Computational Materials Science 157 (2019) 87–98

Contents lists available at ScienceDirect

Computational Materials Science

journal homepage: www.elsevier.com/locate/commatsci

Reduced-order modeling through machine learning and graph-theoretic approaches for brittle fracture applications

Abigail Hunter[a], Bryan A. Moore[a], Maruti Mudunuru[a], Viet Chau[a], Roselyne Tchoua[a,b], Chandramouli Nyshadham[a,c], Satish Karra[a], Daniel O'Malley[a,d,*], Esteban Rougier[a], Hari Viswanathan[a], Gowri Srinivasan[a]

[a] Los Alamos National Laboratory, USA
[b] University of Chicago, USA
[c] Brigham Young University, USA
[d] University of Maryland, Baltimore County, USA

ARTICLE INFO

Keywords:
Brittle crack
Machine learning
Crack interaction
Reduced-order models
Graph theory

ABSTRACT

Typically, thousands of computationally expensive micro-scale simulations of brittle crack propagation are needed to upscale lower length scale phenomena to the macro-continuum scale. Running such a large number of crack propagation simulations presents a significant computational challenge, making reduced-order models (ROMs) attractive for this task. The ultimate goal of this research is to develop ROMs that have sufficient accuracy and low computational cost so that these upscaling simulations can be readily performed. However, constructing ROMs for these complex simulations presents its own challenge. Here, we present and compare four different approaches for reduced-order modeling of brittle crack propagation in geomaterials. These methods rely on machine learning (ML) and graph-theoretic algorithms to approximate key aspects of the brittle crack problem. These methods also incorporate different physics-based assumptions in order to reduce the training requirements while maintaining accurate physics as much as possible. Results from the ROMs are directly compared against a high-fidelity model of brittle crack propagation. Further, the strengths and weaknesses of the ROMs are discussed, and we conclude that combining smart physics-informed feature engineering with highly trainable ML models provides the best performance. The ROMs considered here have computational costs that are orders-of-magnitude less than the cost associated with high-fidelity physical models while maintaining good accuracy.

1. Introduction

Capturing microstructural information and interaction of micro-cracks is critical for determining the failure characteristics of materials in applications such as spalling and fragmentation of metals [1,2]. Often, performing simulations with all the micro-crack information is computationally unmanageable since, e.g., a $1\,m^3$ domain can have billions of cracks and the problem only gets worse as the domain size increases. One then has to utilize some form of upscaling to incorporate the micro-crack information into the larger continuum-scale simulations. The upscaling process is also computationally very expensive. For each continuum-scale grid cell, one then has to perform hundreds of the high-fidelity simulations at the micro-scale that take several CPU-days to infer the upscaled mechanical parameter along with uncertainty bounds, and so for a domain with many grid cells, the computational cost is enormous. Reduced-order models (ROMs) that emulate the high-fidelity simulations and run in seconds to minutes, are promising replacements for the micro-scale high-fidelity simulators [3]. One way to build ROMs is to use machine learning (ML) and graph-theoretic methods and train on the data generated from a high-fidelity model [4]. Our goal is to replace a computationally-expensive high-fidelity model for crack propagation with an efficient ROM that combines ML and graph theory. Graph theory provides a powerful mechanism for representing and interrogating structured systems. The combination of ML and graph theory is ideal for developing ROMs of cracked systems because graphs inherently consider topology and ML mimics complex behavior.

ML is a field that is rapidly evolving and enables computers to learn based on data [5]. The data can come from a variety of sources including measured or inferred data from physical experiments, simulation data from computer models or a combination of these. There have been successful applications of ML across many fields like natural

* Corresponding author at:Los Alamos National Laboratory, USA.
E-mail address: omalled@lanl.gov (D. O'Malley).

https://doi.org/10.1016/j.commatsci.2018.10.036
Received 19 September 2018; Received in revised form 25 October 2018; Accepted 27 October 2018
Available online 07 November 2018
0927-0256/ Published by Elsevier B.V.

language processing, object recognition, and bio-informatics [6–8].

ML falls broadly into three categories based on the objective of the technique; unsupervised ML, supervised ML and semi-supervised learning. In unsupervised learning, the training data is not labeled, yet the goal is to find underlying structures in the data (such as clusters). Supervised learning is useful when the training data is fully labeled. The goal is to find a mathematical function that maps inputs to outputs. Examples of supervised ML algorithms include random forests (RF), support vector machines (SVM) and artificial neural networks (ANNs) [9–11]. Semi-supervised learning can be thought of as a hybrid method, where one uses a small amount of labeled data with unlabeled data in classification or dimensionality reduction applications. This method is used when the cost of placing labels on the data is prohibitive.

The suitability of an ML technique to a specific application depends both on the goals as well as available data. Methods such as decision trees (DT), RF and SVM are generally more effective on small to medium sized datasets, with a few thousand samples or smaller. In particular, more complex methods such as ANNs tend to overfit when learning from tens to hundreds of training data points. The reverse problem is that of underfitting, when the ML model does not have sufficient complexity to accurately model the functional relationship between the inputs and outputs.

ML is an important part of the emerging field of materials informatics [12,13]. One of the aims of materials informatics is to enable rapid prototyping of new materials on a computer rather than in a laboratory [14]. Early work in materials informatics has enabled advances in batteries [15], materials for capturing solar power [16], and magnets [17]. In order to enable ML methods to perform well, it is often necessary to identify the key descriptors [18,19] or features (as they are often called in the ML community) that can be used to make predictions.

In this research, we have chosen to inform several different supervised ML and graph-theoretic algorithms using crack network evolution in brittle materials simulations from a validated, finite-discrete element simulator HOSS [20]. When ML is used to replace a high fidelity model in a predictive setting, supervised learning is an ideal candidate since the simulator's inputs and outputs are effectively a labeled dataset (i.e., the outputs can be regarded as labels). Here we present, compare and contrast four different methods for reproducing aspects of the high fidelity model. We narrow our comparison and discussion with two key metrics: prediction of the failure pathway and prediction of the time at which failure occurs. These are chosen with the goals to validate the methods, and also highlight which features stand out and must be included in the ML for the method to be successful. A natural way to capture the underlying structure of the crack network is through the use of an graph representation that captures the couplings between cracks [21]. In one such representation, each crack is represented by a vertex in the graph and edge properties in the graph can represent properties of pairs of cracks (e.g., the distance between the two cracks, whether or not two cracks have coalesced, etc.). Our models have chosen to exploit this equivalence in order to learn the topological drivers that influence crack propagation using fundamental concepts such as nearest neighbors.

