadrant.

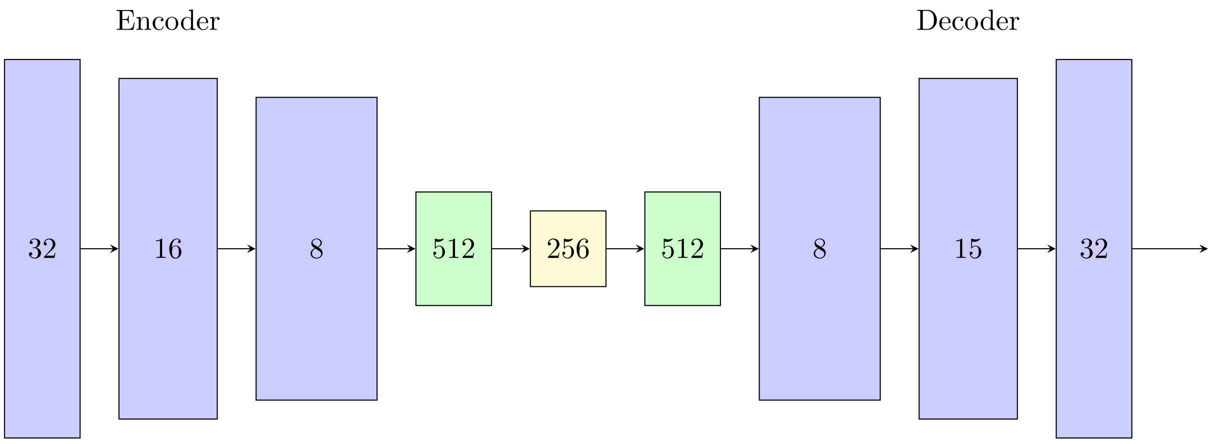


Figure 4.6: A visualization of the network architecture used for the autoencoder. The purple elements are convolutional layers. The green elements are fully connected layers and the yellow is the latent space

Feature extraction	**Vector Size**
Original Patch	4096
FOS	8
GLCM	5
HOG	128
LBP	128
DWT	6
Weyl	24
Autoencoder	256

Table 4.4: The feature vector sizes after each extraction technique.

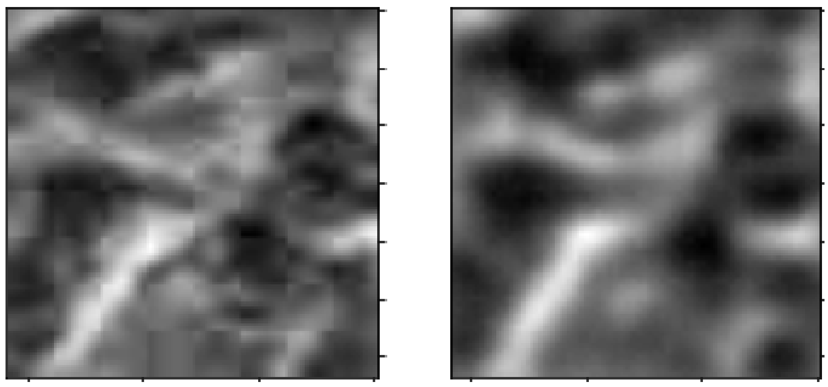


Figure 4.7: A comparison between the input in the autoencoder (left) and the reconstructed image (right).

4.1.5 Evaluation methods

In this work we evaluate two aspects of each model: the clustering quality and the relation between clusters and the toughness properties of a metal samples. Each of these aspects are connected, but can be evaluated individually.

For all clustering models, the Davies-Bouldin index was already calculated to define the optimal number of clusters. These indices can be compared over the different clustering algorithms and feature extraction methods. Were a lower Davies-Bouldin index indicates that there is a high inter-cluster distance and a low intra-cluster variability. A model lower DB-index is assumed to be capable of creating a nice separation between the different image classes. For a visual evaluation method, the feature vectors are reduced into two dimensions using a T-stochastic neighbor embedding (TSNE) [45], with random initialization, and principal component analysis (PCA) [46]. Using these two techniques it is possible to visualize the clustering results in a 2-dimensional scatter plot.

The comparison of the clusters to their respective metallurgical classes is difficult. Firstly as there is no previous research to that classifies metal samples into classes of toughness values, we can not make buckets to use accuracy as estimate for the clusters. Secondly an average value for each cluster does not contain enough information to evaluate the clustering process as some extremes may shift the average or the toughness values are not compactly grouped in a cluster and the average no representation for the majority of metal samples classified into that particular cluster. To enable us to evaluate the clusters thoroughly, we chose to plot the toughness values per cluster in a box plot. A box plot allows us to compare the distribution of values in each cluster and make a more informed evaluation.

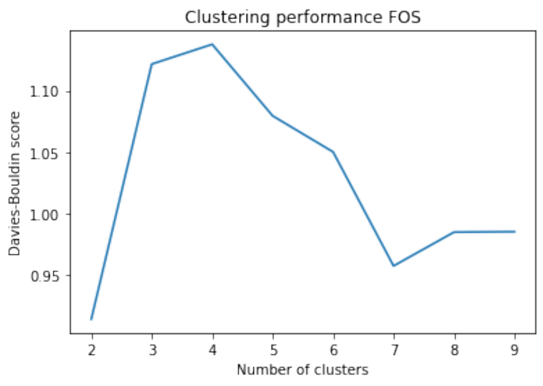
Feature extraction	**100×**	**500×**	**1500×**
FOS	0.85	0.91	0.91
GLCM	0.50	0.50	0.50
HOG	1.25	2.86	3.03
LBP	0.53	**0.44**	0.82
DWT	0.76	0.75	0.76
Weyl	0.88	0.86	0.77
Auto-encoder	-	2.6	2.5

Table 4.5: The Davies-Bouldin evaluation of the feature extraction techniques together with k-means, for each image magnification. The data used came from the unlabeled data set

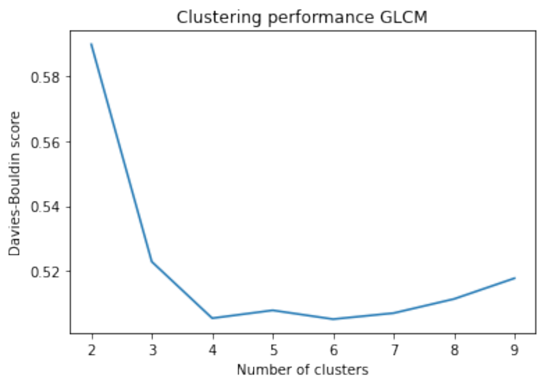
4.2 Optimizing the Cluster Quality

The initial experiments aimed to evaluate the impact of the magnification level of input images and various feature extraction techniques. For each magnification level, al feature extraction techniques were individually optimized and evaluated using the Davies-Bouldin index. The best performance was compared to a baseline model using first-order statistics. The best Davies-Bouldin index for each combination of magnification and feature extractor is presented in Table 4.5. Based on the cluster quality, it can be concluded that four techniques consistently outperformed the baseline model across all magnification levels: GLCM, LBP, DWT, and Weyl transform. Among these techniques, LBP performed the best for a magnification level of 500, while GLCM outperformed the others at different magnification levels. A scatter plot of the clusters from FOS feature extraction together with two best performing algorithm, GLCM and LBP, and HOG, are shown in Figure 4.9. In this figure Is illustrated why HOG has such a high Davies-Bouldin score in comparison to the other algorithms, and that GLCM can be reduced into two dimensions in a very particular manner. The optimization process contains valuable information as well. It shows us for each feature extractor how well it can handle clustering with a different number of clusters. In this problem setting, a higher number of clusters is better as it can result in a more precise grouping of the metal samples. In Figure 4.8, a comparison of the DB-optimization for each of the three best feature extraction techniques and the FOS baseline, is shown for the 500× magnified images. In this image we can see that the most stable algorithm for a high number of clusters is GLCM that performs best for 6 clusters. The other algorithms start relatively low, but the Davies-Bouldin index rapidly increases with the number of clusters. For each algorithm and image magnification, the optimal number of clusters is shown in Table 4.6. This table shows that most algorithms perform best for two clusters, but in some configurations the algorithm perform better with higher numbers.

