Development of a Data-driven Surrogate Model for Scale-bridging in Battery Modelling Application

Harshwardhan Dhumal[1], Tobias Aubel[1], David Koch[1], Nils Karajan[1], Karsten Keller[2], Andre Mielke[2]

[1]DYNAmore an Ansys Company
[2]Institute of Structural Mechanics and Dynamics in Aerospace Engineering, University of Stuttgart

1 Abstract

In numerous mechanical engineering applications, the use of multiscale computational modeling and simulations is imperative. Nevertheless, the computational challenge persists in addressing complex multiscale systems due to the vast dimensionality of the solution space. The field of machine learning (ML) has experienced ongoing development as a feasible option that might potentially expedite, substitute, or complement traditional numerical techniques. Recent studies have demonstrated that (ML) has the capability to effectively address a range of differential equations, including boundary value problems, while maintaining a desirable balance between accuracy and computing complexity. The present study is centered on the finite element simulation of batteries, with a particular emphasis on the modeling of battery cells. The primary objective of this study is to develop a comprehensive finite element (FE) model of a battery cell that accurately simulates the physical characteristics and microstructural behavior observed in real-world battery cells. The incorporation of micro-structural cell response into full-scale models is frequently acknowledged as a challenging and time-intensive undertaking. The desire for the development of computationally inexpensive models is driven by the inefficiency of combining intricate models into bigger simulation models. This paper presents a novel approach that utilizes large-scale datasets and ML techniques to develop a multiscale strategy. In this study, a comprehensive dataset of simulations is generated at the micro-scale through the development of a detailed FE model. The simulations are conducted using the LS-DYNA software suite, incorporating the idea of Representative Volume Elements (RVE) for homogenization purposes. Subsequently, the collected data is employed for the purpose of training a ML-based surrogate model. Ultimately, this surrogate model has the potential to be integrated with LS-DYNA through an UMAT (User Material Subroutine). Consequently, it can be incorporated in subsequent endeavors involving battery modeling and simulations.

2 Introduction

The continuous progress in the field of Automobiles suggests that Electromobility is poised to become an essential component of the energy mix in the foreseeable future. It is imperative to consistently contribute to the rapidly expanding sector of electric vehicles. Taking into consideration the aforementioned point, the project "DIGItalization for SusTAINability (DigiTain)" has been initiated by the Bundesministerium für Wirtschaft und Klimaschutz (BMWK). The objective of this project is to establish diverse processes, methods, and models for the complete digitalization of product development in the context of sustainable electric vehicle architectures. The present research work is a component of the DigiTain project and is driven by the fundamental concepts and principles that underpin the project.

2.1 Motivation

The developments in battery technology have necessitated the development of robust methodologies for battery testing, thus driving progress in battery modelling and simulations. Considering this, it is imperative to use a scale-bridging methodology to effectively establish a connection between these disparate scales, hence optimizing the outcomes of the computational battery model. While multiscale simulation methods have demonstrated their efficacy in precisely modelling complicated systems, they are characterized by their high computing expense, significant time requirements, and challenging implementation process. However, attempts can be made to solve the limitations by the utilization of advanced solutions offered by the field of ML. ML is an academic discipline that emerged because of the pursuit of artificial intelligence (AI). The concept of allowing computers to acquire knowledge from data during the nascent phases of the academic discipline of AI piqued the interest of certain scholars. The utilization of Neural Networks in conjunction with the multiscale modelling methodology can yield optimal outcomes by combining the strengths of both methodologies.

Therefore, the primary objective of this study is to construct a surrogate model that can serve as a valuable instrument for including the scale bridging phenomenon into conventional battery simulations.

2.2 State of the Art

The crashworthiness simulation of a battery pack can be carried out using an explicit FE code like LS-DYNA to analyze the effect of materials and thickness on the strength of the battery [1]. Due to the involvement of different length scales, numerical simulations of heterogeneous systems like battery modules for electric vehicles are very difficult. It is impossible to reproduce the collision scenario of the entire vehicle while resolving all length scales, even with the most advanced computing equipment. These obstacles makes it a bit difficult to comprehend the mechanical response of battery packs in car crash scenarios. The problem of multiple length scales can be resolved using the RVE technique which is based on the classical homogenization theory [2]. Although these methods provide results with exceptional accuracy, they are still computationally very expensive and so, it is necessary to further accelerate the RVE analysis using advanced ML and model reduction techniques in order to pave way for highly effective concurrent simulations. These integrated data-driven modelling framework based on process modelling, machine learning, concurrent simulations and material homogenization are promising in simulation of injection-molded fiber reinforced composites, additive manufacturing and compressive molding components [3].

3 Methodology

3.1 Detailed FE Model

There are already a number of approaches available for simulating the mechanical response of a pouch cell. Generally, there are certain modelling approaches that present each and every single layer of the battery cell and detailed description of the boundary conditions and failiure of the individual components of the cell such as anode, cathode and the separator, which can further aid in predicting the internal short cicuit or damage to the cell. Indeed, the high level of details in the model are accompanied by a high computational cost due to the large number of contact between the layers and the small discretization of the elements. On the other hand, in the macroscopic models the individual layers are not discretized separtely, but are directly averaged or homogenized in either the thickness or the radial direction in case of pouch or cylindrical cells respectively. These models are typically prefered over the detailed models because of their exceptional effeciency in terms of computational power. However, they always fail to capture the failiure modes of the individual layers at the microstructure level. As a result finding a perfect trade off between these two methods and combine the computational excellence and accuracy of both methods has become the field of interest. Hence, in this work a detailed FE model is created at the micro level which is furhter assisted by homogenization method and neural networks in order to reduce the compuational effort.
The detailed layer wise FE model is created using the LS-PrePost software package which is used for preprocessing and postprocessing of CAE models. This model replicates the actual section of a physical pouch cell in a domain of 2.5 x 2.5 x 0.2 mm. The detailed FE model of the battery cell is created by modelling each layer separately and retaining the minute physical and microstructural details of the cell. Fig. 1. illustrates a representation of the microstructural model of the Li-ion pouch cell. The model is built with separate layers for the individual component for anode active material, anode current collector, cathode active material, cathode current collector and separator.

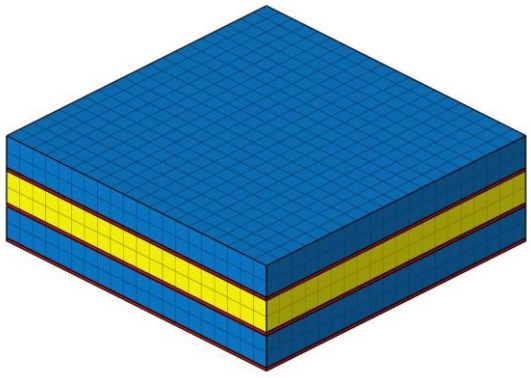

Fig.1: Detailed FE model.

The material properties for this detailed model have been derived from the work done by Schmid et al [4], regarding modelling of Li-ion pouch cell for simulation of their mechanical behavior. One of the problems that is faced during modelling of such heterogeneous geometries is the modelling of the very thin layers as well as resolution of the elements with a very small characteristic length. To overcome this issue a detailed heterogenous model is defined and then used along with the RVE analysis card to have a homogenization simulation. For this study, a concept is employed, where the separator layers are defined predominantly using two types of elements, namely shell and solid elements. The shell elements will try to capture the in-plane behavior of the individual components, while the solid elements will simulate the out-of-plane behavior. The out-of-plane behavior is mainly defined by the shear response, interaction between the individual layers and the pressure in transverse direction [4]. After successfully establishing the microstructural model of the RVE, it becomes crucial to comprehend the formulation of the RVE analysis inside the FE framework.