This work has an eye toward informing models on larger length scales. Ultimately, we aim to inform macro-scale continuum constitutive models with statistical crack information being produced by ML algorithms. Such an approach benefits from the computational efficiency that continuum-scale methods have for modeling large specimens and parts (centimeters and larger), but also has the ability to incorporate sub-scale physics that cannot be resolved in these methods due to computational limitations.

We found that applying ML to predict crack growth in materials can be done by engineering the features that can then be generalized to a wide array of problems. Among other things, damage evolution in brittle materials is strongly dependent on the applied load, particularly the direction and rate of the load applied. To address this problem in crack mechanics, there are three fundamental crack modes: Mode I (opening), Mode II (sliding or shearing), and Mode III (tearing) [22,23]. These three basic crack modes or their combinations can be used to describe any crack mode in a cracked body. In this study, the primary failure mechanism for failure was Mode I, as a result of the purely tensile quasi-static loading conditions. Failure paths are generally perpendicular to the loading direction, and crack propagation is mainly along the horizontal direction.

In Section 2, the set-up of the high-fidelity simulations and the resulting datasets used for training the ROMs are discussed. Details behind the working of the HOSS simulator are also discussed. In Section 3, the algorithms behind four different ML and graph theoretic ROMs are discussed. Validation results of the ROMs is presented in Section 4 followed by a discussion on the comparison of the four ROMs in Section 5. A summary of the study is presented in Section 6.

2. High-fidelity simulations

In this section, we detail the background on the high-fidelity finite-discrete element simulator HOSS along with the problem set-up used and the corresponding dataset obtained.

2.1. Hybrid Optimization Software Suite (HOSS)

For this work, we have chosen to rely on the Hybrid Optimization Software Suite (HOSS) [24–26] for our high-fidelity simulations of an evolving crack network, which will be used to both inform and validate the graph-based ML approaches described in Section 3. HOSS is a hybrid multi-physics software tool based on the combined finite-discrete element method (FDEM). The FDEM approach merges finite element techniques to describe the deformation of the material with discrete element-based transient dynamics, contact detection, and contact interaction solutions, so it can account for both damage evolution, and catastrophic crack or fragmentation. In this section we briefly review some important aspects of HOSS, as a full description of this model is outside the scope of this article. However, we refer the interested reader to the following comprehensive references on the subject for more details: [27–32]. We also note that the work presented here does not depend exclusively on HOSS; other high fidelity modeling approaches that can resolve individual cracks and their evolution could be used for training and validating the ML approaches discussed in later sections.

In the FDEM framework the solid domains, or discrete elements, are further discretized into finite elements. The governing equations are based on conservation of mass, momentum, and energy along with the Newton's laws [27,32]. These equation are solved using an explicit central difference time integration scheme [33], which makes it necessary to use very small time steps to update the system state dynamically. In HOSS's FDEM framework, cracks form along the boundaries of the finite elements. In order to capture fine scale mechanisms, such as crack nucleation, propagation, branching, reorientation, etc., the crack network must be finely resolved spatially, with dozens to hundreds of finite elements along the length of each crack [28]. As a result, the outputs of simulations involving laboratory sized samples with thousands of incipient microcracks can result in petabytes of data. In addition, this need for a highly resolved mesh combined with the explicit time integration scheme can result in a need for high performance computing resources for extended durations in order to model the damage evolution and failure of laboratory sized samples and larger.

The problem of interest for this work (described in more detail in the next section) is a 2D sample under a pure tensile load. HOSS considers two primary modes of failure in 2D: Mode I, which is opening due to tensile load, and Mode II, which is crack growth due to shear loading conditions. Since the problem of interest will be dominated by Mode I crack growth, we focus this discussion on the key details as to how HOSS accounts for Mode I crack growth. However, it must be pointed out that the Mode II crack growth is handled in a similar way to the Mode I case, except different sets of parameters are applied. In addition,

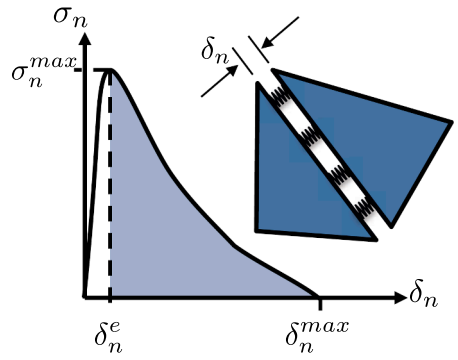

Fig. 1. Schematic representation of how damage is modeled in HOSS. The cohesive points between element edges are shown for Mode I opening. The amount of displacement between the elements, δ_n, is related to the normal stress, σ_n, through a similar curve to that shown schematically. The maximum elastic opening of the spring, δ_n^e, and the maximum (elastic and inelastic) opening of the spring, δ_n^{max}, are noted. If $\delta_n > \delta_n^{max}$ the spring is considered to be broken and the stress at which this occurs, σ_n^{max}, is the tensile strength of the material.

in a HOSS simulation of our pure tension problem, many of the element edges will not be oriented orthogonally to the applied load. Hence, although globally Mode I failure dominates this problem, both shear and opening can occur at a local mesh element scale.

Between the interface of any two finite elements, there are a user specified number of cohesive points (four is used in all simulations presented and discussed here), which are modeled as springs, as shown in Fig. 1. As the two elements undergo tensile load and are pulled apart, the springs within the interface are strained resulting in a small space opening between the elements. Similarly, for shear, or Mode II, deformation, there will be four cohesive points that can deform to allow one element to slide relative to another. A typical behavior of the springs as a function of the opening created between the edges of two finite elements is shown in Fig. 1.