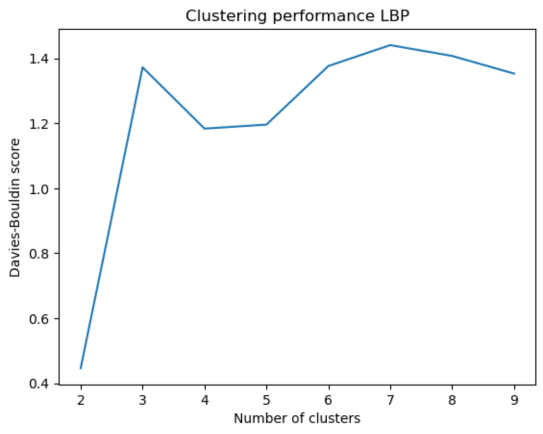
Going further mostly the magnification level of 500× is used. The reason for this is that for the



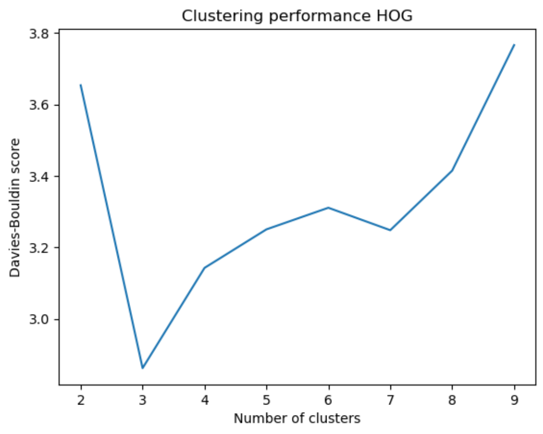
(a) FOS clustering.



(b) GLCM clustering.



(c) LBP clustering.

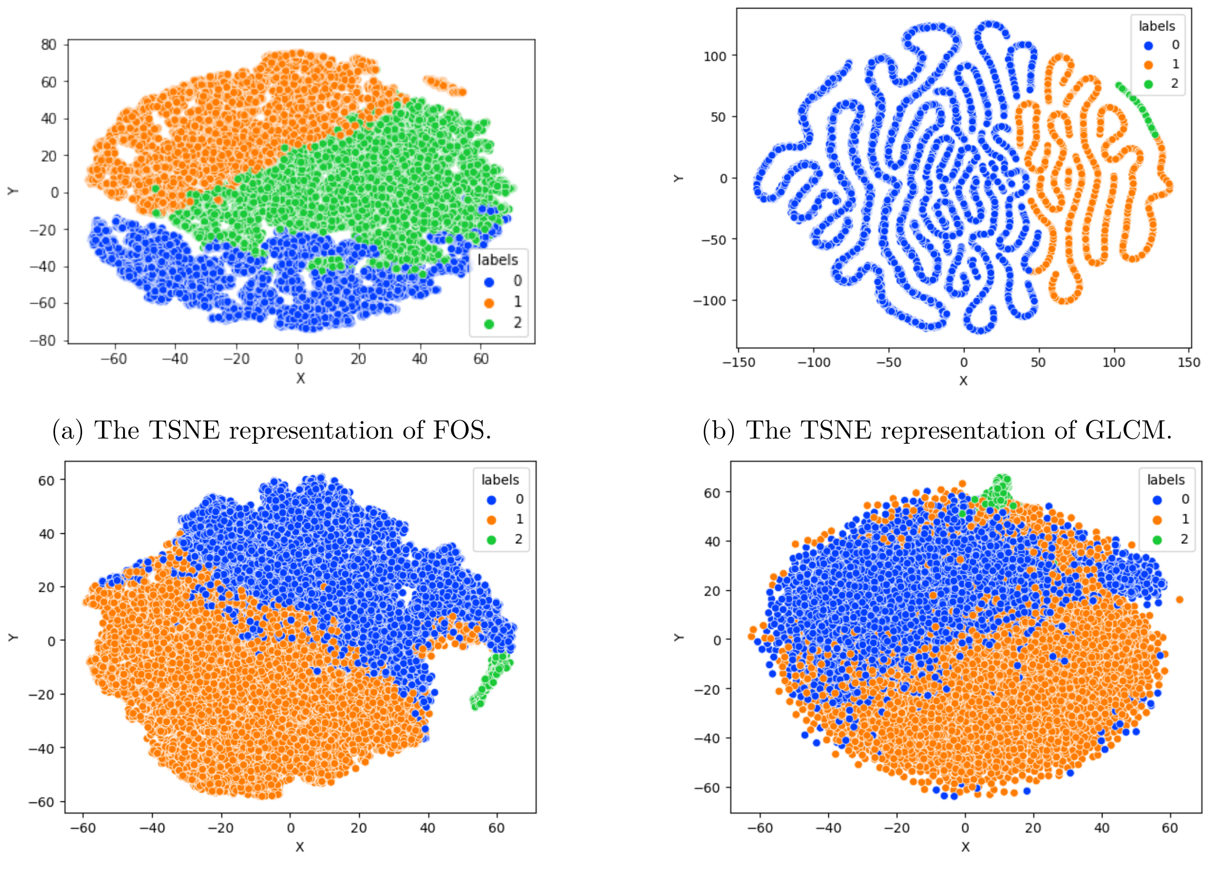


(d) HOG clustering.

Figure 4.8: The Davies-Bouldin index in function of the number of clusters using k-means and an image magnification of 500 for three clusters. The included results are the FOS baseline and the two best performing feature extractors. The final result included is the HOG feature extraction to illustrate clustering results with a high Davies-Bouldin index.

Feature extraction	**100×**	**500×**	**1500×**
FOS	3	2	2
GLCM	**8**	4	6
HOG	2	2	4
LBP	2	2	2
DWT	4	2	2
Weyl	3	3	2
Auto-encoder	-	2	2

Table 4.6: The optimal amount of clusters for the each feature extraction technique per magnification level for the first data set.



(a) The TSNE representation of FOS.

(b) The TSNE representation of GLCM.

(c) The TSNE representation of LBP.

(d) The TSNE representation of HOG.

Figure 4.9: A clustering visualization using TSNE dimensionality reduction. The results included are that of the baseline and the two highest performing feature extractors and HOG of an image magnification of 500.

best performing clustering algorithms this input generally results in the best clustering qualities.

Next the clustering qualities of different clustering methods were evaluated. This time we optimized each feature extraction technique with the following clustering algorithms: k-means, hierarchical clustering, elastic net sparse subspace clustering subspace clustering. These different methods were chosen because of their differences in basic principles. k-means optimizes the cluster centers and assigns all data points closest in euclidean distance, hierarchical clustering also is based on the euclidean distance, but works in a bottom-up manner instead of top-down like k-means. The final two clustering algorithms solve the self-expressiveness optimization problem where sparse representations are sought. Orthogonal matching pursuit solves this in a greedy manner and Elastic net optimization relaxes the problem to a convex problem. The Davies-Bouldin evaluation of these models can be found in Table 4.7.

From this table we can conclude that in most cases k-means is the best clustering algorithm,

Feature extraction	K-means	Hierarchical	Sparse subspace clustering
FOS	0.91	0.93	1.93
GLCM	0.50	0.48	0.69
HOG	2.86	1.25	1.60
LBP	**0.44**	0.62	1.12
DWT	0.75	0.73	2.0

Table 4.7: The Davies-Bouldin index of the different clustering algorithms using the first data set, with an image magnification level of 500×.

except for the feature extraction method GLCM and DWT. For these techniques is hierarchical clustering a small improvement. Sparse subspace clustering seems to always be outperformed by the other two algorithms.

From these experiments we can conclude that for this data set, LBP with k-means is the best performing combination. Shortly followed by GLCM with hierarchical and k-means clustering and DWT with Hierarchical clustering. HOG and the autoencoder perform significantly worse than the FOS techniques in all possible combinations.

We repeated the analysis of the influence of the image magnification levels on the different feature extraction techniques

4.3 From Clusters to a Toughness Classification

The toughness data from the second data set makes it possible to compare different clusters in respect to the specific absorbed energy in the samples. There are also other differences present in this data set, namely the composition. This data set contains many very similar samples with only small differences in chemical composition, processing parameters or even the same, but tested on a different temperature. This causes the data set to have a more natural division of clusters as these similar samples can be more easily grouped together.

4.3.1 Toughness Classification using K-means Clustering

In this section we will repeat the steps from the previous section to compare the results of the two data sets. This will give us a better understanding on the factors influencing the performance of the clustering algorithms. These results are summarized in Table 4.8. Due to the bad performance of the autoencoder feature extraction in the first data set, it was decided not to retrain an autoencoder to obtain the latent codes for the second data set, for this reason

Feature extraction	**100×**	**500×**	**1500×**
FOS	0.20	0.19	0.53
GLCM	0.46	0.48	0.33
HOG	2.70	3.58	3.9
LBP	0.20	0.18	0.16
DWT	0.17	**0.16**	0.76
Weyl	0.88	0.86	0.77

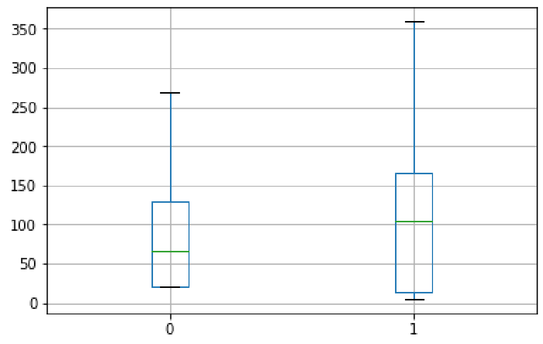
Table 4.8: The Davies-Bouldin evaluation of the feature extraction techniques together with k-means, for each image magnification. The used data came from the labeled data set

this technique is absent from the table.