3.1.1 Contact Analysis

There are potentially two approaches to construct the model with regards to the definition of contact. The two types of models created to compare the contact definition are: model with merged nodes and model contact definition. Fig. 2, displays the outcomes of the RVE model simulation under identical boundary conditions, but with two distinct contact definitions.

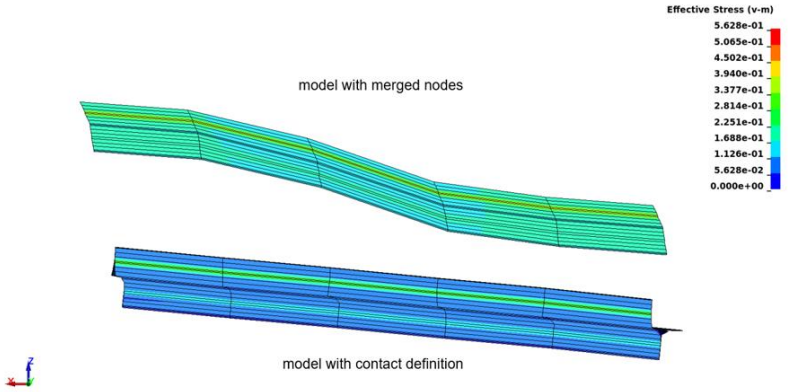

Fig.2: Comparison of contact type definitions.

Fig 3. illustrates that the force displacement response of both models, as obtained from the output file of the RVE homogenization process, has a high degree of similarity. Additionally, the model with merged nodes demonstrates efficient normal termination, boasting a remarkably short runtime of about 37 seconds and requiring minimal computational effort. However, the model with contact definition predominantly results in termination errors caused by convergence issues, exhibits somewhat slower performance, and requires a significant computing effort. Due to this, the model with merged nodes is deemed impractical for subsequent data generation due to its limitations. In contrast, the model with merged nodes is considered more suitable due to its comparable accuracy, straightforward configuration, and lower computational expenses.

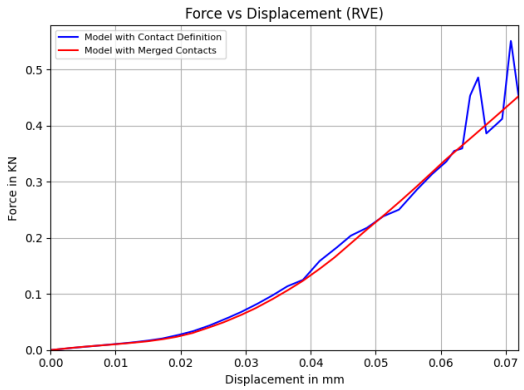

Fig.3: Force vs Displacement plot for the RVE.

3.2 The ML Method

The fundamental concept of ML is allowing the computer to autonomously acquire the link between inputs and outputs, rather than explicitly presenting this relationship and afterwards calculating the outputs based on given inputs. Artificial neural networks (ANNs) demonstrate efficacy in handling numerical data exclusively and representing one-to-one relationships between inputs and outputs, provided such relationships exist. Convolutional neural networks (CNNs) are a specific type of ANNs that incorporate skip connections and weight sharing. These architectural features offer significant benefits for processing picture inputs, as they result in a substantial reduction in the number of weights required compared to fully-connected ANNs. However, Recurrent Neural Networks (RNNs) become suitable for modelling history-dependent material behaviours. In practical applications RNNs face various problems like the vanishing gradient issue that made them inappropriate for long-termdependency problems. This led to the formulation of certain advanced architectures of RNN, namely the Long short-term mermory (LSTM) unit and the gated recurrent unit (GRU). In this work the LSTM unit has been utilized and hence will be discussed in the subsequent section. The LSTM unit can be considered as an enhanced version of the basic RNN unit. Both the RNN nad LSTM units are illustrated in Fig.4.

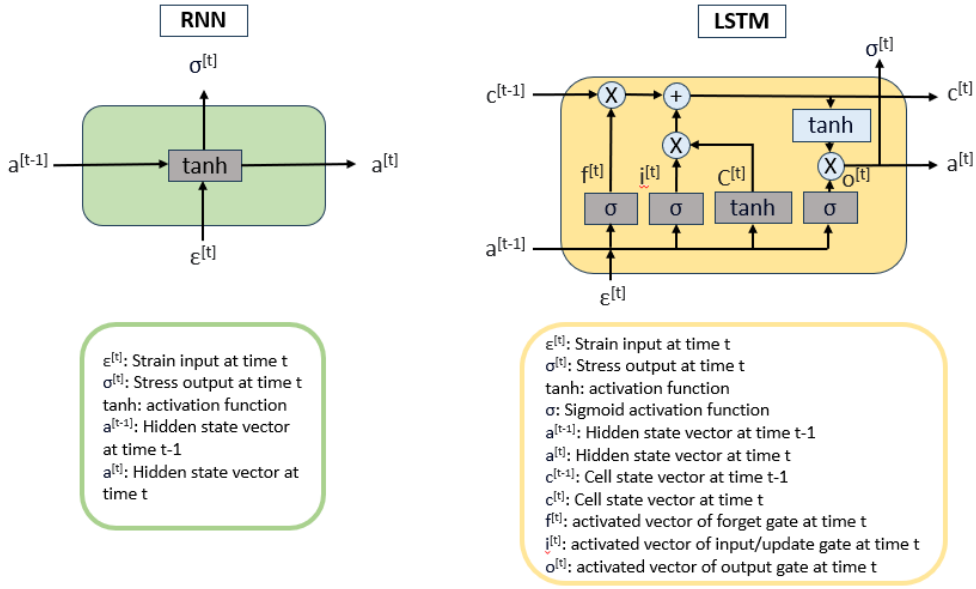

Fig.4: The RNN unit and LSTM unit.

3.2.1 Data Generation

The dataset holds significant importance in ML-related studies as the ML model's learning of underlying mechanisms relies on the data used for training. The objective of this study is to train the model using stress-strain values that the battery cell will experience during a full-scale car crash simulation. To this extent, the FE model of a side impact crash simulation of an electric hybrid vehicle is used. In this model the elemnts of the battery module are modelled using the state of the art concept of Batmac elements and Randals circuit in LS-PrePost. This assists in simulating the coupled response of the battery cell elements which combines the mechanical, electrical and thermal response of the cell. A python script is created for extracting the strain versus time behaviour of different individual elements of the battery cell components. The full scale simulations runs over a simulation time of 14 ms and one can successfully extract the strain paths for the six components of strain from different elements using this python script. LASSO-python is an open source python package is a tool which allows the user to access the output files from LS-DYNA (d3plot, binout, etc) and retrieve the required data from such files, and so it is extensively used for this task. Fig. 6. shows the various strain paths plotted against time, that are extracted from the full-scale crash simulation.