These cohesive points between elements are responsible for representing the material strength response both in tension and in shear. The maximum stress that the spring can withstand in tension is equal to the tensile strength of the material (σ_n^{max}), while the maximum shear stress that the analogous springs can carry is equal to the shear strength of the material. In the first part of the curve ($0 < \delta_n \leqslant \delta_n^e$) the springs follow non-linear elastic behavior where no irreversible damage is accrued. The springs connecting any two finite elements are very stiff in their non-linear elastic response, i.e., $\delta_n^e \ll \delta_n^{max}$ (the stiffness is two orders of magnitude larger than the Young's modulus of the material) in order to reproduce the response of a continuum medium. If the interface between the elements continues to be strained past this elastic limit ($\delta_n^e < \delta_n \leqslant \delta_n^{max}$), the springs enter a strain softening regime which represents the material developing irreversible damage and therefore degrading its strength response. Once $\delta_n > \delta_n^{max}$ the spring is considered to be broken and no longer supports any load. It is worth noting that the area below the softening portion of the curve, indicated as the shaded region in Fig. 1, represents a strain energy density, i.e., energy per length, dissipated during the crack process. Fig. 1 only presents a schematic representation of this curve. The actual shape of the curve is found through fitting to experimental results that describe softening in geomaterials [34]. In this work we only consider geomaterials, hence plastic deformation is not considered as part of the inelastic material response, however, such behavior could be incorporated into the curve shown in Fig. 1.

2.2. Problem definition, set-up, and dataset

Since the primary goal of this work is to compare several different ML and graph-theoretic methods on equal footing, we have chosen to generate a single dataset for use by all of the methods. A schematic description of the problem of interest is shown in Fig. 2. A 2 m × 3 m concrete specimen in 2D is loaded in pure tension, hence we expect that Mode I failure will be the dominant material response in all simulations. Within the sample, there are 20 initial cracks each with an initial length of 30 cm and three different orientations, 0, 60 and 120 degrees with respect to the bottom of the sample. The material is pulled from the top at a constant velocity of 0.1 m/s, and the bottom boundary is fixed. The simulations were run for 700,000 time steps, with a time step of 10^{-8} s. Results from HOSS were output every 2000 time steps, providing 350 output files of data per simulation. Each HOSS simulation took about 4 h of computational time on 400 processors. The sample is considered to have failed when a single crack path connects two opposite boundaries of the material i.e., a crack spans the entire width of the sample. At the point of failure, the material is unable to bear further load, and we consider this catastrophic failure. From a material science standpoint, this simulation configuration imposes similar conditions as a fundamental tension test. Tension testing has been widely used for material testing, particularly for measuring many material properties including Young's modulus, Poisson's ratio, ductility, yield strength, etc. [35].

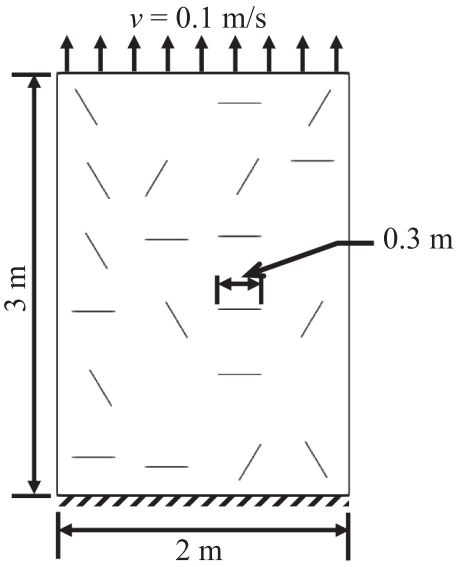

Fig. 2. Schematic example of a HOSS simulation setup for the problem of interest. Overall 185 HOSS simulations were completed in which the location and orientations (0, 60, or 120 degrees) of the initial cracks are randomly chosen for each simulation. All other geometry, loading, and material parameters were kept the same for all simulations.

The material is assumed to be elastically isotropic in all simulations and the material parameters and boundary conditions used for the concrete sample are: material density (ρ): 2500 kg/m^3; Young's modulus (E): 22.6 GPa; Shear modulus (G): 9.1 GPa; Poisson's ratio (ν): 0.24166; Ultimate Tensile strength (σ_U): 4.0 MPa; applied velocity (v): 0.1 m/s at $y = 3.0$ m and 0 at all other boundaries.

Using HOSS, we have run a suite of 185 simulations each producing ~15 GB of output. In each simulation the location and orientations of the initial cracks are randomly chosen. Hence, the crack network evolution will be different for every simulation, yet the overall material response should be nominally the same. 150 of these simulations were used as training and testing data for the ML algorithms discussed in later sections. The remaining 35 were used as validation for the ML algorithms in a holdout validation approach. We note that the 35 simulations that were used for validation were not provided the co-authors who were training the models until after their models had been trained. Of these 35 simulations, 25 failed during the simulation, and 10 have substantial crack propagation but have not yet failed. For the 35 simulations in the validation dataset, the damage accumulation over

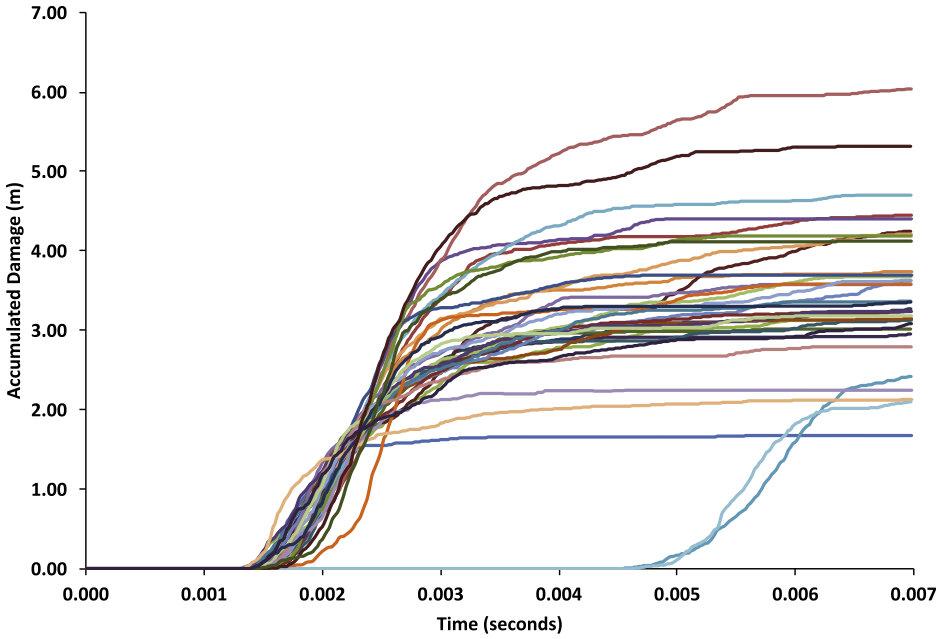

Fig. 3. Accumulated damage versus simulation time for the 35 simulations in the validation dataset (25 that fail after 0.007 s and 10 that have not). Accumulated damage is calculated by determining the amount of distance each crack has propagated every time step and summing to obtain the total distance of new crack.