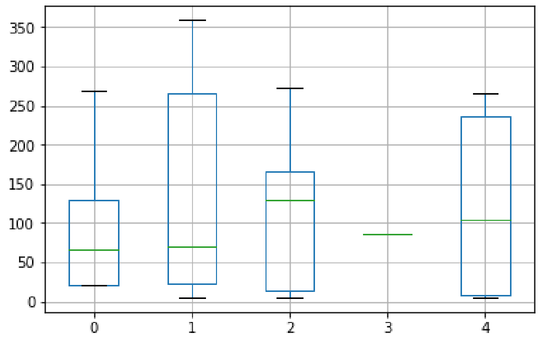
In this table we can see that the Davies-Bouldin evaluation of the cluster quality is significantly better than for the first data set. The main cause for this should be the more closely related samples in the second data set making the clustering process easier. However, some of the feature extraction techniques do not experience this improvement. GLCM produces, for the smaller magnification levels, very similar results and HOG performs even worse than on the first data set. By reducing the dimensions of the feature vectors with a T-Stochastic Neighbor Embedding, we could plot 2-dimensional visualizations of the clustering results.

With the new data set we now have access to the toughness values of each sample. As previously described we $(64, 64)$ clustered image patches of these samples. Before we could compare the toughness values, we had to recombine the patches into their respective samples. For this we used the most frequent cluster label per sample, as other methods such as averaging are not usable, because the average of cluster classes has no real meaning. The downside of recombining these patches before classification is that, when using a higher number of clusters, some clusters will not be present in the final classification. To combat this we also experimented with using the patch classification as the final result, but due to the high amounts of patches and inconsistencies between the images, this lead to very similar clusters. Using this technique it is also possible to estimate a certainty measure of a classification. This certainty is valuable information when using such a classification as it gives an insight on how similarly the patches are classified. This certainty measure is calculated as the percentage of patches that was classified in the same cluster as the complete sample. Where a high certainty means the model thinks similarly for all individual patches and a lower certainty means that a significant number of image patches were assigned to a different cluster.

In the following paragraph, the toughness classification results of the k-means clustering algorithms are discussed. Firstly, first order statistics had good clustering evaluation, but when comparing the toughness distributions over the clusters, we can see that this technique does not



(a) The toughness distribution of the FOS k-means clustering results for 2 clusters.



(b) The toughness distribution of the FOS k-means clustering results for 2 clusters.

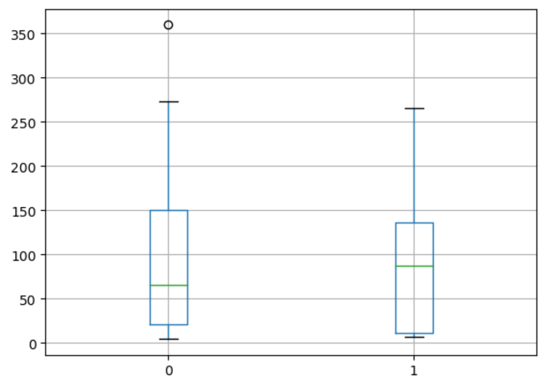
Figure 4.10: Two toughness distributions with a different number of clusters, using FOS and k-means.

actually capture enough information from the image to make an informed decision on toughness values. Most clustering results show to have a very similar distribution, as shown in Figure 4.10. From these results, we can conclude that, although FOS only achieves a slightly higher DB-score than the best performing feature extraction techniques, it does not translate to clusters of samples with a similar toughness. In the following we will cover the other feature extraction algorithms with k-means clustering.

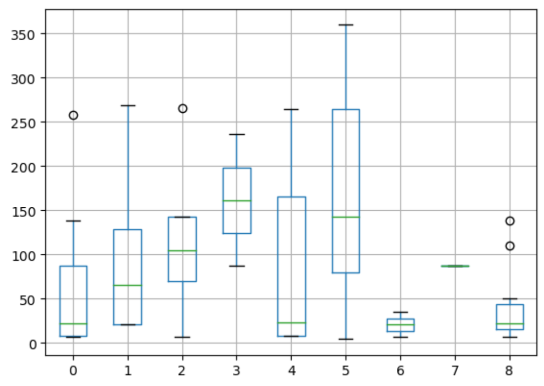
Firstly, we will discuss GLCM, this technique performed, in contrast to the other algorithms, very similar to the first data set in terms of clustering performance. When evaluating the toughness distributions of the clusters, we could see that a small amount of clusters was not able to separate the samples based on their toughness, but a larger amount of clusters has a more precise separation of clusters, a comparison of clustering with two clusters and with 9, is presented in Figure 4.11. The problem with these clusters is that there remain still some clusters that have a very broad distribution of samples. Clusters 4 and 5 in Figure 4.11b, do not provide any value in the classification process. The other cluster, however, have a relative narrow distribution of samples.

Histograms of oriented gradients is the next feature extractor. HOG did not achieve good DB-scores in both of the data sets. The distributions of the clusters are also not very distinguished from the other clusters, for all magnifications level inputs. The clustering distributions for 2 and five clusters are given in Figure 4.12. A major reason for this could be the low image qualities, as blocking and posterization artifacts introduce gradients that do not relate to the microstructure of the metal sample. These added gradients result in noisy feature vectors, making it very difficult to assign the samples into meaningful clusters. The algorithm partly succeeds for a higher number of clusters, as shown in 4.12b, but not as good as for example GLCM.

Local Binary Patterns achieved the best Davies-Bouldin indices for both data sets. This feature

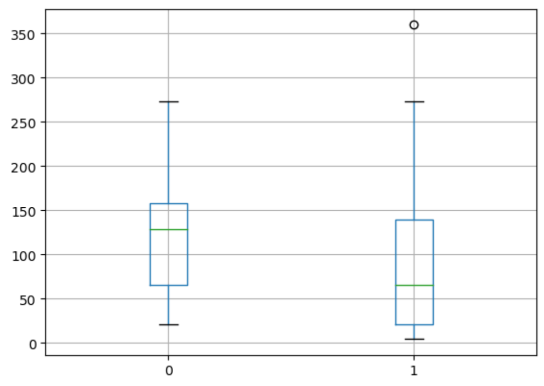


(a) The toughness distribution of the GLCM k-means clustering results for 2 clusters.

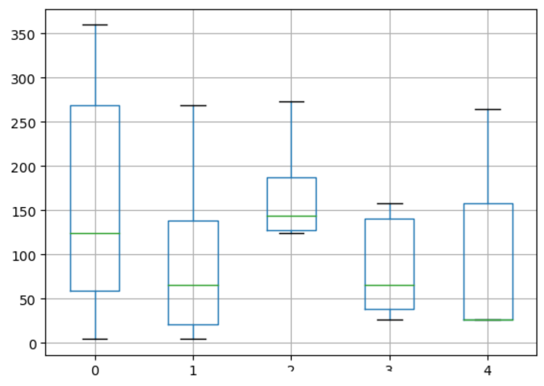


(b) The toughness distribution of the GLCM k-means clustering results for 2 clusters.

Figure 4.11: Two toughness distributions with a different number of clusters, using GLCM and k-means.



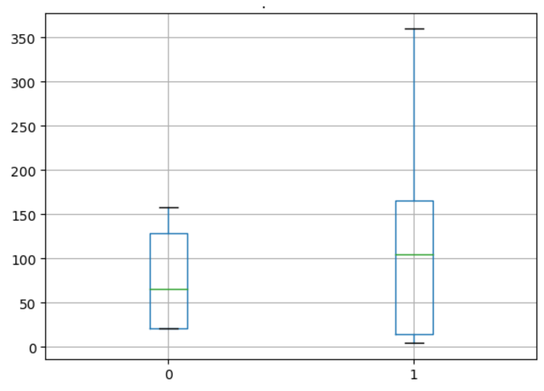
(a) The toughness distribution of the HOG k-means clustering results for 2 clusters.



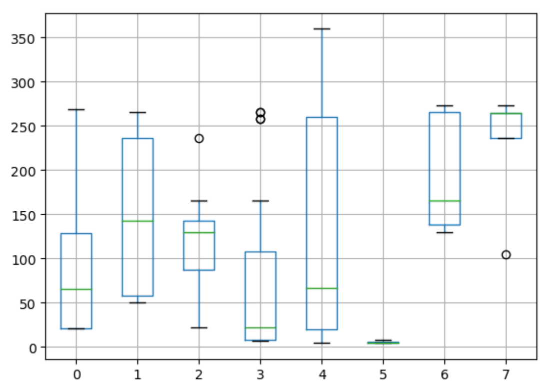
(b) The toughness distribution of the HOG k-means clustering results for 2 clusters.