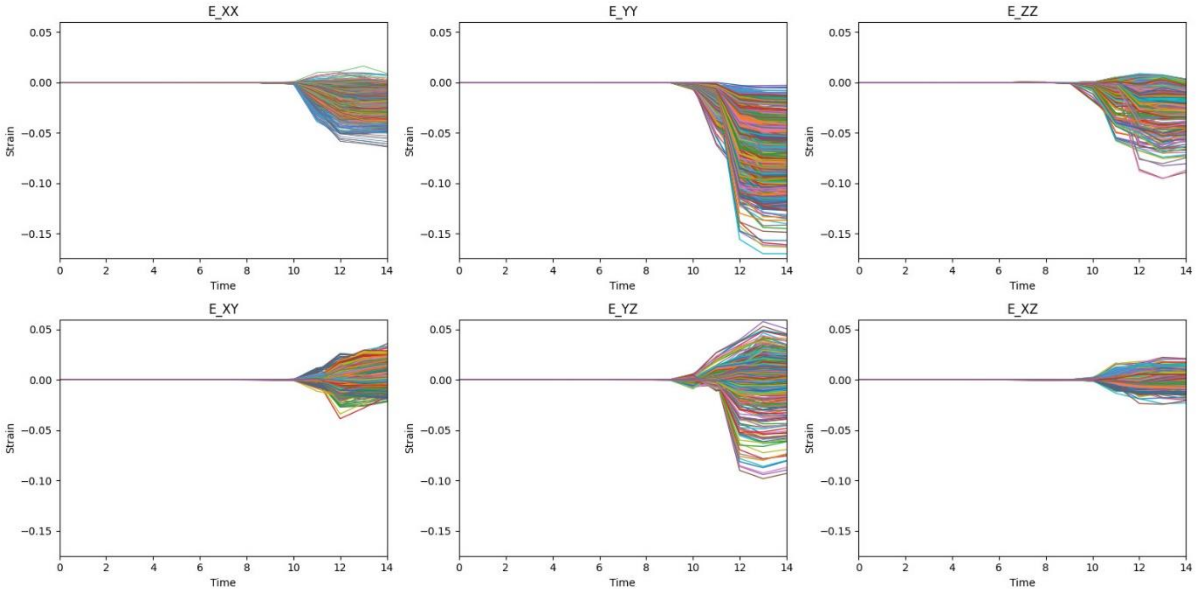

Fig.5: Strain paths from full-scale crash simulation.

The previously generated detailed FE model will form the backbone of the the entire data generation process. The process of acquiring data to train our model is rather simple and straightforward. Let us consider just one strain path from the strain paths which were extracted in the previous step. The strain-time response of the combined six components of strain will serve as the input for the RVE techique-based simulation of the detailed FE-model of the battery cell. The rveout file is generated at the end of the simulation and holds the macroscopic or homogenized values for the RVE model. The strain values and the corresponding stress values can be extracted from this output file at every timestep and stored accordingly. A tool has been created using python which fetches a unique strain path, writes these values to the keyword file of the RVE model, runs a LS-DYNA simulation, extracts and stores the stress strain data successfully in a data structure and keeps on repeating this process recursively untill all the strain paths have been utilized. Each simulation generates a separate data point which in turn generates the stress and strain response that can be used as outputs and inputs respectively for the model training process. Python is used to further clean and prepare the generated data and present it in a more organized way in order to make it ready for the further steps. Two different data structures are created for strain and stress respectively. The dimension of the data structure is such that it can hold the information of all the six individual strain components and different timesteps for different elements respectively. Each simulation generates output for 70 different timesteps leading to 945 unique simulation runs. The final dimension of the data structure is 945 x 70 x 6. Towards the end of this section, two different data structures for the stress and strain values are created, which can be effeciently used as output and input to the ML model in the subsequent steps. Fig 6 tries to illustrate the entire data generation process in a simplified way.

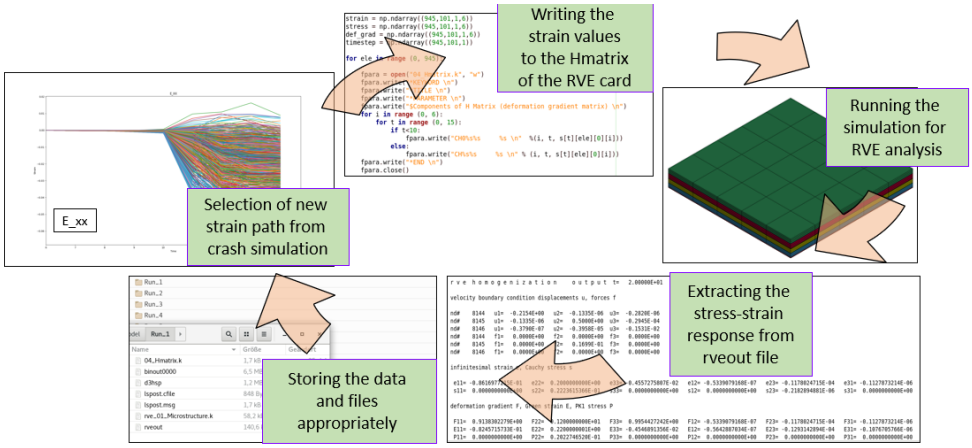

Fig.6: Workflow for data generation.

3.2.2 The Sliding Window Approach

The sliding window approach is employed to achieve an efficient training and verification process for the model [5]. This method of training helps the model to learn the pattern of the sequential data to make accurate predictions of the stress response. This is accomplished by partitioning the complete dataset into smaller segments of time-dependent data, each spanning the timesteps specified by the designated window size. Each of these smaller fragments can be denoted as a window and the data can be redistributed such that each of the windowed section can slide through the entire dataset and hence the name sliding window approach. For better understanding, consider a data point from the previously generated dataset. The individual strain path from Fig. 7 spans over 70 timesteps and the same data structure has the dimension as 1 x 70 x 6, where the evolution of all six components of strain over the 70 individual timesteps can be observed. Assume the window size for this case to be 5. So according to the sliding window approach, a new dataset will be generated with each of the data point having the dimension size as 1 x 5 x 6, where the first data point contains the information of the strain components for the first five timesteps, the subsequent data point will have the strain components for timestep number 2 to timestep number 6. This window will slide smoothly across all the individual data points, having an overlapping data storage until it reaches the values at the final timestep. So, in conclusion the dataset in the beginning having the size as 1 x 70 x 6 will be converted to the dataset having the size as (70-5) x 5 x 6 = 65 x 5 x 6. Recall the data generated in the section 3.2.1 with the dimension size of 945x70x6will be translated to the new size of (945 x 65) x 5 x 6 = 61425 x 5 x 6 using a user defined script that works on the principle of the above stated, sliding window approach.