time as calculated with HOSS is presented in Fig. 3. Damage accumulation is calculated by summing the growth of all propagating cracks within the simulation. The first thing of note in Fig. 3 is that crack propagation does not begin until after some period of loading (typically, about 0.0015 s in this case). During this time, the material is loading elastically (i.e., δ is growing from zero to δ_n^e in Fig. 1). Later in the results section (Section 4), we will see that this time delay until crack growth is difficult to predict when training the ML algorithms unless crack interactions are accounted for. This is illustrated through one of the metrics for comparison, time at which failure occurs.

Another metric we will use for comparison of the ROMs presented in the next section is the location of the failure path. Fig. 3 shows that the cracks start to propagate very quickly and they level off. The majority of simulations level off near 3 m, which is the width of the sample. As we would expect with a low strain rate, Mode I failure situation, a single large crack will evolve and dominate the growth response. Of course other cracks will have some propagation, but in most cases for this quasi-static loading scenario failure is driven by a dominant crack pathway. The ML approaches presented in the next section are able to make a prediction regarding the location of this dominant crack path.

3. Reduced-order models

In this section, we compare four different approaches to modeling the high-fidelity simulations presented in the Section 2, using ML and graph theory. Each of the following methods incorporate some physically based assumptions *a priori* in order to train the ML approaches to model crack network evolution. The first of these assumes that the component of the crack that is orthogonal to the loading direction is the primary driver of Mode I damage growth, resulting in a method in which the formation of the graph is based on the horizontal (orthogonal to the loading conditions presented above) projections of the cracks. This is followed by an approach that assumes that the pairwise interactions between micro-cracks is the primary driver of damage evolution. In this case vertices are placed at the center of each crack and edges represent coalescence. In the remaining two approaches, the graphs are constructed so that vertices represent crack tips and edges represent the surfaces of the crack. In addition, both of these methods are based on the assumption that there is a process zone present at the crack tips, which is a region of localized stress that can result in crack growth. The first of these methods uses the idea of a process zone to identify regions of probable catastrophic failure, while the other uses ML to inform the parameters for a simple constitutive model based on the idea of elliptical process zones present at each crack tip. We will summarize the key details of each method, highlighting the key differences between them.

3.1. Orthogonal Projection (OP) approach

This method tracks the orthogonal projection of oriented cracks, which for the loading conditions discussed here is the horizontal projection. This simplifies the problem by replacing cracks of varying orientations (but fixed lengths) with cracks of varying lengths (but fixed orientations). However, this simplification assumes that the Mode I failure mechanism has a dominant dependence on the perpendicular component of the cracks, and the additional effect randomly oriented cracks contribute to the evolution of the crack network can be approximated with ML algorithms.

Unfortunately, neglecting the orientation dependence also presents some limitations. One example of this is the ability to predict the failure path, particularly features such as crack branching and crack interaction with vertical neighbors. Since all cracks are horizontal and only grow in the direction orthogonal to the applied loading, the predicted failure path will always be a straight horizontal crack spanning the width of the sample, which is not a realistic failure path. Despite this limitation, the time to failure can still be predicted and the predicted failure path can still provide information about the likely region of failure in the sample.

A graph is formed by taking the orthogonal (horizontal) projection of the initial micro-cracks provided from HOSS. Nodes are assigned to each end of the newly defined horizontal cracks, which now vary in crack length rather than crack orientation. Edges are defined between each pair of vertices that are part of the same crack, completing the initial graph. Henceforth, the graph representation of the crack network only grows cracks in the horizontal direction, relying on ML algorithms informed with training data from HOSS to account for any differences in the accrued damage due to eliminating the orientation dependence.

In the OP approach, the change in crack length *da* (for a crack of

length $2a$) determined through the motion of each crack tip over time is the feature that is predicted and evolved using ML algorithms. First, training data is collected from HOSS simulations. At every time step, the amount a crack tip moves is calculated and if oriented non-orthogonally to the loading direction a horizontal projection is calculated to determine *da*. Polynomial interpolation regression (PIR) was the ML model trained and then used to predict *da*. PIR approximates a function with a polynomial of degree *n* by using ridge regression, more specifically by building a Vandermonde matrix akin to a polynomial kernel [36]. We used the scikit-learn [37] implementation of this model. In addition to the validation that was performed using 35 simulations that were held out for validation purposes, additional validation was performed during the model development process on the 150 simulations using a leave-one-out cross-validation approach.

OP Model Summary:

1. Tracks the growth of the horizontal projection of oriented cracks to reduce the problem complexity.
2. Learns and predicts the change in crack length *da* based on *a* using polynomial interpolation regression (PIR).

3.2. Micro-Crack Pair Informed Coalescence (McPIC) approach

The McPIC model assumes that crack interaction and coalescence is the dominant mechanism guiding the overall material response. Again due to the computational expense of accounting for discrete cracks and also the analytical complexity of describing multiple cracks, many continuum-scale models do not account for crack interactions at all [38,39]. Such an assumption will impact the overall material response, and in particular can result in longer failure times than those calculated with models that neglect crack-to-crack interactions. Extracting neighboring pairs of micro-cracks incorporates both the damage accumulation due to crack growth (via coalescence) and the local interaction effects between nearby cracks, connecting these mechanisms to predictions for the global material response.