Figure 4.12: Two toughness distributions with a different number of clusters, using HOG and k-means.

extraction algorithm can thus find a set of clusters better separated than other clustering techniques. Using k-means clustering however, it does not succeed as well as GLCM into assigning the metal samples into meaningful clusters. The clustering results using two and eight clusters, are shown in Figure 4.13. These results show again the same phenomenon as in other feature extraction techniques, namely one clusters contains a very wide distribution of samples. However the other clusters show to be quite narrow and meaningful. LBP can suffer from the same problems as HOG, making the feature vector contain more noise due to the JPEG compression.



(a) The toughness distribution of the LBP k-means clustering results for 2 clusters.



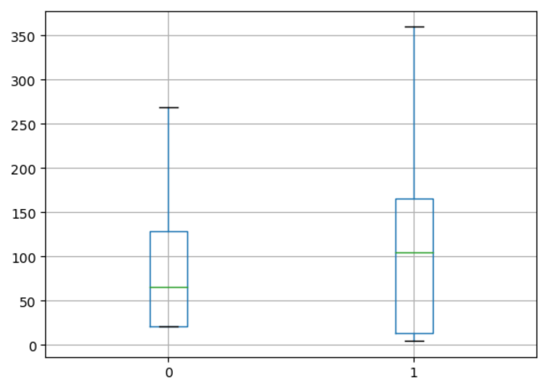
(b) The toughness distribution of the LBP k-means clustering results for 8 clusters.

Figure 4.13: Two toughness distributions with a different number of clusters, using LBP and k-means.

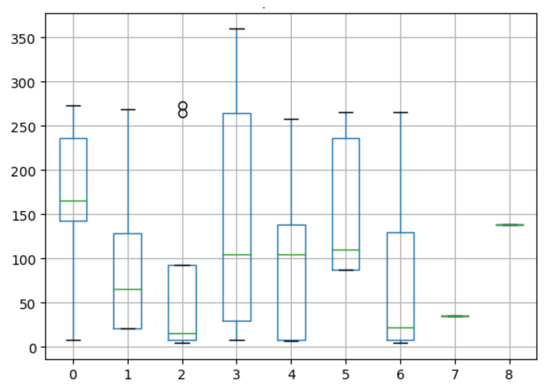
The k-means clustering of LBP vectors often resulted in a some large clusters and the other clusters would contain a very small amount of patches. The LBP clustering algorithms can be concluded to be quite successful, but not the best

The final feature extraction algorithm, using the same pipeline architecture as the previously discussed techniques, is the Discrete Wavelet Transform. In this technique we use again first order statistics, but first a DWT was performed on the image patch. This removes a lot of variance of lighting conditions and highlights edges in the image. DWT achieves together with LBP the best DB-index, and performs very good for the two lowest image magnification levels. The classification results of the DWT-pipeline are shown in Figure 4.14. In Figure 4.14b, is shown that k-means can also infer meaningful clusters from these feature vectors. Only cluster 4 is a miscellaneous cluster, where a very broad variety of metal samples is assigned to. Compared to the results of the GLCM feature extraction, the DWT-pipeline has a less granular classification with a few clusters indicating a more ductile material others indicate more brittle material. There are no distinct classes in between, as in Figure 4.11b. DWT reduces the variability present in FOS, but it still extracts edges caused by compression artifacts in the images.

The pipeline for the Weyl transform is as previously described, different from the previous pipelines. In this pipeline, instead of taking the most frequent sample to classify an image, the feature vectors of the image patches are averaged before the clustering is performed. The reason for this is twofold: Firstly, the Weyl Transform currently only supports (16×16) patches. In these small patches alone there is not enough information available to infer details about the microstructure of the metal sample and subsequently about the toughness property. The Weyl Transform works very well when averaging overlapping patches, so for this reason the coefficients

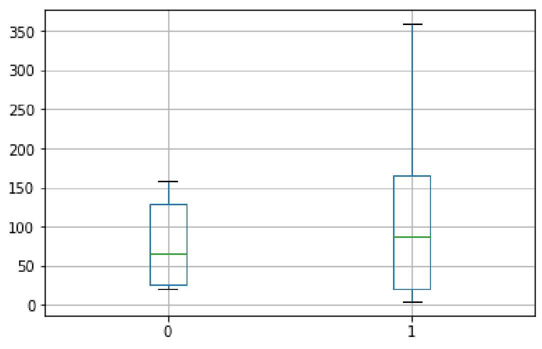


(a) The toughness distribution of the DWT k-means clustering results for 2 clusters.

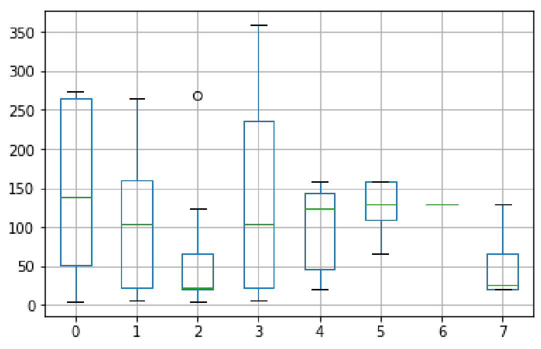


(b) The toughness distribution of the DWT k-means clustering results for 9 clusters.

Figure 4.14: Two toughness distributions with a different number of clusters, using DWT and k-means.



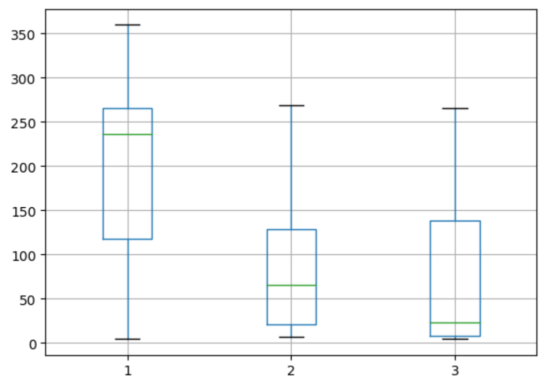
(a) The results for 2 clusters.



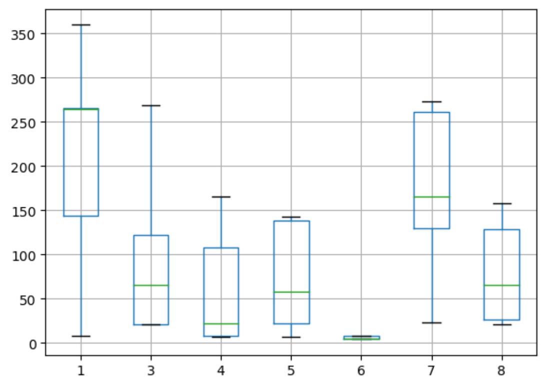
(b) The results for 8 clusters.

Figure 4.15: Two toughness distributions with a different number of clusters, using the Weyl Transform and k-means.

of these overlapping image patches are calculated and then averaged before clustering. The consequences of changing the pipeline in this manner means that there are fewer data points available to train the clustering algorithms, which may make it more difficult to find a good meaningful clustering and may require more data to train more complex models. The resulting toughness distributions of the clusters for two pipelines with 2 and 8 clusters respectively is shown in Figure 4.15. The results for two clusters are very similar to other feature extraction techniques, but the results for 8 clusters show a more granular classification, similar to the results of the GLCM feature extraction. These are a bit unexpected as the Weyl Transform can only take the autocorrelations of very small patches into account and just averages the features.



(a) The toughness distribution of the DWT-LBP k-means clustering results for 2 clusters.



(b) The toughness distribution of the DWT-LBP k-means clustering results for 9 clusters.

Figure 4.16: Two toughness distributions with a different number of clusters, using DWT-LBP and k-means.

As a final experimental technique, we first calculated the DWT and afterwards LBP on the one of the resulting filtered images. The reason for this idea comes from the problems seen in using LBP on its own and the improvement DWT brings with very simple metrics on these filtered images. DWT helps to remove noise in the image that is problematic for LBP. An example of an image patch in the HH quadrant is shown in Figure 4.5. In the previous data set this was shown to have a big improvement on the clustering evaluation, achieving the lowest DB-index for the k-means algorithm. Clustering results of this pipeline are shown in Figure 4.16. These results are quite promising. Using these clusters with 9 clusters. We can divide the data into three separated classes based on their toughness.