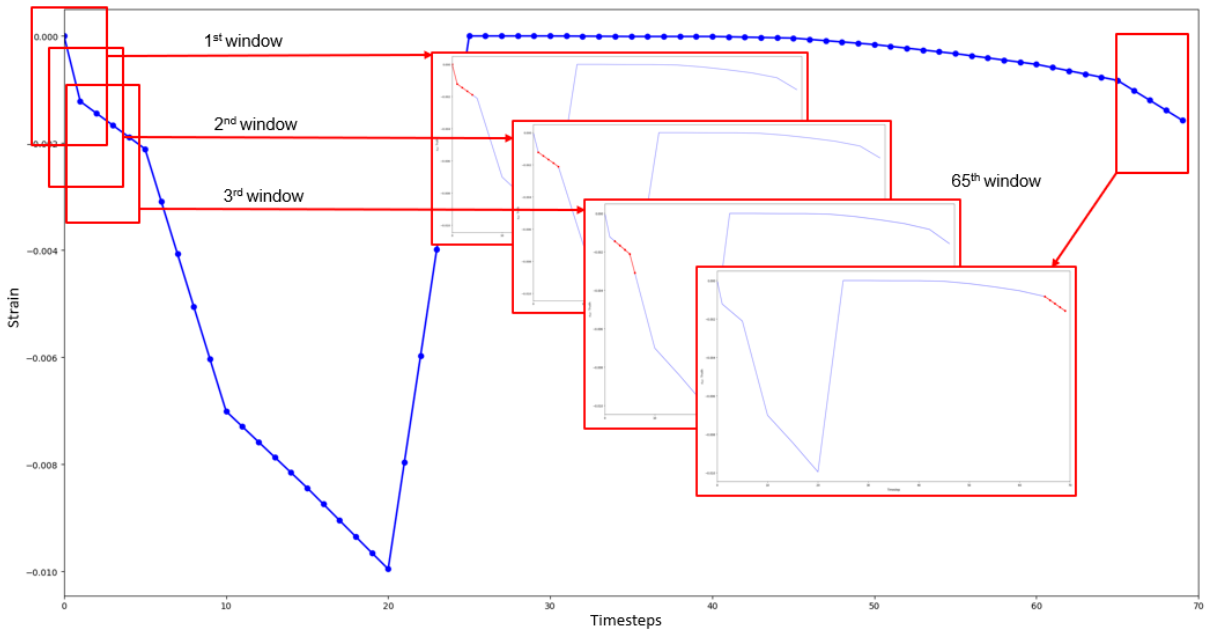

Fig.7: The sliding window approach.

3.2.3 Data-driven Surrogate Model: Training and verification

As discussed earlier, RNNs have the unique capabilities to replicate the stress-strain response of the battery cell model. Further the LSTM unit which is one of the types of a RNN is chosen to overcome the drawbacks of a classical RNN type neural network. The code for the model training and validation part is created in python programming environment and the Tensorflow package has been imported to define different types of artificial layers like a dense layer, LSTM layer etc. The LSTM model is built using two LSTM layers having 40 neurons each and the tanh activation function. The optimizer Adam is used along with MSE as the loss function. The data generated in the previous step is organized in a time-dependent way and can be used directly to train a regular deep neural network. For this purpose, each data point is further manipulated to have six components of strain at the input end and the corresponding stress components of stress at the output end. The sliding window approach discussed in the previous section is used to prepare the data with a final shape of 61425 x 5 x 6. Finally, there are 61425 unique data points of input and output data. This dataset is divided into a 90:10 train test split and the training data is applied to the input layer of the LSTM model.

Parameter	LSTM Model	Modified LSTM Model
Number of hidden LSTM layers	2	2
Number of neurons per layer	50	50
Activation Function	tanh	tanh
Optimizer	adam	adam
Loss Function	mse	mse
Metrics	R2 score	R2 score
Epochs	500	500
Window Size	5	3
Input Features	ε11, ε22, ε33, ε12, ε13, ε23	ε11, ε22, ε33, ε12, ε13, Δε23, Δε11, Δε22, Δε33, Δε12, Δε13, Δε23
Input Dimension	61425x5x6	63315x3X12
Output Feature	σ11, σ22, σ33, σ12, σ13, σ23	Δσ11, Δσ22, Δσ33, Δσ12, Δσ13, Δσ23
Output Dimension	61425x5x6	63315x3x6

Table 1: Details of the training process for LSTM and modified LSTM model.

Further, the various details of the LSTM architecture are illustrated in Table 1. Finally, the model training is initiated with a validation split of 20% and the epoch for training is set to 500.The trained LSTM model is verified against the test data that is separated initially from the training data and so it is never learned by the ML model. A python script is created to further test the model and create comparison plots between the values predicted by the model and the ground truths. This script imports the ground truth values for strain and stress in two different arrays which holds the response of the battery with respect to time. Further, the script runs recursively in a loop by loading the strain component at a specific timestep, calling the LSTM model to predict the stress values, and storing them in a separate array. In the end the true values of stress and strain as well as the predicted stress values are plotted against timesteps.

3.3 Implementation of Surrogate Model with LS-DYNA

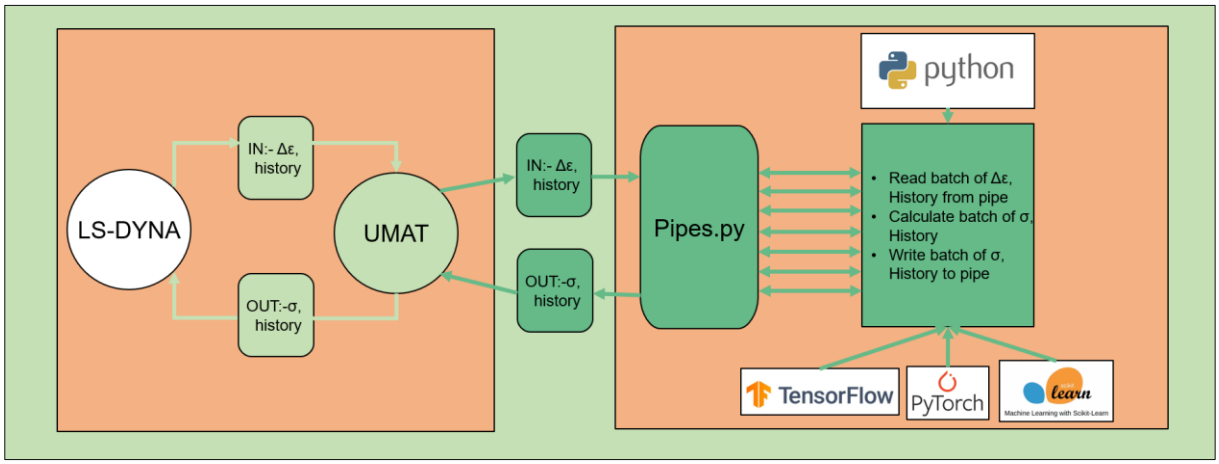

Fig.8: Pipes.py: The pipelining framework.

During a LS-DYNA simulation the LS-DYNA solver extracts the information from individual material/integration points and runs the UMAT subroutine. The UMAT file typically holds the

information of the material model and is responsible for predicting the stress response at that timestep from the given strain increments and history information. But the idea of this research is to replace the classical UMAT which is ideally defined in a fortran file by a user-defined python script. This can be achieved by using the pipelining framework developed by Joachim Sprave, which is illustrated in detail in Fig. 8. The methodology employed in this study involves the flow of data between the python script and other components over many parallel pipes. The typical flow of data is such that, the strain and history variables are transferred to the python script, where the stress prediction takes place and the updated stress values are transferred back to the UMAT or in turn the LS-DYNA solver through the pipes. This framework allows the replacement of the traditional fortran-based material model/description by a python file and facilitates the utilization of various packages like tensorflow, pytorch and scikit learn through python to achieve efficient operation of the input/output data. The surrogate model which is trained in the previous step, will be accessed through a python script and the concurrent simulation of the LS-DYNA model and the surrogate model can be effectively achieved. Fig. 9 helps to understand how the flow of data takes place through the entire co-simulation environment.