In the McPIC approach, we isolate all pairs of neighboring micro-cracks and utilize their properties and interactions as training data for the ML algorithms. Nodes are placed at the midpoint of existing micro-cracks and edges are formed when two cracks coalesce. Each micro-crack pair has a feature vector including geometric information such as, the length of each crack, their respective orientations, and the distance between the two micro-cracks. When extracting training data from the HOSS simulations, cracks must be tracked over time to determine when coalescence occurs and when new crack pairs form. A schematic of the coalescence classification process is shown in Fig. 4 for both a non-coalescing crack pair and a coalescing crack pair. The variables included in the feature vectors are also shown.

Additionally, for the purposes of the ML algorithm, the boundaries of the sample are treated as cracks. This is necessary for defining catastrophic failure, which is when crack pairs have coalesced to form a cracked surface spanning the entire sample width. This can only occur if a crack pair, in which one crack is a boundary crack, coalesces. Hence, the minimum distance from a crack to the nearest boundary, D_B, is included in the feature vector. These boundary cracks are defined to have zero length, an orientation of 90 degrees and the minimum distance to boundary is set to the distance from the one interior crack to the boundary. Treating the left and right boundaries as cracks, each simulation yields 190 unique pairs of cracks. Geometric information about the crack network, and the minimum distance of a crack pair to a boundary can all be extracted from provided initial conditions. Whether the cracks within a crack pair coalesce, and when, are the labels for those features. The latter information is extracted from the HOSS training simulations and used to train the ML algorithm.

For the results presented here, ANNs are the ML algorithms used in the McPIC approach. One concern about using a large network is that if it has more parameters than training datapoints it is prone to over fitting.

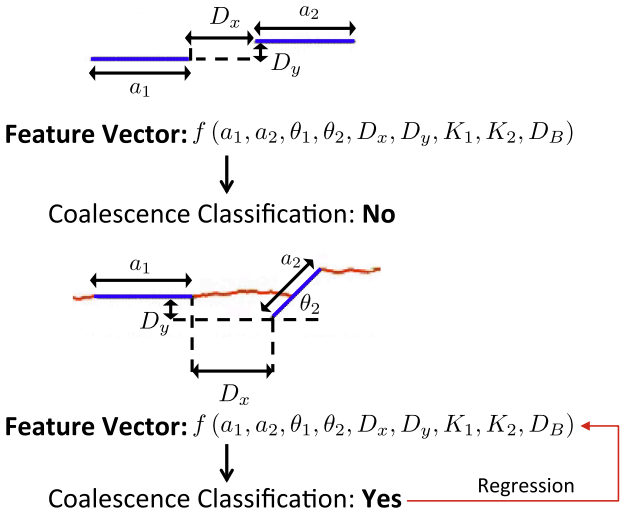

Fig. 4. An example of a non-interacting crack pair and a coalesced crack pair. The feature vector includes the crack lengths (a_1 and a_2), crack orientations (θ_1 and θ_2), the x and y distances between the cracks (D_x and D_y), the stress intensity factors at each crack tip (K_1 and K_2) and the minimum distance to the boundary from either crack (D_B). The simulation above has 20 initial cracks which yield 190 data points (with boundary crack-pairs considered).

The particular ANN used here was relatively small with four layers of 12 neurons, 8 neurons, 4 neurons and 1 neuron in each layer. Decreasing the number of neurons from one layer to the next is common practice in ANN creation. This structure siphons the number of inputs gradually without large amounts of data loss at any particular stage.

Using ANNs, we are able to make predictions about the materials path to failure (i.e., a sequence of cracks that coalesce amongst each other as well as with the left and right boundaries), and the time that elapses before the material fails. Data points were first trained on ANN classifiers which predicted whether crack pairs coalesce. All cracks that coalesce were trained on ANN regressions to predict how much time elapses before coalescence occurs. Thus, feeding the trained classification and regression networks a set of initial crack pair features would output which cracks coalesce and how long it takes them to coalesce. The time to material failure can then be determined by computing the earliest time at which a crack path that includes both the left and right boundary occurs. We note that this method has also been applied using RFs and DTs in place of the ANNs used here. During the model development process, the 150 simulations that were used for training were randomly split between true training data and testing data with 80% of the simulations being used for training and 20% of the simulations being used for testing. We emphasize that the 35 simulations that were used for validation purposes were not involved during the model development. Further details on these methods and the comparison across ML algorithms can be found in [40].

Crack pairs are unique and chosen according to their nearest neighbors (Euclidean distance from one crack to another). Thus a single micro-crack can be a part of multiple crack pairs. This differs from many of the other approaches presented in this section because it does not solely focus on horizontal interactions and propagation. Rather, cracks can interact and propagate with neighbors that may be above (vertical propagation) or, more likely, at an angle (combined horizontal and vertical propagation) to any one crack. Another potential advantage of this method is that if the pairwise interaction between cracks is modeled accurately, the model could, in principle, be applicable regardless of the sample size, number of initial cracks, or variations in length or orientation distribution, as long as the loading conditions remained consistent. In addition, it would be expected that the model would extend to other materials within the same material class (i.e., brittle, non-metals). This crack pair data implicitly contains a wealth of

information related to crack interactions, crack propagation speeds, and the influence of orientations on crack dynamics, yet is much more concise than the detailed information needed to perform the full HOSS simulation (e.g., high-resolution stress fields, nodal velocities, etc.).

McPIC Model Summary:

1. Focuses on crack-pair coalescence and the corresponding time for coalescence to occur. There are two separate but similarly constructed ML models for detecting if there is coalescence and when it occurs.
2. Feature vectors for each crack pair includes information such as the length of each crack, their respective orientations, the distance between the two micro-cracks, the minimum distance to the boundary from either crack, and the stress intensity factors at the crack tips.
3. Uses neural networks for the ML algorithm, but can also utilize random forest or decision tree approaches [40].

3.3. Network-based Fracture Process Zone (NFPZ) approach

In this subsection, we provide a brief description of the network-based crack process zone (NFPZ) approach to obtain the most probable failure pathways for dynamic crack propagation under tensile loading. This model is based on the idea that a process zone (PZ) exists at each crack tip, where vertices are placed. The PZ is a region surrounding the crack tip where a stress concentration resides resulting in damage accumulation as a crack propagates over time [41,42]. Since the stresses in the PZ are very high, micro-cracks are formed in the vicinity of the crack tip [43,44]. Over time, as the crack advances, these micro-cracks merge and become a single entity to give continuity to the already existing crack. Edges between vertices represent the crack surface, and new edges are formed when the micro-cracks coalesce forming larger cracks. The PZ bridges the cracked and uncracked regions. Failure of brittle and quasi-brittle materials under tensile loading typically starts with the development of a PZ around the crack tip [45,46], hence this approach assumes that the most probable path for failure directly correlates with where there are high concentrations of PZs.