4.3.2 Hierarchical clustering and Sparse Subspace Clustering

In this section, we will discuss the results obtained from hierarchical clustering and sparse subspace clustering algorithms. These two methods differ significantly from k-means clustering in their underlying principles. k-means clustering follows a top-down iterative approach to optimize cluster centers and assign data points based on their Euclidean distance. On the other hand, hierarchical clustering is a bottom-up approach. It starts with each data point being treated as its own cluster and then gradually merges the most similar clusters until the desired number of clusters is reached or a distance threshold between clusters is exceeded. Sparse Subspace Clustering (SSC), like k-means, is a top-down algorithm. However, it does not employ a distance metric to cluster data points. Instead, it solves a sparse optimization problem to determine an optimal representation for each data point. This representation can then be used

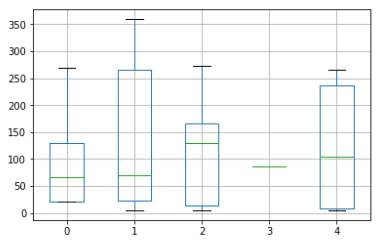
Feature extraction	**K-means**	**Hierarchical**	**Sparse subspace clustering**
FOS	0.19	0.19	1.93
GLCM	0.48	0.48	0.56
HOG	3.58	4.34	1.60
LBP	0.18	0.19	0.19
DWT	**0.16**	**0.16**	1.56
Weyl	0.72	0.87	1.18

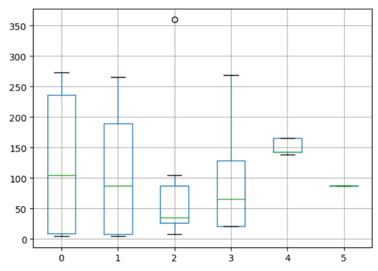
Table 4.9: The Davies-Bouldin index of the different clustering algorithms using the second, labeled, data set, with an image magnification level of 500×.

to construct a similarity graph, which is utilized for clustering. The significant advantage of SSC is its ability to capture complex relationships among data points. However, one drawback is that the results may be less stable, with a few clusters dominating over others. Another downside is the substantial increase in computational complexity associated with both hierarchical clustering and sparse subspace clustering. These algorithms take considerably longer to run compared to their k-means counterpart, especially when dealing with feature vectors of larger sizes.

These algorithms will be compared to each other and the k-means baseline of each feature extractor. Just like in the previous data set, we collected all the best DB-indices for each clustering algorithm. The results are summarized in Table 4.9. The table shows that the cluster quality evaluation of hierarchical and k-means clustering mostly have very similar metrics and that sparse subspace clustering performs worse in general. However, it must be noted that previous results have already shown that the a the Davies-Bouldin index does not directly correlate with the quality of the toughness distributions. In the following paragraphs we will compare the image toughness distributions per cluster for hierarchical and sparse subspace clustering to the baseline of each feature extraction technique, k-means. The goal is to compare if the more advanced clustering algorithms can better infer information about the metallurgical properties of a metal sample from a feature vector resulting from the feature extraction technique and which clustering algorithm performs best in combination with each feature extraction technique.

FOS is once again considered as the baseline algorithm since it is unable to capture any geometric features from the image. When using k-means, FOS demonstrated a good DB-evaluation. However, the resulting toughness classification was not meaningful due to the clusters having very similar distributions. The best results from k-means, hierarchical, and SSC clustering algorithms are depicted in Figure 4.17. As expected, the more complex clustering algorithms did not improve the distributions of the resulting clusters. Therefore, it can be concluded that FOS is incapable of extracting meaningful information about the toughness property from microscopic images. The extracted features lack sufficient information and are heavily influenced by lighting variations between images, which have no correlation with the metallurgical properties.


(a) K-means clustering


(b) Hierarchical clustering

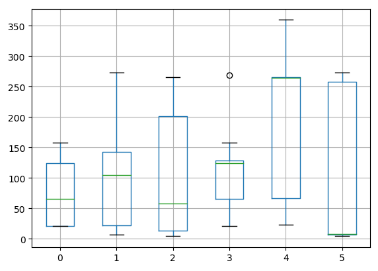
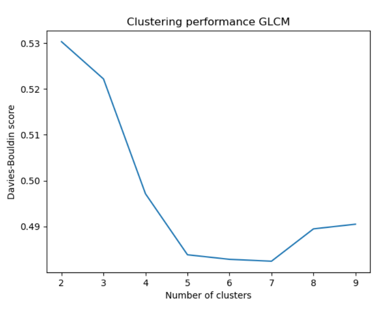

(c) Sparse subspace clustering.

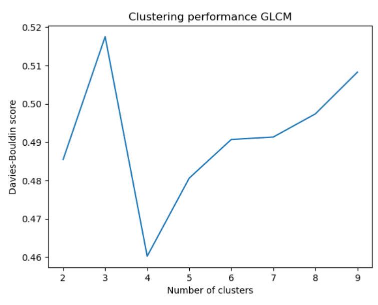
Figure 4.17: The relation between the number of clusters and the Davies-Bouldin index, when using different clustering algorithms on the FOS feature vectors.

GLCM was the best performing feature extraction when only k-means is taken into consideration. It also showed very good Davies-Bouldin evaluations for a higher number of clusters. A good clustering performance when using more clusters is desirable as it means that when the clustering gets more fine, the separation distance of the clusters does not suffer. This is not the case anymore for when using these other algorithms. The Davies-Bouldin index of hierarchical clustering increases dramatically after using more than 5 clusters and SSC has one low Davies-Bouldin index, namely for three clusters. The relation between the number of clusters and the Davies-Bouldin index is shown in Figure 4.18. For k-means we can see that the cluster quality improves steadily until 7 clusters where the best DB-index is achieved. For the other clustering algorithms the opposite is true. The graph shows an early minimum followed by a steady increase. For the actual distributions of the image toughness, the best results for each clustering algorithm are shown in Figure 4.19. In the results of the SSC clusters, there are only three clusters present, the cause of the small number of clusters is that the SSC algorithm, only adds very small clusters when more clusters are specified. When in the final step of the process, the most frequent cluster labels are selected, these clusters do not show up in the final results. From the Davies-Bouldin optimization and the final distributions of the clusters we can conclude that for GLCM, the k-means clustering algorithm performs the best, the hierarchical algorithm shows some meaningful results and SSC is not applicable in practice together with GLCM.

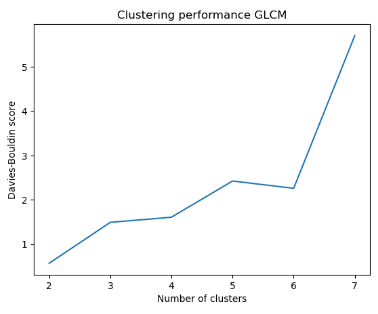
The HOG pipeline was able to distinguish the differences is toughness between metal samples quite good, considering the significantly worse cluster quality evaluation in comparison with other feature extractors. One of the possible reasons we found for HOG, having noisy and thus less informative feature vectors, are the compression artifacts from the JPEG compression algorithm. In figure 4.20. The k-means and sparse subspace clustering algorithms are able to find some interesting classification for the metal samples, but the hierarchical algorithm is able to split the samples quite nicely. only two clearly separated groups can be taken from the boxplot, but there really is a minimal overlap between these groups, making the clustering results have



(a) K-means clustering

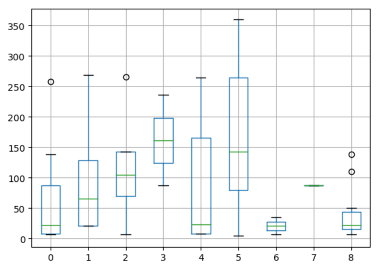


(b) Hierarchical clustering

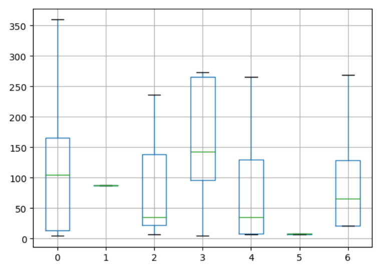


(c) Sparse subspace clustering.

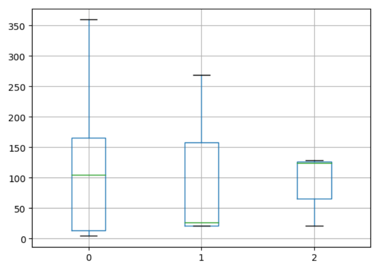
Figure 4.18: The relation between the number of clusters and the Davies-Bouldin index, achieved by the different clustering algorithms on the GLCM feature vectors.



(a) K-means with 9 clusters.



(b) Hierarchical with 7 clusters.

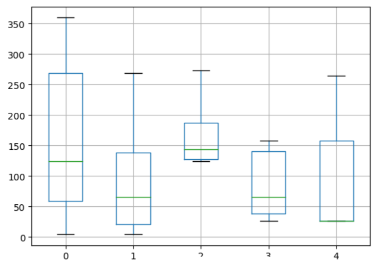


(c) SSC with 3 clusters.

