Identifying recrystallization in stainless steel using machine learning on acoustic data

Kaourintin Tamine, Adrien Fermeli-Furic, Sophia Artioli
Department of Computer Science, EPF Lausanne, Switzerland

***Abstract*—Laser Powder Bed Fusion (LPBF) of metallic alloy is a new technique to manufacture metal in 3D structures with immense design freedom and flexibility compared to current conventional industry techniques. Unfortunately, it is very hard to have consistent repeatability from the experiments because of a lack of proper quality monitoring. In particular, to check for specific micro-structure transformations such as recrystallization in stainless steel, high resolution techniques such as Synchrotron X-Ray diffraction are used. This work's primary goal is to find alternate solutions that are faster and cheaper while still providing excellent precision on the prediction of whether or not recrystallization has successfully taken place. Three different models are used on acoustic data with varying amounts of success with the best ones having over 90% accuracy.**

I. Introduction

Claire Navarre is a doctoral assistant at the LMTM (Laboratory of Thermo-mechanical Metallurgy) and uses laser-induced heat treatments to induce recrystallization in highly deformed stainless steel plates in order to correct faults in their micro-structure. Recrystallization lowers the strength but improves the ductility and may lead to a refined grain size. The current state of the art to check for the occurrence of recrystallization is mainly based on electron back-scattered diffraction which is complicated and unpractical. If we consider this problem from a machine learning standpoint, there are only two possible states of the alloy: it can be seen as a binary classification problem which implies that if a sufficient data set of features is obtained, we could use machine learning techniques to solve the problem and make it faster, cheaper and more convenient to solve. As laser powder bed fusion is a very noisy process the idea is that these features can be extracted from acoustic emissions. Acoustic sensors are inexpensive, easy to install for an experiment, and the extracted data can be processed by a single computer to predict if recrystallization occurred or not. There is also the added benefit that the implemented ML pipeline used to assert the recrystallization state could work on the fly during experiments instead of post-experiment, making it possible to have live feedback and re-apply the laser on areas where the wanted phenomenon did not occur. The idea of using sound comes from previous works [She+18], [Pan+21] where ML has been successfully applied to solve similar problems in the field of LPBF.

II. Current state of the art

There have been multiple techniques used to solve these similar sound-based issues in the literature. Prior to sound-based techniques, video analysis with IR has been used successfully [Mas+21], but requires expensive equipment and is unpractical to set up. Sound analysis through standard neural net-based implementations was used to identify sintering or melting events, or through more signal-based neural net architectures called convolutional neural networks [She+18]. There have also been attempts at more flexible models that need less guided training inputs [Pan+21]. These recent works have obtained accuracy in identification ranging from 80% to 92% on similar classification problems in the LPBF field.

III. Data analysis, pre-processing and exploration

A. Acquiring the data

To obtain data for training, dislocations in stainless steel samples are introduced by cold rolling (pressing the sheets). The experiments consist in laser-induced heat treatments on two different states of the stainless steel plates: on one hand, the heat treatment is performed directly on the rolled sample (recrystallization occurs), on the other hand the heat treatment is performed on the rolled sample that was previously recrystallized (recrystallization does not occur). During the experiments, the same acoustic measuring devices are placed at the same distance to maximize similarity and make the models predict better.

B. Issue: Information loss through re-sampling

The experiment sampling acoustic sensors are sampled at a frequency of 3Mhz. In sound-based machine learning most data sets are sampled at 16 Khz. To accommodate for this, we extend the total time of extracted sound from experiments by declaring a 16kHz sample when generating .wav files. A ten seconds experiment becomes a thirty minutes audio because of change in the declared sampling frequency but since no information is lost it does not matter.

C. Issue : microphone distance

Referring to the experiment set-up of Figure 1, the microphone distance to the laser more than doubles between the beginning and the end of the experiment, going from 37mm to 87mm. It might be better for model training to give the distance as a feature, or use the different channels in unison for training.

D. Exploratory data analysis

The data from the microphones is very clean, there are no undefined values or outliers.

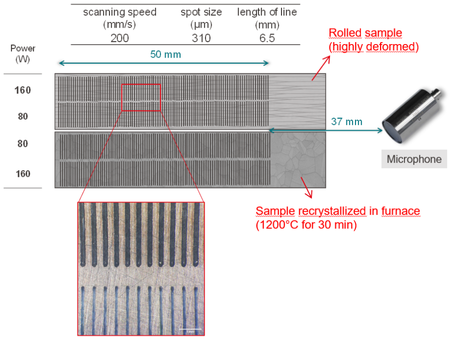

Fig. 1. Experimental setup dimensions

It looks very exploitable from a signal processing sense as we see from the Fourier transforms (Figure 2) of subsets of the experiments : We can extract frequencies even without any prior high pass filtering for noise. One might think it possible to identify the condition of the alloy with more primitive techniques like simple frequency distribution analysis. That is not the case: the laser is constantly moving, stopping, restarting, shifting and that makes it very hard to analyze the waveform as a whole or in smaller splits with more conventional techniques.