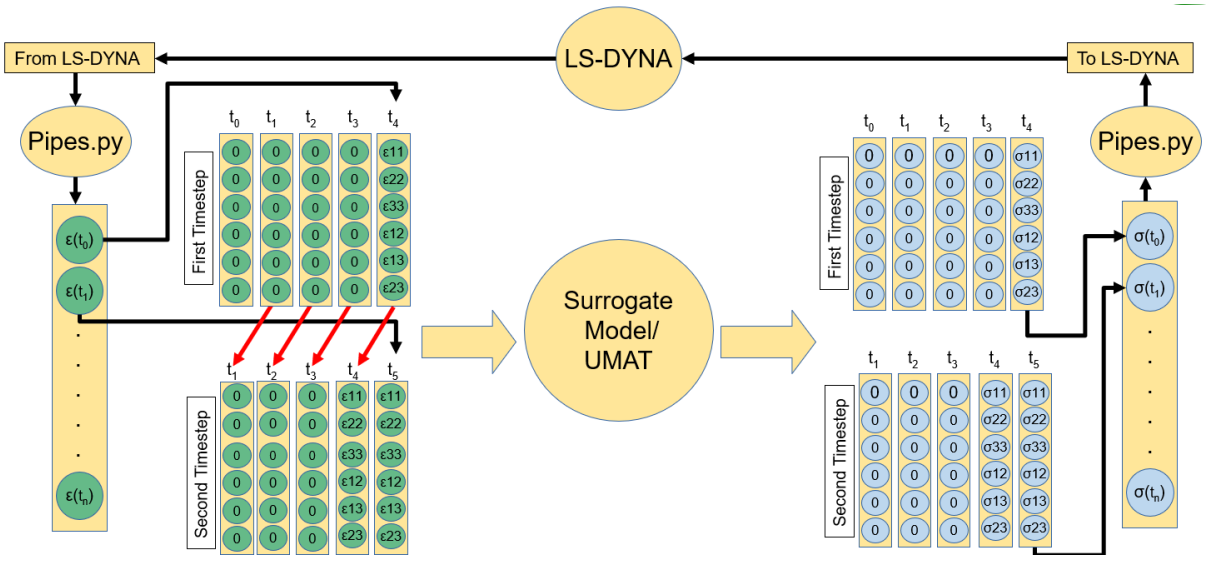

Fig.9: Implementation of surrogate model.

The LS-DYNA solver, pipes.py framework and the Surrogate Model form the backbone of the co-simulation loop. As discussed earlier the communication between the surrogate model and the LS-DYNA solver is established using the pipes.py framework. For understanding the process, assume the model is trained for absolute strain and stress values for five timesteps each so that the final shape of the input and output data to the model is 5 x 6. At the beginning of the simulation both the arrays for stress and strain are initialized with zeros. During the first timestep of the simulation, the strain increments for the same time step are extracted from the LS-DYNA solver and transferred to the surrogate model through the pipelining framework. These values will be placed at the last position of the strain array that will be applied to the surrogate model, which will predict the corresponding stress array. From this array, the stress components from the last position will be extracted and transferred back to the LS-DYNA solver for further operation. Now for the next timestep the values from the last four positions of the strain array will be placed in the first four positions of the strain array so as to store the history information and the new strain increments from the LS-DYNA solver will sit in the very last position of the array. The surrogate model will then predict the stress values, which will be sent back to the LS solver through the pipes. Now the process described above will occur in a recursive manner till the end of the simulation time. This proven methodology can be used in the upcoming stages to set up different load cases using the surrogate model to validate the results against the detailed FE-model.

4 Results and Discussion

In this section the verification and validation results of the implemented surrogate model are presented. The verification results for the data-driven model against unseen data are presented along with the challenges occurred and the corresponding proposed solution. The model is trained using data obtained from an implicit RVE analysis. Consequently, it is expected that the model's predictive accuracy may be compromised when applied to simulations conducted under different conditions. Due

to this, it is initially validated for an implicit type of simulation. Further, the same model setup is simulated using an explicit simulation and the results of the surrogate model are validated.

4.1 Testing and Verification of Surrogate Model

The LSTM model is trained using the procedure prescribed in section 3.2.3 and verified with the help of the developed python script. The corresponding results for the LSTM model are illustrated below in Fig.10.

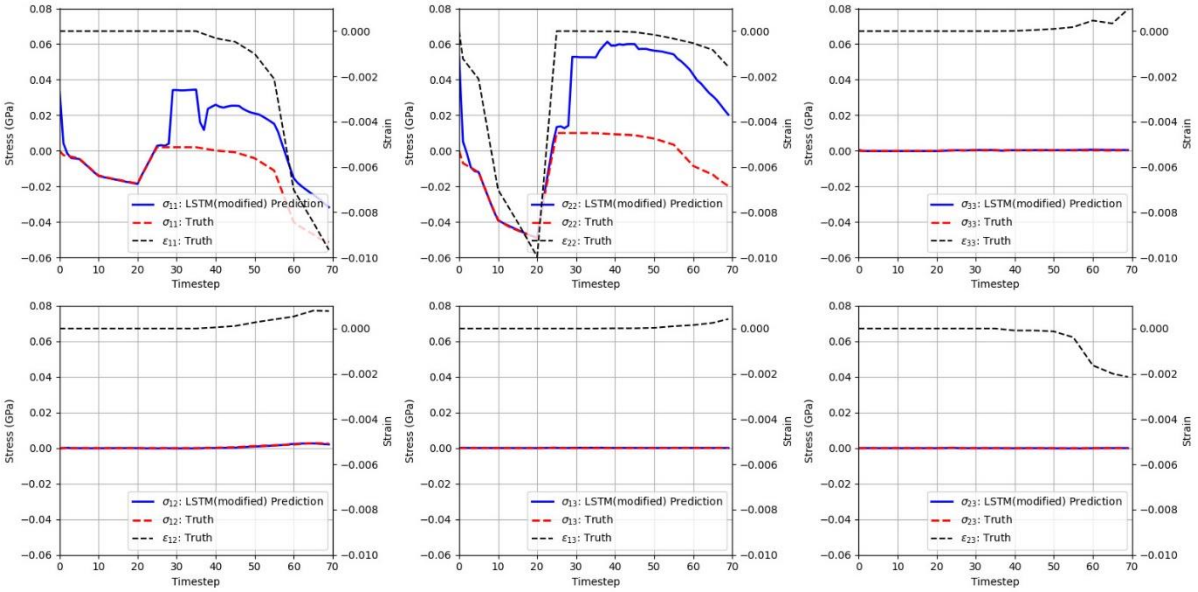

Fig.10: Verification results for LSTM based model.

The plots from Fig.10 indicate that the model's predictive capabilities are inadequate for the provided strain inputs which is further explained by the low R-squared score of 0.20809. It is evident that the model produces an inaccurate output and exhibits instability over the concluding timesteps. So, it is imperative to develop a neural network architecture to enhance the overall efficiency of the models. To do so a modified model architecture has been developed as depicted in the Table 1.