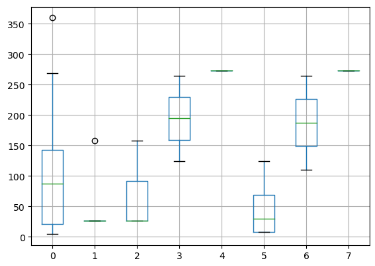
Figure 4.19: A comparison of the best clustering results of k-means, hierarchical and sparse subspace clustering using the GLCM features.

a strong correlation with the toughness properties of the samples. These results contradict the earliest predictions from Table 4.9, because there Hierarchical clearly has the worst cluster quality and SSC has the lowest Davies-Bouldin index, by a clear margin. This shows again that a result of compact and clearly separated clusters has no guarantee to correlate strongly with the toughness properties of the images. HOG clearly looks at the amount of gradients present in each image patch, capturing crucial microstructural information to base the clustering on. Considering these results it may be interesting to try this feature extraction algorithm on the raw image formats without the problematic compression artifacts.

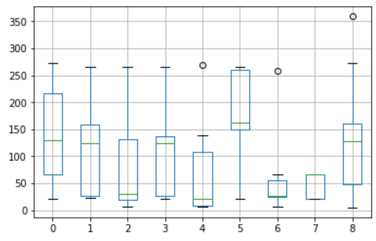
LBP is always able to extract feature vectors that lead to the most compact an separated clusters. In every previously shown result, LBP has produced one of the best Davies-Bouldin scores. The toughness distributions per cluster, however, showed some separation, but not as clearly separated or narrowly distributed as other feature extraction methods. The best results for each clustering algorithm are shown in Figure 4.21. In this case sparse subspace clustering is not really able to improve the results given by the k)means algorithm. There is some separation, but the cluster classification is not really informative. The Hierarchical algorithm shows a similar



(a) K-means with 5 clusters.

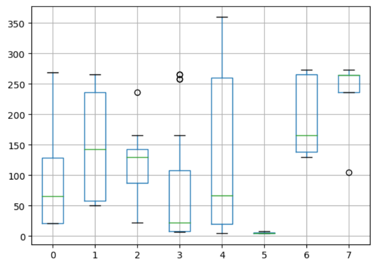


(b) Hierarchical with 8 clusters.

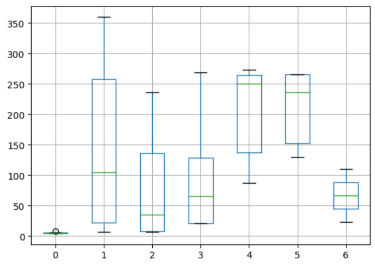


(c) SSC with 9 clusters.

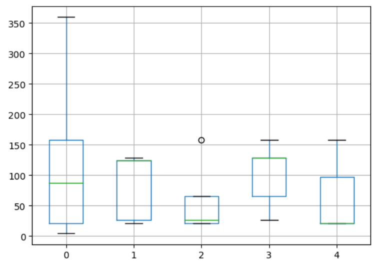
Figure 4.20: A comparison of the best clustering results of k-means, hierarchical and sparse subspace clustering using the HOG features.



(a) K-means with 8 clusters.



(b) Hierarchical with 7 clusters.

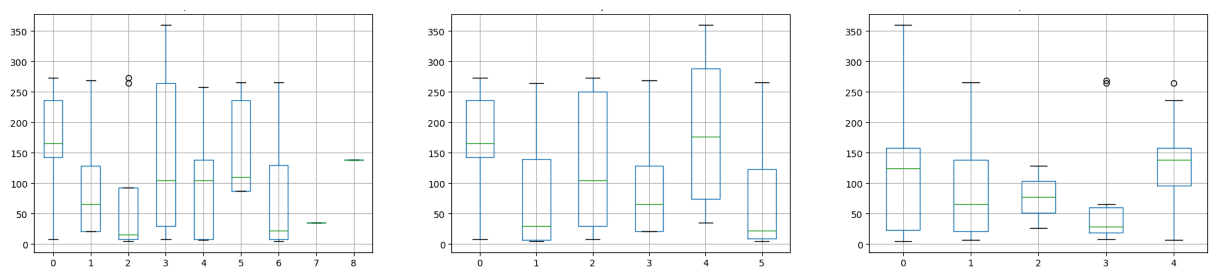


(c) SSC with 5 clusters.

Figure 4.21: A comparison of the best clustering results of k-means, hierarchical and sparse subspace clustering using the LBP features.

performance where we can distinguish one general cluster and the other clusters combined can be separated into three classes of toughness. We can conclude the LBP is not the best suited feature extraction technique for clustering based on toughness.

The DWT feature extraction attains the lowest Davies-Bouldin scores for the second data set for both k-means clustering and hierarchical clustering. The sparse subspace clustering results are, however, subpar compared to most other feature extractions. The best toughness distributions per cluster for all three clustering algorithms are displayed in Figure 4.22. The results of k-means showed already some separation in toughness distributions, but cluster 2 contained both very brittle and very ductile samples, meaning that a classification into this cluster gave no real meaningful information about the toughness property of the sample. When clustering with the hierarchical and sparse subspace algorithms this problem did not disappear, although most other clusters did show separated distributions. The problem with the DWT feature vectors is that after the discrete wavelet transform is calculated, the dimensions are further reduced by taking first order statistics of the results. To improve on this problem in the next paragraph we put another transformation after the DWT, namely LBP. This way the discrete wavelet transform



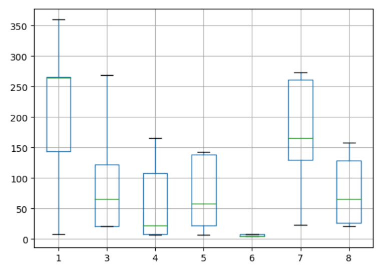
(a) K-means with 9 clusters. (b) Hierarchical with 6 clusters. (c) SSC with 6 clusters.

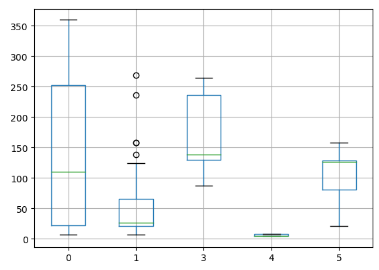
Figure 4.22: A comparison of the best clustering results of k-means, hierarchical and sparse subspace clustering using the DWT features.

can reduce noise and dimensions in the image and LBP can then extract the feature from the improved image patch. In this manner the results could improve greatly.

In the k-means experiment, the fusion of the discrete wavelet transform with local binary patterns has demonstrated itself as a promising technique for grouping samples according to their toughness property. The primary objective behind combining these two methods is to initially diminish the noise caused by compression artifacts and other variables. Figure 4.23 illustrates the resulting classifications obtained for the all clustering algorithms. the k-means algorithm is capable to separate the samples into three categories based on their toughness. Two classes can be defined, but cluster one is not really meaningful. Hierarchical clustering with six clusters, however, shows some very nice clusters. Only five clusters show up in the final result, but apart from the first one, which contains all sorts of samples, each cluster can be seen as its own class, with a very distinct separation. The SSC results showed again quite nice results, but not as good as the hierarchical version. Two classes can be identified based on the toughness property here. The reason that this works quite will is due to the fact that the discrete wavelet transform reduces the noise introduced by compression, but still retains all other information, useful for the LBP algorithm. We can conclude that the combination of DWT with LBP is an improvement on both LBP and DWT and that it is best combined with the k-means clustering algorithm.

The last feature extraction method used in this work, is the Weyl Transform. The Weyl Transform is based on a mapping of multiscale autocorrelations in a signal. To achieve this, the image patch is stacked into a vector to create the information vector on which the transform is performed. Using as a feature extraction technique, the cluster evaluation via the Davies-Bouldin index is average. The k-means toughness distribution, however, was better than most other feature extractors. All the best classification results are shown in Figure 4.24. From these images we can conclude that the k-means produces the best results and that hierarchical and sparse subspace clustering do not provide a good separation of toughness values between clusters. The k-means clustering already shows some decent results being able to separate between the lower


(a) K-means with 9 clusters.


(b) Hierarchical with 6 clusters.

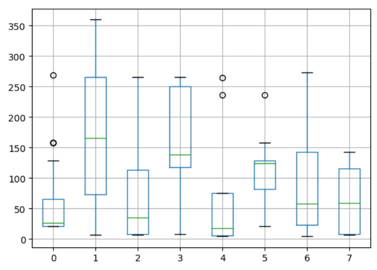
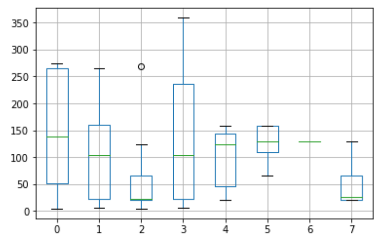
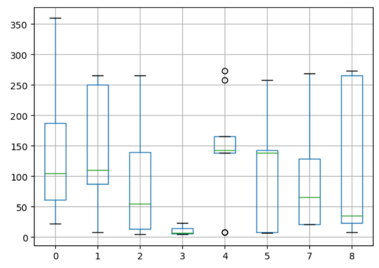

(c) SSC with 8 clusters.