In order to identify regions in which failure is likely to occur, we first identify the cracks that are orthogonal to the tensile loading. Based on the initial conditions, this corresponds to all of the 0 degree angle cracks. Since these cracks are best oriented for Mode I opening under the loading conditions, the model anticipates that they will grow the fastest. Hence, we hypothesize that the likely pathway for the ultimate failure pathway will include one or more 0 degree angle cracks. For each tip of these cracks, we identify the nearest neighboring crack which could interact or coalesce with it. This nearest neighbor could have any orientation.

Interaction and coalescence occurs if the PZs of two neighboring cracks overlap. The size of the PZ, d_{12}, is given as follows [47]

$$d_{12} \propto \left(\frac{\sigma}{\sigma_y}\right)^2 \left(a_1 + a_2\right) \tag{1}$$

where σ and σ_y are the applied and yield stresses of the material, and a_1 and a_2 are the crack lengths. For these simulations, it was learned that the size of the PZ, d_{12}, to be ~30% of $(a_1 + a_2)$, that is, $d_{12} \approx 0.3(a_1 + a_2)$. Hence, if two neighboring cracks fall within this size of PZ, we assume that they are going to coalesce to form a larger crack.

Once the PZs and potential coalescing cracks have been identified, we next identify the zone(s) of failure. Potential failure zones are regions corresponding to a probable failure pathway, and it is expected that the specimen is going to fail catastrophically in one of these zones. These zones contain a set of pre-existing crack-pairs that have been identified for potential coalescence through calculation of the PZs. Through this interaction, there is a strong possibility to form large cracks. The large cracks that are formed following coalescence are more favorable to propagation compared to smaller cracks [48].

There may be one or more potential failure zones in a sample, however only one of these pathways will actually correspond to the sample's catastrophic failure. Next, we look for weighted shortest paths connecting the sides of the domain that are parallel to the tensile loading direction. We impose a constraint that the paths have to traverse through the set of PZ-based large-cracks comprising the failure zones. Pre-existing cracks are given low edge weights and newly formed cracks are given edge weights based on the Euclidean distance. To compute all possible shortest paths for our weighted graph, we use Dijkstra's algorithm [49].

Further details on this model can be found in [50].

NFPZ Model Summary:

1. Identify the set of pre-existing cracks that fall within the PZ and connect them.
2. Identify the failure zone. This corresponds to the region which contains larger cracks formed after connecting the pre-existing cracks that are in the PZ.
3. Find the likely failure pathways using Dijkstra path algorithm.

3.4. Ellipse Process Zone (EPZ) approach

The final ML approach presented here is the Ellipse Process Zone (EPZ) model. This model utilizes the assumption of an ellipsoidal PZ present at crack tips, similar to the NFPZ model just discussed. However, in this approach, a simple constitutive model is developed based on heuristics obtained from observing 2D HOSS simulations. We initially assume that the PZ depends on the magnitude of stress intensity at each crack tip. The stress intensity factor is a physical quantity used in crack mechanics to predict the stress state ("stress intensity") near the tip of a crack caused by a remote load or residual stresses [51]. To quantify the relationship between stress intensity and material parameters we introduced the crack growth function (C^f). Our hypothesis is that if we are able to learn this crack growth function with ML algorithms we can mimic a 2D HOSS simulation. We formulate the crack growth function in terms of the material parameters as follows,

$$C^f \propto \frac{D_y v E \sqrt{a} \cos^2(\theta)}{h w \sigma_U \rho D_x}. \tag{2}$$

The above relation is based on heuristics which relate the crack growth function, (C^f), at a crack tip to different material properties. C^f is observed to be directly proportional to Young's modulus (E), the square root of the crack length (a), the squared cosine of the orientation angle (θ) of the crack taken with respect to the x-axis, the resultant velocity due to load applied on the material (v), and vertical distance of the crack tip from the side the load is applied (D_y). It is observed to be inversely proportional to the material density, ρ, the nearest horizontal distance between crack tip and edge (D_x), height (h) of the material, width (w) of the material, and ultimate tensile strength (σ_U). Our goal is to improve the crack growth function using an ML model. In this work we have initially assumed the proportionality factor to be 1.

In the EPZ model, we represent the initial crack configurations in the 2D material using graph theory principles wherein each crack tip is described as a vertex and each crack as an edge in graph. The graph theory principles in the code are implemented using the NetworkX package [52]. The crack growth function is calculated at every crack tip or vertex in the graph at every time step, indicating a propensity for crack growth. The dynamic crack propagation is simulated using an algorithm also developed from heuristics. The optimal parameters of the algorithm and model are learned using least squares regression technique as implemented in the SciPy [53] package. The EPZ model is computationally cheap and predicts the time and path of failure.

Nodes with a greater propensity for crack growth correlate to crack-tips with larger PZs. In addition, crack tips that have a large (small)

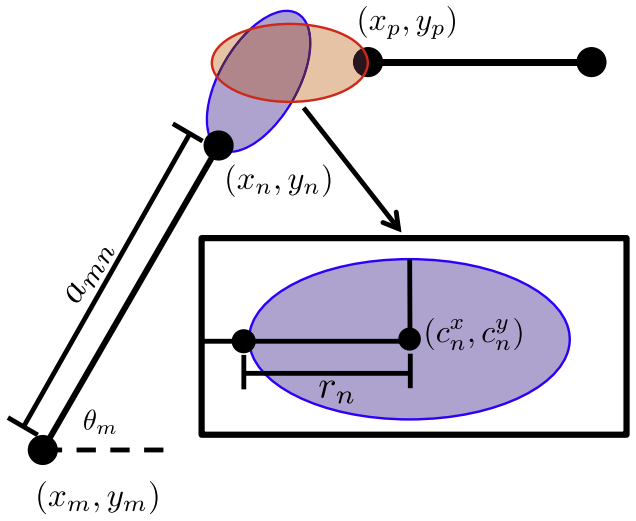

Fig. 5. A schematic representation of two cracks and their overlapping PZs as described by the EPZ method. Since the elliptical PZs at vertices n and p are overlapping, these two cracks are expected to coalesce.