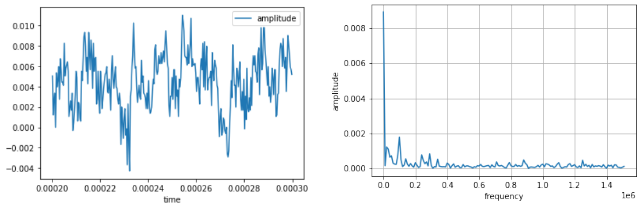

Fig. 2. Extracted sub-signal and Fourier transform of the extracted sub-signal

E. Data splitting

We use airborne sensors with a frequency of 3Mhz. The article by the laboratory of advanced materials Thun, Switzerland mentions that optimal time slices were 160 ms so we will use this time slicing to generate our subset of features.

In addition to the optimal slices, The laser power off and microphone power off are not synchronized. So we identify the noise values for which the microphone detects nothing at the end with a threshold and we drop these data from the data set to avoid feeding useless data to our model and make computations faster.

1) Train and Test set generation: In order to have a good division of the data set, we need the train set and the test set to contain disjoint parts of the recordings for each experiment. Indeed we cannot mix frames because we might end up having parts of the same audio on different splits, which would make the test step less effective. Therefore, we build the train set either by taking the set of samples associated with a continuous part of the recording of size 70% of the data or either by combining two sets of samples associated with two different continuous parts, one at the beginning of the recording and one at the end such that the sum of their size is equal to 70% of the data. The test set is the rest of the data so it contains 30% of the data. Now that the data is split, we can use it to train a model. In order to classify the data points between RX and deformed, we assess the label 0 to the class deformed and the label 1 to the class RX.

IV. Machine Learning Models

Following the processing and analysis of the data, we implemented three Machine Learning models in order to attempt identifying the recrystallization phenomenon of the alloy upon Laser Fusion.

A. Wavelet Transform and Support Vector Machine

The initial method implemented to identify the recrystallization during the experiment was the Support Vector Machine learning technique. The input of the model consisted of the wavelet transform over the raw signal, followed by a dimensionality reduction technique to reduce the number of features. The main library used to perform on the data is Scikit's Python library for predictive data analysis.

1) Feature Extraction from raw acoustic signals: Wavelet Transform: To extract meaningful information from the signals Continuous Wavelet Transform was used on the raw data split in slices of 160 ms. This signal representation offers significant advantages over other frequency transform functions such as the Fourier transform. In particular, it reduces the amount of data and enables noise filtering. The mother wavelet used for this purpose is the ricker basis. The transform allows the use of a widths parameter, to which the wavelet is stretched to before the convolution of the wavelet with the data, generating additional features from the base raw signal.

2) Dimensionality reduction: Principal Component Analysis: The wavelet used for feature processing was stretched up to a factor of 30, the given transform was then standardized and introduced into the Principal Component Analysis model. The original data set contained 30 features which were reduced to less than 4 using PCA. The number of principal components is chosen as the minimum number such that 95% of the variance is retained. This allows the capture of strong patterns in the data set and avoided overfitting of the signals on the training set. Additionally, performing PCA on the input data improved the speed of calculation during train time.

3) Support Vector Machine: Once the data has been processed, it is introduced into an SVM model with the following parameters; the choice of the Kernel depends on the shape of the data since our data is non-linear, therefore the most adequate kernel for this context is the Radial Basis Function. Furthermore, the number of support vectors chosen for the model was decided upon testing. This parameter had an insignificant change on the predictions, hence we have decided to use the minimal number of vectors (1) to perform the SVM, which also improved the speed when fitting the data.

B. YAMNet : a transfer learning TensorFlow based implementation

YAMNet is a pre-trained deep neural network that can predict over 500 classes of different audio events. In the context of this project, we will use the model as a high-level feature extractor. We try to use a concept called transfer learning to convert Yamnet to solve our own problem. The idea is the following: we keep the neural net layers that take the input files and transform them into features. This part has been trained on millions of audio signals and the hope is that the extracted features of Yamnet are also the same that is needed for classification. We then design our own neural net that takes these extracted features as input and performs a classification decision. The advantage of this is that we do not need extensive training to teach our model to extract relevant features and instead only have to train a simple mono-layer model.