Figure 4.23: A comparison of the best clustering results of k-means, hierarchical and sparse subspace clustering by first applying DWT and then LBP on the image patches.


(a) K-means with 8 clusters.


(b) Hierarchical with 9 clusters.

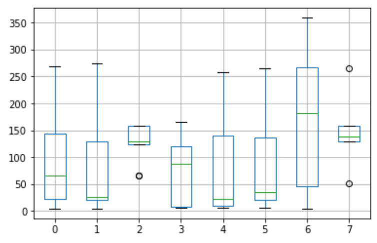

(c) SSC with 8 clusters.

Figure 4.24: A comparison of the best clustering results of k-means, hierarchical and sparse subspace clustering using the Weyl features.

half and upper half of toughness samples. Except for one cluster that contains an average in the lower half, but also some of the highest toughness values. In the hierarchical clustering results, the distributions of the clusters is much more narrow and it is even possible to classify these results into three to four classes. The clear separation between clusters in this example is the best so far. The downside of this result is that the clustering was performed using 8 clusters, but only six of them show up in the final image classifications. This means that we cannot know what the remaining two clusters would contain if more images were available and some would be classified into that cluster. The sparse subspace clustering results only divided the dataset in more or less two categories, clusters 2, 6 and 7 contain the more ductile materials, while the other clusters contain the more brittle materials. For the Weyl Transform we can thus conclude that the k-means algorithm produces the best results with 3 fairly well separated classes of clusters, although one cluster contains a more widely spread contains both brittle and ductile samples.

4.4 Conclusion

During the experimental phase, our initial focus was to assess the different models using clustering evaluation metrics exclusively. The primary objective of this phase was to fine-tune the parameters of the feature extraction techniques and clustering methods based on cluster quality. To compare the resulting clusters from various combinations, we employed the Davies-Bouldin index as our evaluation metric. From this first phase of experimentation, we reached the conclusion that overall, k-means clustering exhibited the best performance. However, it is worth noting that for certain feature extraction algorithms, hierarchical clustering showed a slight improvement. Sparse subspace clustering, on the other hand, did not outperform the k-means clusters in any scenario. Among the feature extraction techniques, the Local Binary Patterns (LBP) approach achieved the lowest Davies-Bouldin index, with the Gray-Level Co-occurrence Matrix (GLCM) technique coming in a close second. However, it is important to highlight that GLCM showcased a significant advantage in terms of generating superior cluster quality when a higher number of clusters was employed.

In the second part of the experiments, we compared the resulting distributions relating to the toughness property of the samples. In this part a second data set was used that contained the specific absorbed energy measured in each Charpy test which is the toughness property of the sample. Due to the fact that toughness is related to the temperature of testing, this data set contained a lot of samples that were very similar, but tested on another temperature. This caused the Davies-Bouldin evaluations for most of the algorithms to be lowered by a serious amount. The observations from the Davies-Bouldin scores in Table 4.8, is that most of the algorithms that previously outperformed FOS, based on the cluster evaluation, still do. Only GLCM produced very similar results from the first data set lowering its Davies-Bouldin score from 0.50 to 0.48. The Davies-Bouldin score does not evaluate the link between the samples in a cluster and their respective toughness properties. It does give an indication on how well the model is able to separate the patches into classes, with a better performance relating to a more stable and reliable model.

The final analysis of the toughness distributions for each model is still the most important part. In this part of the experiments we solve 2 questions. First, how many classes of samples with related toughness properties can be naturally found in the data set. Second, which feature extractor and clustering algorithm show the best capability to recognize the toughness property of a material and can group them well together. Before we give a conclusion on the results, the first observation made, was that a higher number of clusters resulted in clusters containing more similar samples, but some some of these clusters then represented a similar group of samples. We could thus conclude that more clusters did not necessarily find more classes, but the separation of classes was improved. The clustering results of GLCM with 9 clusters showed a very nice separation of clusters. Only two clusters had quite a wide spread of samples, but the mean samples showed that these clusters were still quite informative on the toughness property of the samples. Secondly, the Weyl Transform combined with k-means, produces very similar results. When comparing the distributions we can distinguish three fairly well separated classes. The final clustering algorithm we would like to discuss is a combination of DWT and the LBP feature extraction algorithm. This combination was, when clusters with k-means, able to separate the data into three distinct well separated classes and median of the first cluster is taken into account, a case could be made for four clusters. The reason this feature extractor is able to improve on the base LBP algorithm is due to the fact that DWT removes noise without removing information information that is necessary for the LBP feature extractor.

In summary, the recommended approach for analyzing the images is as follows:

- Utilize images captured at a magnification level of $500\times$
- Crop the images to remove the added textual data and divide the images into patches of size $(64, 64)$.
- Apply the Gray-Level Co-occurrence Matrix (GLCM) feature extraction technique on the patches
- Finally, cluster the resulting feature vectors using the k-means algorithm with 9 clusters.

The GLCM model demonstrates a high level of stability, as indicated by its low Davies-Bouldin index. Additionally, it has the capability to differentiate between three to four classes of materials with distinct toughness properties. By following this approach, the most reliable results can be expected for material classification and characterization based on the analyzed images.

5 Conclusion and Future Work

Pattern recognition is not a new subject and there are countless publications evaluating and improving recognition techniques or applying the techniques to a new subject. Even in metallurgical research, the chemical composition of metal samples can already be accurately classified using pattern recognition techniques on microscopic images [10, 11]. However, there is, to the best of our knowledge no research yet that tries to evaluate the metallurgical properties of metal samples via pattern recognition techniques, without using the knowledge of chemical composition or other classification results. Defining a relation between microstructure and metallurgical properties can reduce the financial an time costs for development of new custom materials, as it allows the developer to make more informed decisions and obtain a better understanding about the relations between the processing steps and the final properties of the developed material.

In this thesis, we conducted an unsupervised analysis on microscopic images obtained from metallurgical research. The decision to employ unsupervised analysis was motivated by the characteristics of the datasets used. The first dataset did not contain any information regarding the metallurgical properties of the samples, while the second dataset included the absorbed energy values obtained from Charpy tests conducted on each sample, which are directly related to the toughness property of the material. Dividing these energy values into distinct classes was not straightforward, and the objective set by OCAS was to identify inherent classes within the data, rather than training an algorithm to distinguish between predefined classes.

The clustering process consisted of multiple steps. The images were first cropped to remove any textual data. Then the image were divided into patches of shape $(64, 64)$. These patches were subject to a transformation using different feature extraction techniques. The aim of transforming the image patches was to represent them in a low-dimensional feature vector that contained all information regarding the microstructure to enable the clustering algorithms to find the necessary similarities regarding the toughness property of the metal samples. The clustering method can then group these patches together in clusters based on their toughness. The results could then be evaluated using the Davies-Bouldin index. These evaluation allowed to analyze and select the optimal amount of clusters for each clustering and feature selection algorithm to achieve the best possible cluster results.

For the second data set there were a few additional steps. For each image, the most frequent cluster label assigned to its patches, was considered to be the cluster of that image. Using the cluster labels for each image it was possible to evaluate the distribution of the toughness values assigned to a cluster. With boxplot the resulting distributions could be compared and the best model could be chosen.

In conclusion the best model to use is the combination of the GLCM feature extractor with k-means clustering for 9 clusters. In Figure 4.18a, the resulting cluster distributions are presented.

Finally, we propose some future work that can lead to improvements on this subject:

- To develop a model for toughness classification, supervised learning may provide dramatically better results than unsupervised as the model will learn to use certain features for clustering classification. The disadvantage is that the classes have to be specified instead of learned from the data, but the final results may be more based on the toughness property.
- The labeling process of all the images may be difficult and time consuming. To keep the advantages of supervised learning while reducing the required amount of labels, semi supervised machine learning techniques can be utilized.
- In this work we used images that were severely compressed using the JPEG algorithm. This resulted in different compression artifacts being present in the images. These artifacts introduce noise that will have an influence on the final clustering results. In future it may be beneficial to re evaluate the different techniques presented in this work on uncompressed formats such as ppm or png.
- There are other more advanced clustering algorithms that were not included in this thesis. These may provide better and or more stable results, but this is no guarantee as in this work the most simple algorithm often performed the best.

In summary this was the first step in the direction of deriving metallurgical properties from fracture surfaces using artificial intelligence. The techniques in this work still allow for optimizations, but they show that AI techniques can successfully be implemented in metallurgical research as a tool for toughness analysis of the metal samples.

Bibliography

[1] M. Radetzki, “Seven thousand years in the service of humanity—the history of copper, the red metal,” *Resources Policy*, vol. 34, no. 4, pp. 176–184, 2009.

[2] R. K. Sodhi, S. Paul, *et al.*, “Metal complexes in medicine an overview and update from drug design perspective,” *Cancer Therapy & Oncology International Journal*, vol. 14, no. 1, pp. 25–32, 2019.