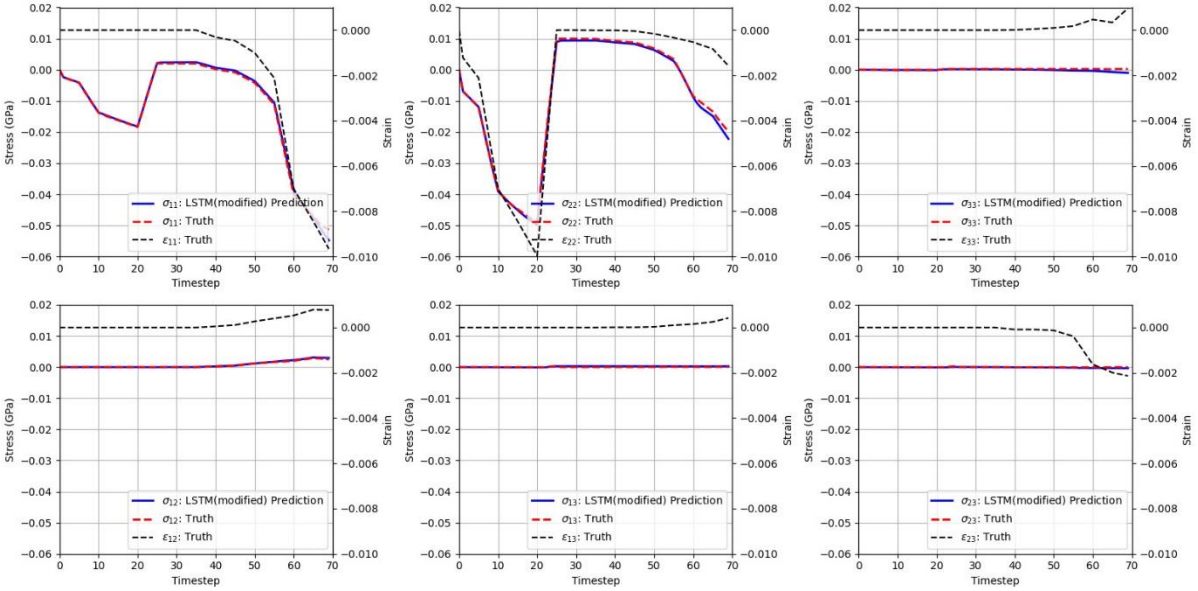

Fig.11: Verification results for modified LSTM based model.

Initially a hyperparameter study is carry out from which it is concluded that the model gives better predictions when trained with the data having a window size of 3 timesteps. Secondly the input and output feature provided to the model are modified to make the model a bit more robust. The model is supplied with both the absolute strain components and the incremental strain components at each timestep and the output to the modified model are the six components of the stress increments. By

providing such data to the model, the model can learn the trend of the input data in a more optimized way and make accurate predictions by avoiding the abrupt jumps in the predictions in the region where the strain components of the model are constant over longer periods of time. This new modified model is trained for 500 epochs and is tested against unseen data using the same python script from the previous step. These verification results are illustrated in detailed in Fig. 11. There is huge improvement in the model's prediction ability and the model can efficiently make accurate predictions as compared to the ground truth value. The high accuracy of this modified model can be attributed by the high R-squared score of 0.99825. The current model will be used in the next steps for further implementation and validation of the model in a LS-DYNA co-simulation.

4.2 Implicit Analysis

An implicit analysis is signified by very high timestep size and corresponding high strain increments for each timestep. As the surrogate model has been trained under similar simulation conditions, it can reproduce the results like a classical FE model. For the validation procedure, the results of the surrogate model and the detailed FE model are compared. To achieve this, a single-element model is constructed in LS preprocessor with the same dimensions as the detailed FE model of the battery cell. This single-element model is then initialized with the UMAT which holds the information of the surrogate model. Now both the models namely, the detailed model created using the classical FEM approach and the one modelled using the data-driven approach are subjected to similar loading conditions to have accurate comparison of the two approaches. Four different load cases are used in this step for validating the models. The first one is 'Uniaxial Compression'. For this load case the nodes from the left-most face of both the models are connected at a single node using nodal rigid bodiesd and are provided with a single point constraint (SPC) boundary condition to restrict all the degrees of freedom for these nodes. While the nodes on the right-most face are all provided with a prescribed displacement in the negative Y direction to have a compressive effect on the entire model. Similarly, for the 'Uniaxial Tension' load case the prescribed displacements are applied in the positive Y direction to simulate the tensile behavior of the model and for the 'Cyclic Tension-Compression' load case the models are initially subjected to a compressive loading followed by a tensile loading condition. To simulate the crushing behavior of the battery cell the 'Crush Test' load case is applied to the models. For this load case, the bottom face of the models is completely constrained while prescribed displacements are applied to the nodes from the upper face of the models in the negative Z direction to simulate the crushing effects of the battery cell. The detailed description of these four load cases can be seen in Fig. 12.

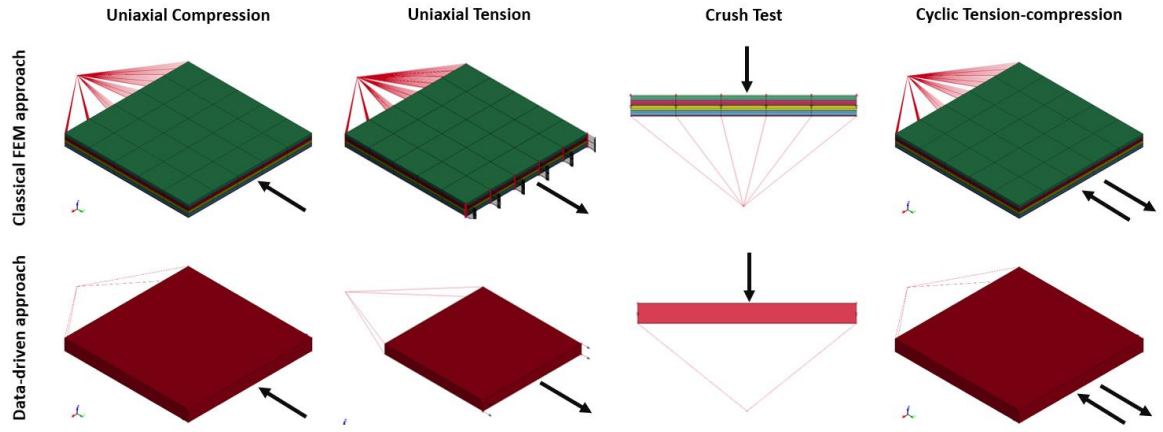

Fig.12: Load cases for implicit analysis.

The load cases discussed above are simulated for the models created using the classical FEM approach as well as the one created using the data-driven approach. The next step is to identify a reliable method to effectively compare the response of both the models. In this work, the comparison of models is done by the values of the reaction force extracted from the node at the center of the rigid elements, that has been defined using a SPC boundary condition. This information regarding the trend of the reaction force can be extracted from the binout file through the bndout section in LS-PrePost. This is possible only if the SPC2BND flag from the control card is set to 1, which allows the user to

observe the reaction forces occurring at a completely constrained node using the binout file. The comparison of results for all the different load cases is presented below in Fig.13.