1) Input: The neural network takes audio waveforms in .wav format as input to make independent classification from a large-scale data set of manually annotated audio events. The model accepts an array containing a waveform of arbitrary length, represented as monaural 16 kHz samples in the range [-1.0, +1.0]. We think that going from 3Mhz samples to 16kHz doesn't change anything because we still provide the same amount of information, the audio of an experiment just goes from 10 seconds to 30 minutes (instead of subsampling).

2) The model: In this project, we will use the model's first layer to generate the features and feed them into our own neural network that contains a unique hidden layer. We will then train the network on a small amount of data to get the classification of our audio samples. The classification model is a simple sequential model with one hidden layer and two outputs.

3) Output: The imported YAMNet model returns three outputs, namely the class scores, the embeddings and the log mel spectogram as seen in Figure 3. We check that the YAMNet model is behaving properly on the data by running it with its original classifier layer. The classification outputs for our signal is Main Hum, which is a sound that sounds close to what our laser sounds like. So the model seems to be working.

C. Sequential neural network

1) Feature extraction: Librosa is a python package for music and audio analysis. It allows you to load and manipulate audio files in .wav format. A data sample corresponds to a .wav file of 160ms duration of the experiment. Using Librosa, we extract the following features from the wav files. chroma stft : Compute a chromagram from the waveform. rms : root-mean-square (RMS) energy for each frame from the audio samples. spectral centroid : indicates where the "centre of mass" for a sound is located and is calculated as the weighted mean of the frequencies present in the sound. spectral bandwidth : bandwidth of the signal in the frequency domain. spectral rolloff : computes center frequency for a spectrogram bin such that at 85% of the energy of the spectrum in this frame is contained in this bin and the bins below. zero crossing rate : the rate of sign-changes along a signal, which means the rate at which the signal changes from positive to negative or back. mfcc : Mel-Frequency Cepstral Coefficients of the signal. This is a set of features which describes the overall shape of a spectral representation of the signal.

Except for the mfcc, each of the features extracted using Librosa returns a new signal as a 1D array with values computed frame by frame. For each of them, we compute the mean of the signal and use it as a new feature. Besides, the mfcc of the signal returns a set of 20 new signals, for each of them we compute the mean as well in order to get 20 additional features. This way, we end up with 26 features per sample.

2) Model: In order to build a neural network model for this task, we decided to use the Keras library, a deep learning library based on TensorFlow. With this tool, we build sequentially a neural network containing 4 layers. The first layer is the input layer which contains 256 neurons, its activation function is the reLU function and its input dimension is 26 (the number of features). There are two hidden layers, both with reLU activation, the first one contains 128 neurons and the second one 64 neurons. Finally, the last layer is the output layer which contains 2 neurons for the binary classification between RX and deformed. Its activation function is the softmax function which returns a percentage of confidence of prediction for each label. Now that the model is built, we must find the best optimizer and loss function to train the model as well as possible. They are compared in the result section.

V. Results

A. SVM

The model was trained using slices of 26ms on the audio that were selected at random. Furthermore, it was tested on a corresponding 30% portion of the dataset. Following the wavelet transform, PCA yielded in a decrease of the features' dimension between 73% and 86%, corresponding to a reduction of 4 to 2 features respectively. The overall accuracy of the model is 50%. Adjusting the model parameters did not have any influence on the results, this demonstrated that the low accuracy was a consequence of the features insufficiently distinguishing the two states. For this reason, the features of the signal were extracted using Librosa, as detailed in the Feature extraction section of the simple neural network. As demonstrated in table I, the overall score improved to 99% of accuracy and high precision and recall rates.

B. YAMNet

The YAMNet model is very efficient at outputting features for its initial classifier. It has been so well trained for this goal that any of the extracts from the data set are classified with close to exactly the same numerical values in embeddings. Our subsequent sequential layer is unable to differentiate between the two experiments because it does not have enough difference in the extracted features. This results in, depending on how the training is conducted, systematically classifying

SVM w/ Wavelet	precision	recall	f1-score
class 0	0.5	0.49	0.5
class 1	0.5	0.51	0.5
accuracy			**0.5**
SVM w/ Libra	precision	recall	f1-score
class 0	0.98	1	0.99
class 1	1	0.98	0.99
accuracy			**0.99**

TABLE I
AVERAGE CONFUSION MATRIX USING SVM

either always RX or always conduction : the YAMNet transfer learning structure is clearly not adapted to solve our problem. It doesn't even make sense to talk about metrics like accuracy, F1 or recall because the problem comes directly from the features.