[3] R. Smith, G. Lewi, and D. Yates, “Development and application of nickel alloys in aerospace engineering,” *Aircraft engineering and aerospace technology*, vol. 73, no. 2, pp. 138–147, 2001.

[4] C. Buchanan and L. Gardner, “Metal 3d printing in construction: A review of methods, research, applications, opportunities and challenges,” *Engineering Structures*, vol. 180, pp. 332–348, 2019.

[5] W. D. Callister, D. G. Rethwisch, *et al.*, *Materials science and engineering: an introduction.* Wiley New York, 2018, vol. 9.

[6] M. Odendaal, F. Vermaak, and E. du Toit, “Cost estimation and management over the life cycle of metallurgical research projects,” *Southern African Business Review*, vol. 19, Jan. 2015.

[7] A. I. Zaitsev, “Prospective directions for development of metallurgy and materials science of steel,” *Pure and Applied Chemistry*, vol. 89, no. 10, pp. 1553–1565, 2017.

[8] M. E. Launey and R. O. Ritchie, “On the fracture toughness of advanced materials,” *Advanced Materials*, vol. 21, no. 20, pp. 2103–2110, 2009.

[9] J. An, J. She, H. Chen, and M. Wu, “Applications of evolutionary computation and artificial intelligence in metallurgical industry,” *Evolutionary Computing and Artificial Intelligence: Essays Dedicated to Takao Terano on the Occasion of His Retirement 2*, pp. 77–87, 2019.

[10] M. Larmuseau, M. Sluydts, K. Theuwissen, L. Duprez, T. Dhaene, and S. Cottenier, “Compact representations of microstructure images using triplet networks,” *npj Computational Materials*, vol. 6, no. 1, p. 156, 2020.

[11] M. Larmuseau, M. Sluydts, K. Theuwissen, L. Duprez, T. Dhaene, and S. Cottenier, "Race against the machine: Can deep learning recognize microstructures as well as the trained human eye?" *Scripta Materialia*, vol. 193, pp. 33–37, 2021.

[12] S. Zhang, D. Sun, Y. Fu, and H. Du, "Toughness measurement of thin films: A critical review," *Surface and Coatings Technology*, vol. 198, no. 1, pp. 74–84, 2005.

[13] R. Moskovic and P. Flewitt, "An overview of the principles of modeling charpy impact energy data using statistical analyses," *Metallurgical and Materials Transactions A*, vol. 28, no. 12, pp. 2609–2623, 1997.

[14] R. Shant, T. Kyada, R. Goyal, and T. S. Kathayat, "Understanding the delamination and its effect on charpy impact energy in thick wall linepipe steel," *Journal of Materials and Metallurgical Engineering*, vol. 4, pp. 31–39, Jan. 2014.

[15] A. Hasnaoui, H. Van Swygenhoven, and P. Derlet, "Dimples on nanocrystalline fracture surfaces as evidence for shear plane formation," *Science*, vol. 300, no. 5625, pp. 1550–1552, 2003.

[16] B. J. Inkson, "Scanning electron microscopy (sem) and transmission electron microscopy (tem) for materials characterization," in *Materials characterization using nondestructive evaluation (NDE) methods*, Elsevier, 2016, pp. 17–43.

[17] M. W. Marcellin, M. J. Gormish, A. Bilgin, and M. P. Boliek, "An overview of jpeg-2000," in *Proceedings DCC 2000. Data Compression Conference*, IEEE, 2000, pp. 523–541.

[18] D. Ping Tian *et al.*, "A review on image feature extraction and representation techniques," *International Journal of Multimedia and Ubiquitous Engineering*, vol. 8, no. 4, pp. 385–396, 2013.

[19] C. Tomasi, "Histograms of oriented gradients," *Computer Vision Sampler*, pp. 1–6, 2012.

[20] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and the scikit-image contributors, "Scikit-image: Image processing in Python," *PeerJ*, vol. 2, e453, Jun. 2014.

[21] G. Othman and D. Q. Zeebaree, "The applications of discrete wavelet transform in image processing: A review," *Journal of Soft Computing and Data Mining*, vol. 1, no. 2, pp. 31–43, 2020.

[22] G. Lee, R. Gommers, F. Waselewski, K. Wohlfahrt, and A. O'Leary, "Pywavelets: A python package for wavelet analysis," *Journal of Open Source Software*, vol. 4, no. 36, p. 1237, 2019.

[23] D. Zhang and D. Zhang, "Wavelet transform," *Fundamentals of Image Data Mining: Analysis, Features, Classification and Retrieval*, pp. 35–44, 2019.

[24] P. Bhagat, P. Choudhary, and K. M. Singh, "Chapter 13 - a comparative study for brain tumor detection in mri images using texture features," in *Sensors for Health Monitoring*, ser. Advances in ubiquitous sensing applications for healthcare, N. Dey, J. Chaki, and R. Kumar, Eds., vol. 5, Academic Press, 2019, pp. 259–287.

[25] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, no. 6, pp. 610–621, 1973.

[26] Y. Qu, J. Hao, and R. Peng, "Machine-learning models for analyzing tsom images of nanostructures," *Optics Express*, vol. 27, p. 33 978, Nov. 2019.

[27] Z. Guo, L. Zhang, and D. Zhang, "A completed modeling of local binary pattern operator for texture classification," *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1657–1663, 2010.

[28] Q. Qiu, A. Thompson, R. Calderbank, and G. Sapiro, "Data representation using the weyl transform," *IEEE Transactions on Signal Processing*, vol. 64, no. 7, pp. 1844–1853, 2016.

[29] T. Zhao, S. Lazendić, Y. Zhao, G. Montereale-Gavazzi, and A. Pižurica, "Classification of multibeam sonar image using the weyl transform," in *Image Processing and Communications: Techniques, Algorithms and Applications*, Springer, 2019, pp. 206–213.

[30] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[31] Q. V. Le *et al.*, "A tutorial on deep learning part 1: Nonlinear classifiers and the backpropagation algorithm," *Mountain View, CA*, 2015.

[32] P. Kim and P. Kim, "Convolutional neural network," *MATLAB deep learning: with machine learning, neural networks and artificial intelligence*, pp. 121–147, 2017.

[33] D. Bank, N. Koenigstein, and R. Giryes, *Autoencoders*, 2021.

[34] S. Ray, "A quick review of machine learning algorithms," in *2019 International conference on machine learning, big data, cloud and parallel computing (COMITCon)*, IEEE, 2019, pp. 35–39.

[35] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[36] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, pp. 395–416, 2007.

[37] T. Stegen and A. p. (Pižurica, *Clusteren van hyperspectrale beelden met behulp van deep learning*, und, 2022.

[38] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 11, pp. 2765–2781, 2013.

[39] C. You, C.-G. Li, D. P. Robinson, and R. Vidal, "Oracle based active set algorithm for scalable elastic net subspace clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3928–3937.

[40] Huang, Shaoguang and Zhang, Hongyan and Du, Qian and Pizurica, Aleksandra, "Sketch-based subspace clustering of hyperspectral images," eng, *REMOTE SENSING*, vol. 12, no. 5, p. 32, 2020.

[41] F. Nielsen and F. Nielsen, "Hierarchical clustering," *Introduction to HPC with MPI for Data Science*, pp. 195–211, 2016.

[42] K. R. Shahapure and C. Nicholas, "Cluster quality analysis using silhouette score," in *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, 2020, pp. 747–748.

[43] M. Mughnyanti, S. Efendi, and M. Zarlis, "Analysis of determining centroid clustering x-means algorithm with davies-bouldin index evaluation," in *IOP Conference Series: Materials Science and Engineering*, IOP Publishing, vol. 725, 2020, p. 012 128.

[44] S. V. Petrovic, "A comparison between the silhouette index and the davies-bouldin index in labelling ids clusters," 2006.

[45] G. C. Linderman, M. Rachh, J. G. Hoskins, S. Steinerberger, and Y. Kluger, "Efficient algorithms for t-distributed stochastic neighborhood embedding," *arXiv preprint arXiv:1712.09005*, 2017.

[46] R. Bro and A. K. Smilde, "Principal component analysis," *Analytical methods*, vol. 6, no. 9, pp. 2812–2831, 2014.

FACULTY OF ENGINEERING
AND ARCHITECTURE

Uncovering Patterns in Microscopic Images from Metallurgical Research with Unsupervised Learning Algorithms

Henri Vandendorpe
Student number: 01808971

Supervisor: Prof. dr. ir. Aleksandra Pizurica
Counsellors: Srdan Lazendic, Ir. Lode Duprez (OCAS - Arcelormittal Global R&amp;D Gent)

Master's dissertation submitted in order to obtain the academic degree of
Master of Science in Computer Science Engineering

Academic year 2022-2023

GHENT
UNIVERSITY