propensity for growth are included (discarded) from the EPZ calculation. By not including vertices that are unlikely to grow, we can further reduce the computational expense of this approach. For results presented in this work, the cutoff is defined as vertices with a normalized propensity factor, $\overline{P}_k$ that is greater than the average of the maximum C^f and mean value of C^f. This definition of cutoff was obtained after optimizing the results of EPZ model with many training cases of HOSS simulations. Using C^f, we can calculate the length of the semi-major axis (r_n), for the ellipse connected to vertex n as:

$$r_n = \overline{C}_n^f a_{mn} \gamma, \tag{3}$$

where the subscripts indicate the vertex in the graph representation, a_{mn} is the crack length spanning vertices m and n (see Fig. 5), and γ is a time-dependent scaling parameter which is learned during training. Once we determine the length of the semi-major axis, the size and location of the ellipsoidal PZ can be calculated assuming that one vertex is the crack tip (i.e., a vertex).

Every vertex is associated with another vertex through the connection by an edge, as shown in Fig. 5. This is representing the crack with two tips at points $(x_n,\ y_n)$ and $(x_m,\ y_m)$, which indicates the location of vertices n and m that share an edge. Next, to determine the location of the center, $(c_n^x,\ c_n^y)$, of the ellipse associated with vertex n, we first compute a normalized vector ($\mathbf{N}$) in the direction of orientation $\theta = \theta_n = \theta_m$ of the crack. The vector $\mathbf{N}$ is computed as:

$$\mathbf{N} = \frac{(x_m, y_m) - (x_n, y_n)}{a_{mn}} \tag{4}$$

The center of the ellipse located at a distance r_n from the crack tip associated with vertex n is then calculated as,

$$(c_n^x,\ c_n^y) = (x_n, y_n) - \mathbf{N} * r_n. \tag{5}$$

Using the center points and semi-major axis length, we can calculate the ellipse PZ region using the equation for an ellipse given by,

$$\frac{((x_n - c_n^x)\cos(\theta_n) + (y_n - c_n^y)\sin(\theta_n))^2}{r_n^2} + \frac{((x_n - c_n^x)\sin(\theta_n) + (y_n - c_n^y)\cos(\theta_n))^2}{(2 * r_n(1-e))^2} = 1, \tag{6}$$

where e is the eccentricity of the ellipse optimized using the SciPy [53] optimization toolkit during training and θ is the orientation of the crack with respect to the x-axis in the global coordinate system of 2D. We note that the training data was not split between training and testing sets for this method. The holdout validation was the only means of testing and validation. The orientation, θ is a dynamic quantity and is computed at each time step. In order to determine the orientation θ for a given crack tip located at $(x_n,\ y_n)$, first we check at present time step if there is any other crack tip within the elliptic PZ of crack n. If so, we assume that the two crack tips move towards each other. If so, then we propagate the tip at $(x_n,\ y_n)$ toward the other tip at, say $(x_p,\ y_p)$. Then the orientation, θ_n of the crack tip $(x_n,\ y_n)$ is given as,

$$\theta_n = 180° - \frac{180° \arctan\left(\frac{y_p - y_n}{x_p - x_n}\right)}{\pi} \tag{7}$$

If the orientation is greater than 45 degree, we propagate the crack tip with the averaged orientation based on lengths of cracks corresponding to crack tips $(x_n,\ y_n)$ and $(x_p,\ y_p)$. This is given as,

$$\theta_n = \frac{a_{mn}\theta_n + a_{pq}\theta_p}{2}, \tag{8}$$

where $a_{mn},\ a_{pq}$ are the crack lengths and $\theta_n,\ \theta_p$ are orientations corresponding to the crack tips at $(x_n,\ y_n)$ and $(x_p,\ y_p)$. In this way, a crack can reorient in several time steps to coalesce with a nearby crack. If there are no overlapping elliptic PZs, then the crack is assumed to propagate orthogonal to the loading conditions (horizontally in the case discussed here). The algorithm for orientation assumes Mode I failure, so cracks are expected to propagate orthogonally (horizontally to the vertically applied load in this case) to the applied load. However from many training examples, we observed that if the crack's orientation deviates too far from the orthogonal direction, it will curve over several time steps until it can start propagating in direction perpendicular to the applied load. To account for this, we propagate these cracks with half of the current orientation until the crack is oriented horizontally.

Once the elliptic PZs are determined, we can look to evolve the graph. The propagation is done by creating a new daughter vertex at a distance of $dL_n \propto (1 + a_{mn})$ from crack tip $(x_n,\ y_n)$ in the orientation as described above. The constant of proportionality are determined during the training. Once the distance and orientation for crack propagation is determined, a new vertex is created at that location, and the parent vertex becomes inactive. In the case of coalescence, both vertices become inactive and no new vertices are created. In order to detect failure time, at every time step, we check if any vertices intersect the boundaries of sample and if there exist a direct path between two opposite boundaries of the material.

EPZ Model Summary:

1. Developed based on heuristics obtained form observing HOSS simulations.
2. Based on physical constitutive relationships.
3. Uses least squares optimization techniques to learn from the data.

4. Results

For comparison of the four approaches introduced above we have chosen two metrics. The first is prediction of the failure pathway, and the second is prediction of the time at which failure occurs. All the previously discussed methods can provide results for the first metric. However, NFPZ cannot predict the time to failure. These metrics were chosen because they can be validated with HOSS results to determine the efficacy of each approach. In addition, we can compare and contrast methods in order to investigate which physical assumptions lead to the best results and which variables are best modeled with the ML algorithms.