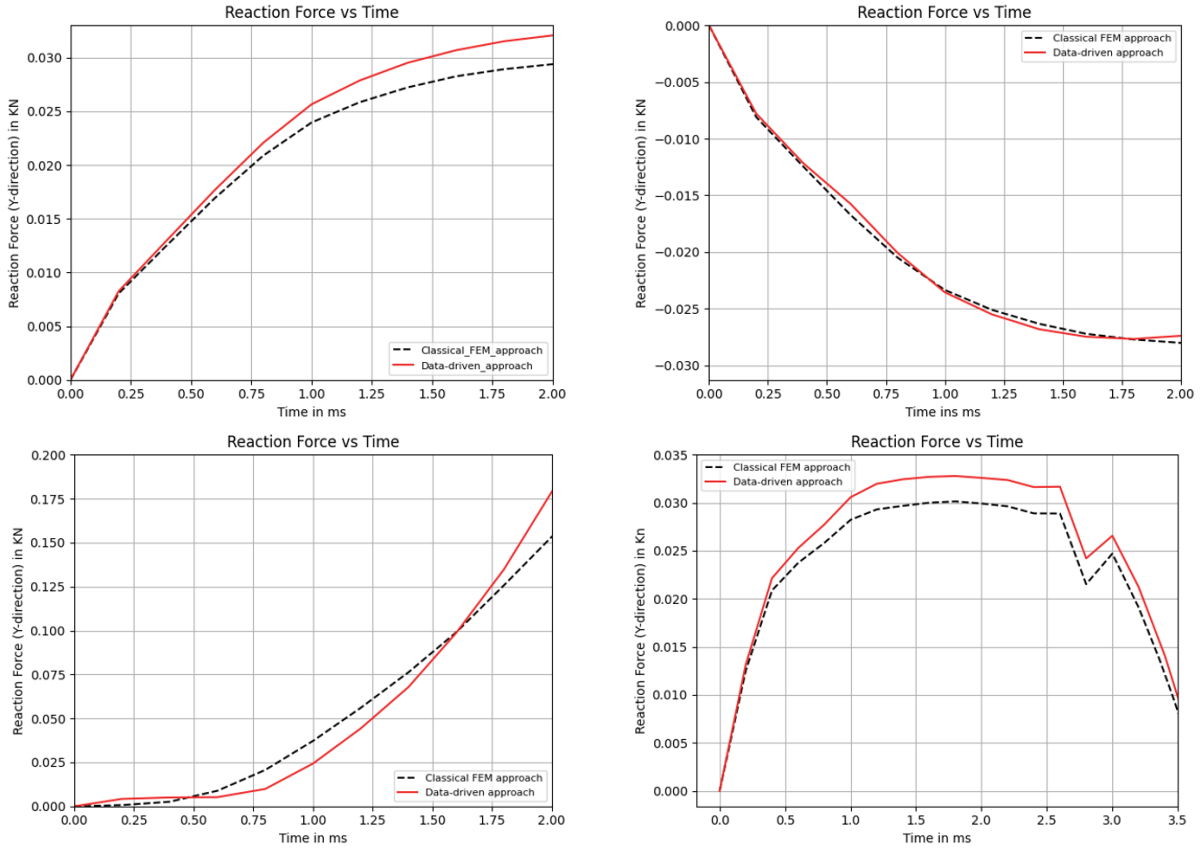

Fig.13: Comparison of reaction forces: (top-left) uniaxial compression (top-right) uniaxial tension (bottom-left) crush test (bottom-right) cyclic tension-compression.

It is clear from the results presented in Fig. 14, that the data-driven approach and in turn the surrogate model has the potential to reproduce the results produced by the classical FE modelling method. The methodology devised for the development of a co-simulation including the LS-DYNA model and the surrogate model has demonstrated significant efficacy, yielding results with high amount of precision. In case of the detailed FE model, the solver must implicitly find the solution for a very large number of elements, which makes the task computationally very difficult. On the other hand, the model simulated with the surrogate model comprises of only one element and so the computational effort for this method drastically reduces. Hence, the data-driven approach is faster as compared to the former FEM approach. The detailed model takes 4-5 minutes to run a particular load case, while the surrogate model runs the same simulation in less than one minute.

4.3 Explicit Analysis

After achieving good results in case of an implicit simulation, the next immediate step is to validate the results of the surrogate model for an explicit simulation. Two different models are created in the same way as prescribed earlier for the case of an implicit simulation. Both the models are simulated for the same load case of uniaxial compression and the comparative results for reaction forces are presented in Fig. 14. The data-driven approach clearly fails to make accurate predictions and the results are completely diverging from the ground truths or in this case the results generated by the classical FEM approach. It is important to understand the reason behind the divergence of the surrogate model predictions from the expected results. This can be clearly attributed to the input provided to the surrogate model. As established earlier, the model has been trained for an implicit timestep size which is quite large as compared to the timestep size incurred during an explicit analysis. The magnitude of the step size and in turn the strain increment size for an explicit simulation is approximately 1000 times smaller than that of an implicit simulation. When the model receives such small values as input it tends to produce erroneous outputs which further get accumulated and multiplied recursively over time. This phenomenon can be clearly seen from Fig. 15.

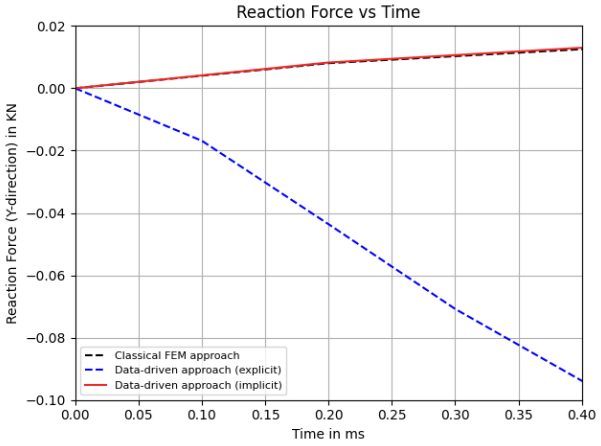

Fig.14: Comparison of results from Implicit and Explicit simulations (prediction in case Explicit simulation not good as the surrogate model is trained on data only from an Implicit simulation)

Fig. 15, shows how well the model predicts as compared to the ground truth, when the LSTM model is validated for a bigger step size and strain increments for 70 timesteps.

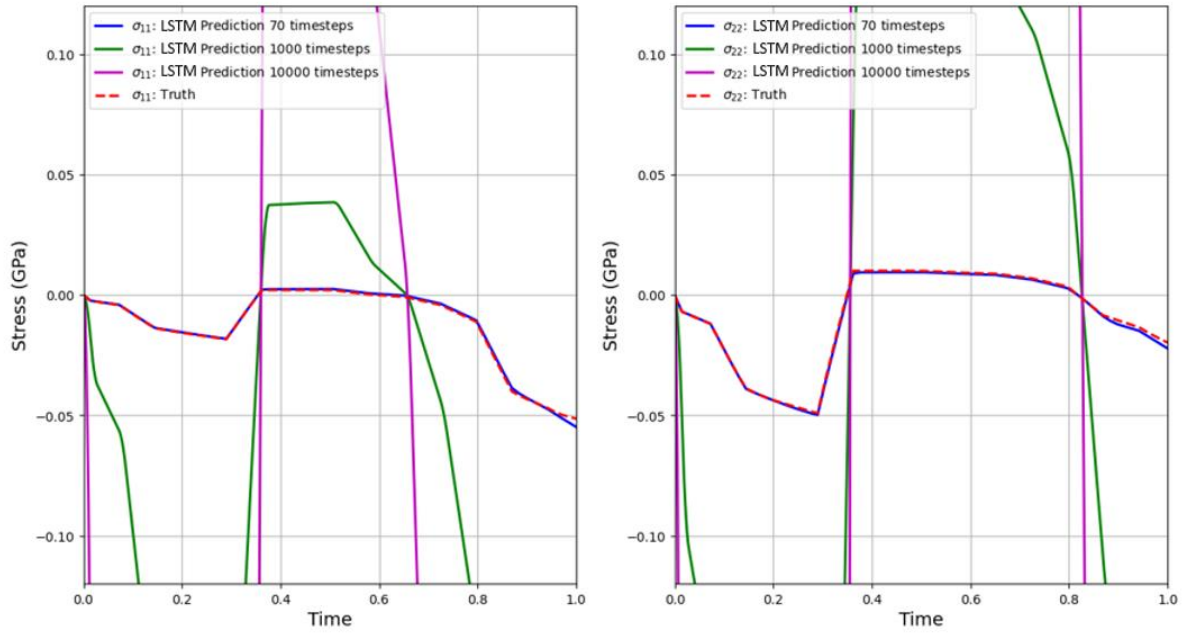

Fig.15: Surrogate model prediction for bigger step size with 70 steps.