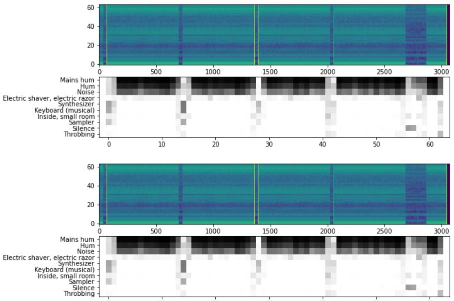

Fig. 3. log-mel spectrogram results of yamnet feature extraction of two random extracts from deformed/rx

C. *Sequential neural network*

Using Keras, we train the model on the training set we generated. Keras library offers many choices of loss functions. Among them, only one loss function was performing well which was the sparse categorical crossentropy function which computes the crossentropy loss between the labels and predictions. With this loss function, the model ends up getting 1.0 accuracy on the training set for any optimizer, while the other loss functions led to very low accuracy for the model on the training set for every optimizers (at most 0.6). After selecting the loss functions we had to choose the best optimizer for the model. Keras library offers three different optimizers which are SGD optimizer, the adam optimizer and the RMSprop optimizer. To find the best of them ,we tested each optimizer on 100 random splits from the data in which the training set was 70% of the samples and the test set 30%. The results of the different optimizers are written in Table II. We can see that the SGD optimizer has lower performances than the two others. Indeed, the mean accuracy on the test set using SGD was about 91.8% while the two other optimizers led to an almost identical mean accuracy of about 99,4%. To try to decided which of them is the best we computed the mean and the standard deviation of the loss on the test set. We observe that the mean loss is about 1.24 times greater with the adam optimizer than with the RMSprop optimizer. Besides, the standard deviation of the loss is about 1.76 greater with the adam optimizer than it is with the RMSprop optimizer. Based on these additional information we decided that the RMSprop optimizer was the best for our model.

Performance of the model for different optimizers	Adam	SGD	RMSprop
Mean accuracy	0.9941	0.91815	0.9947
Mean loss	0.0407	0.4920	0.0328
Std loss	0.0745	0.0385	0.0422

TABLE II
PERFORMANCE OF THE MODEL FOR DIFFERENT OPTIMIZERS ON THE TEST SET FOR 100 ITERATIONS

VI. DISCUSSION

Overall, we have reached an accuracy of 99% to classify the recrysallization of the alloy with the SVM and Sequential Neural Network. As mentioned previously, Yamnet's neural net embeddings are so efficient for general classification and not tuned enough towards our problem that they are unable to provide enough information for RX classification.

A. *SVM*

As stated in the Results section, using the Continuous Wavelet Transform on the raw signal to extract data did not perform as well as using Librosa. The Wavelet transform generated features did not yield satisfying results for the SVM. This is due to the fact that the wavelet transform was performed on slices of 160 ms, however, the SVM model does not support large data sets. The capability of the SVM is limited to slices of 18ms in order to have a feasible run time. This is a fairly insufficient time frame to train the model with respect to the shape of the signal and how the experience is conducted. Indeed, these small slices may be silent zones or other sounds that are not significant enough for the model to predict an output accurately at test time. Whereas extracting features with Libra on the slices of 160 ms, generated a data set of 57 samples of 26 features each. These features characterized the recrystallization of the alloy efficiently and allowed to run the SVM in a computational feasible time.

VII. SUMMARY

At the start of the project, we expected YAMNet to perform better than the Neural Network and much better than the Support Vector Machine. However, the results attest in similar scoring using a simple Support Vector Machine and a Sequential Neural Network, reaching an overall accuracy of 99%. To that extent, we have concluded that the principal difficulty of this classification problem was to extract the correct features from the raw audio signal. On the other hand, the machine learning model used over the data set need-not be deep or trained over long epochs, as we have reached the same accuracy on a simple SVM than on the neural network with the appropriate signal decomposition. Lastly, an interesting improvement could be to implement one of the models in the experiment pipeline in order to make the classification occur in real time during the experiment.

[She+18] S.A. Shevchik et al. “Acoustic emission for in situ quality monitoring in additive manufacturing using spectral convolutional neural networks”. In: *Additive Manufacturing* 21 (2018), pp. 598–604. ISSN: 2214-8604. DOI: https://doi.org/10.1016/j.addma.2017.11.012. URL: https://www.sciencedirect.com/science/article/pii/S221486041730132X.

[Mas+21] Giulio Masinelli et al. “Artificial Intelligence for Monitoring and Control of Metal Additive Manufacturing”. In: *Industrializing Additive Manufacturing*. Ed. by Mirko Meboldt and Christoph Klahn. Cham: Springer International Publishing, 2021, pp. 205–220. ISBN: 978-3-030-54334-1.

[Pan+21] Vigneashwara Pandiyan et al. “Semi-supervised Monitoring of Laser powder bed fusion process based on acoustic emissions”. In: *Virtual and Physical Prototyping* 16.4 (2021), pp. 481–497. DOI: 10.1080/17452759.2021.1966166. eprint: https://doi.org/10.1080/17452759.2021.1966166. URL: https://doi.org/10.1080/17452759.2021.1966166.