While on the contrary, the model predicts very high values and diverges completely from the ground truth curve. As discussed earlier, the data-driven models do not have the ability to retain high accuracy in case of certain scenarios that are not the same as the training conditions. Due to this reason a model that has been trained on data from an implicit simulation cannot reproduce good results in case of an explicit simulation. The very first solution that can be adopted to tackle this problem, is to train a new surrogate model using data with a very small step size comparable to that of an explicit simulation. Also, after a closer examination of Fig.15 it can also be concluded that the model predictions are indeed scaled to a higher magnitude, but the model still manages to retain the same trend as compared to the ground truth. So, in the next step a new model is trained using a rather finer dataset with very low step size along with application of a scaling factor of 10^5 to the incremental strain values to obtain a rather optimized version of the surrogate model that can be used for attaining accurate predictions in case of an explicit simulation.
This improved surrogate model is further used in a concurrent simulation to validate it's results against the detailed FE model. The steps for setting up the model are the same as mentioned in the previous section and the comparative results for the load case of 'Uniaxial Compression' and 'Uniaxial Tension'

are depicted in Fig.16. It can be observed that the improved model demonstrates good level of accuracy and manages to follow the similar path as in case of the implicit analysis model and the classical FE model. Although the improved model manages to reproduce accurate results, it is way slower as compared to the classical FEM approach. Due to very low step size, the total number of timesteps in case of an explicit simulation increases exponentially. And so, the total number of times that the surrogate model will be accessed increases proportionally. The pipelining framework slows down the entire simulation process, and with a higher number of steps the total run time of the simulation is affected tremendously. Due to this a regular explicit simulation that requires 4-5 hours to run using the FEM approach will need more than a day to successfully carry out the same simulation.

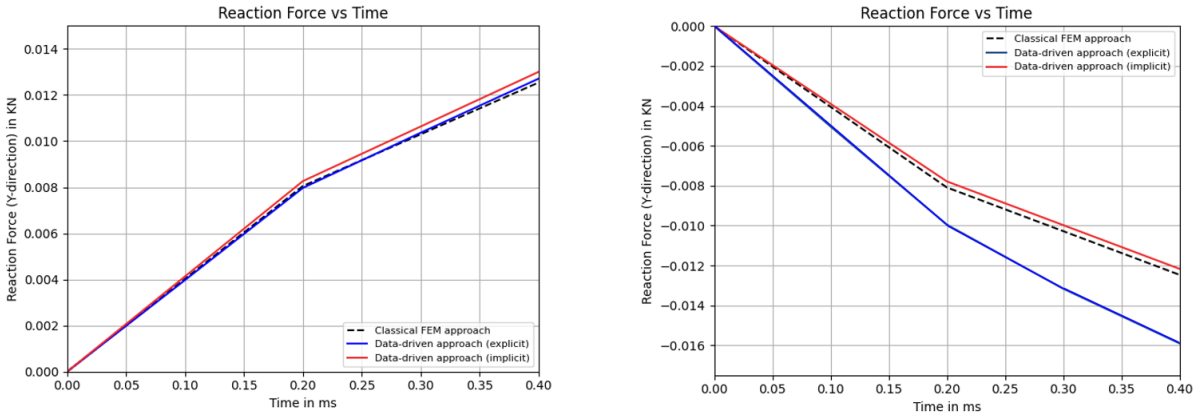

Fig.16: Comparative results for an explicit simulation a) uniaxial compression and b) uniaxial tension.

5 Summary

In this paper the methodology for development and implementation of a data-driven approach for improving the current Battery modelling techniques is presented. In the very first step, a detailed FE-model of the lithium-ion battery cell is created in the micro-scale domain. This detailed model can be treated as a RVE and further used to set up a RVE analysis simulation, in which displacement gradient components can be used as inputs to simulate the stress-strain response at the micro-level as well as obtain homogenized results of the model at the macro-level. This type of model enhances the scale-bridging abilities and makes the model more suitable for multiscale simulations.

This detailed FE-model is then used to generate a huge dataset by applying different strain paths to the RVE analysis of the model, which are extracted from the large-scale car crash simulations. A well-structured autonomous method is developed, which runs an individual RVE analysis and stores the corresponding stress strain response in a recursive manner.

This dataset is used to train a data-driven surrogate model. The sliding window approach is employed to generate a more robust dataset for model training purposes. A LSTM based architecture is selected to train the surrogate model which demonstrate poor prediction ability when verified against unseen data. To overcome this issue, a modified version of the LSTM based model is successfully trained using both absolute and incremental strain values as input along with incremental stress values as output. The model is further enhanced by scaling the incremental values as compared to the absolute values to achieve a rough normalization of the entire dataset. Finally, this model exhibits excellent results when it is validated against unseen data and can further be used for implementation in a LS-DYNA co-simulation.

The framework for implementation of the surrogate model along with a LS-DYNA simulation is established with the help of pipes.py (the pipelining framework). With the implementation of the surrogate model, it is further validated against the classical FEM approach for various load cases. The surrogate model demonstrates accurate predictions in case of an implicit simulation along with very low computational effort and time. But the same model fails to perform adequately in case of an explicit simulation. With some improvements in terms of the training data, the model does succeed to provide better results in case of the explicit simulation, but it also highlights the heavy influence of the training data, parameters, and training conditions on the performance of the data-driven approach. This in turn motivates the future research that can be carried out in the direction of self-consistent ML models, that are independent of the training attributes.

Although the current work presents the methodology for use of a single-element model in conjunction with the surrogate model, the concept can be further developed to be used for a model with multiple elements. This advancement will open the possibilities for use of such data-driven surrogate models to be used in large-scale car crash simulation and facilitate optimal battery design procedures.

6 Acknowledgements

This work has been carried out within the research project DigiTain (Digitalization for Sustainability) funded by the Federal Ministry for Economic Affairs and Climate Action (BMWK).

7 Literature

[1] Y. Pan, Y. Xiong, W. Dai, K. Diao, L. Wu & J. Wang: " Crush and crash analysis of an automotive battery-pack enclosure for lightweight design", International Journal of Crashworthiness, volume 27, 2022, pp. 500-509.

[2] H. Wei, D. Lyu, W. Hu, CT Wu: “RVE Analysis in LS-DYNA for High-fidelity Multiscale Material Modeling”, LS-DYNA New Feature and Application.

[3] Z. Liu, H. Wei, T. Huang, C.T. Wu: “Intelligent multiscale simulation based on process-guided composite database”, preprint, 2020.

[4] A. Schmid, M. Raffler, C. Dünser, F. Feist, C. Ellersdorfer: " Two Modelling Approaches of Lithium-Ion Pouch Cells for Simulating the Mechanical Behaviour Fast and Detailed ", 13th European LS-DYNA Conference, 2021.

[5] A. Bao, E. Gildin, J. Huang, E. Coutinho: " Data-Driven End-To-End Production Prediction of Oil Reservoirs by EnKF-Enhanced Recurrent Neural Networks ", SPE Latin American and Caribbean Petroleum Engineering Conference, 2020.