Once an ML method is shown to produce valid results it could potentially be used to inform material models, particularly those at macroscale continuum length scales where it is computationally intractable to resolve every individual crack and their evolutions. In order to inform continuum-scale constitutive models, statistical information, such as probability density functions, that describes damage accumulation over time would be needed. For this purpose, exact locations of the failure

pathways and the coalesced cracks becomes less important since the discrete crack network cannot be accounted for in these types of models. Rather, the problem set-up described in Section 2.2 would, perhaps, be representative of a single cell or element in a large scale finite element or hydrocode model. Predicting when failure occurs and how much damage has accumulated will be important to accurately predict the overall material response, which includes accounting for degraded material properties as damage accumulates (i.e., the softening curve) and when and where failure of the specimen or part occurs (i.e., which elements or cells will fail first). Providing such information is the subject of current and future research, although our second metric (prediction of failure time) is directly relevant to this goal.

4.1. Predicting the dominant failure path

Of the 35 simulations in the validation suite, 25 achieved full material failure at the end of the simulation and 10 did not. A successful prediction of the failure pathway is defined when the initial cracks within the crack spanning the sample are the same in both the HOSS calculated failure path and the path predicted by an ML approach. In the results presented here, partial matches are not considered correct or given any credit. In other words, if an ML approach predicts an additional initial crack in the failure path or only includes some but not all initial cracks that the HOSS result shows, it is considered an incorrect prediction of the failure path. We note that including partial matches would significantly improve the statistics, however these are not included. In particular, the McPIC and EPZ methods not only predict growth and coalescence of the proposed failure path, but also propagation of other cracks within the system. If the overall damage accumulation was considered, these two methods are able to predict potential matches for propagation and coalescence outside of the crack growth that directly leads to material failure.

The dataset has been divided into three categories: simulations that have achieved full material failure without the dominant crack pathway branching (20 simulations total), simulations in which the material failed through crack branching of the dominant crack pathway (5 simulations total), and simulations that did not achieve full material failure (10 simulations total). In the last case, the failure path is defined as the crack path that encompasses the largest percentage of the sample width. Fig. 6 shows a representative example for each of the three categories. The first row, Fig. 6(a)–(d), present results for a simulation in which the material failed without any crack branching. The second row, Fig. 6(e)–(h), shows an example of a simulation where the material sample is near failure, but the dominant crack pathways did not fully coalesce before the end of the simulation. Finally, Fig. 6(i)–(l) in the third row show an example of a sample that failed with a pathway that branched about halfway across the sample. The first column (Fig. 6(a), (e) and (f)) show the HOSS results for each of the three different simulations. The remaining three columns present the predicted results from the McPIC, NFPZ, and EPZ methods, respectively. Figures from the OP method are also not included, since the failure pathway predictions will always consist of horizontal lines spanning the sample width.

The first row of figures, Fig. 6(a)–(d), is perhaps the most straightforward of the three cases showing a failed sample via a pathway that does not include any crack branching. Figures from the McPIC, NFPZ and EPZ methods show that all three of these approaches correctly predict the failure path. Although no figures are shown, the OP approach is also successful in predicting the failure path for this simulation. It is also worth noting that the McPIC and EPZ methods also capture the additional propagation of the crack outside of the dominant failure path.

In the second row, Fig. 6(e)–(h) show images corresponding to a sample that has not yet fully failed. In this case, the McPIC, NFPZ and EPZ methods all predict that the sample has failed even though the HOSS results show that this is not the case. In nearly all of the 10 simulations in which the sample has not yet failed, all the approaches predict early failure. This could indicate that there is a key feature missing in these algorithms related to the crack growth rate. Of the three images, the most accurate prediction of the likely failure path is produced by the NFPZ approach. The McPIC approach also produces promising results for this case. The EPZ approach predicts failure along the bottom of the sample, which is incorrect in this case. However, this method does show propagation throughout the sample, much of which corresponds to propagation seen in the HOSS results. The EPZ approach also produces crack evolution that best reflects crack curvature seen in the HOSS results.

Finally, Fig. 6(i)–(l) present a sample that failed via crack branching. Of the three methods shown in the figure, the NFPZ method best captures the failure path, but does not account for any crack branching behavior. It is worth nothing the while the McPIC did not correctly predict failure of the sample or the location of the branching, it does predict a branched configuration. Crack branching highlights the importance of accounting for crack-to-crack interactions. Since the McPIC method has focused training resources toward all possible crack pairs it is more likely to capture branching mechanisms. The incorrect placement of the branch may highlight a need to consider more that just pairwise interactions between the cracks contained within the crack network.

The overall number of correct predictions made by each method for each of the three categories are shown in Table 1. From Table 1 we see that the failure path in simulations in which the material fully failed without any crack branching were most often predicted correctly across the methods, with the OP and McPIC methods providing the most correct, exact matches with the HOSS simulations. It must be noted that the criterion for a correct match was very strict, expecting better than human performance. Full credit was awarded when only the exact cracks predicted by HOSS were replicated by the ML algorithms, and no partial credit was given when most of the path was identified correctly (e.g. correctly except with one crack missing, or including an extra crack). It must be noted that for several of these cases involving branching, the human eye proved quite unreliable in exactly pinpointing the failure path. Given these harsh metrics for success, the fact that the most successful methods predicted the failure path 45% of the time was encouraging.

While the number of correct predictions seems low, we again note that these are only perfect matches with respect to the HOSS results. Partial matches would improve this number, so these values act somewhat as a lower bound in accuracy. Fracture networks and failure pathways can be extremely complex, and exact matches can be quite difficult to achieve. While this was a highly relevant first step towards emulating the expensive HOSS simulations in a fraction of the time, it must be noted that statistics derived from these ML predictions including crack length distributions and overall damage value are aggregated or averaged quantities, and hence more forgiving in their computation. These results do provide us with valuable insight as to which methods are best capturing crack propagation and coalescence, and also indirectly which mechanisms or features are important to include in the ML training information (see Section 5).

Table 1 also shows that there was not a single exact match for simulations where the material failed via a branched failure path across all the proposed methods. Crack curvature and branching is difficult to determine accurately. Of the proposed methods, only the McPIC and EPZ would be capable of capturing these failure path configurations. The OP method only considers horizontal propagation of cracks in the material system, and the NFPZ can only predict a single path not a branched structure (a tree in graph theory parlance).

The final category in Table 1 refers to the 10 simulations in the validation dataset that did not achieve full material failure. In this case, the McPIC method outperformed the other methods, accurately predicting the dominant crack pathway 50% of the time. In the case of nearly failed samples, the ML approaches often underpredict the failure time. In other words, the ML approaches predicted a fully formed

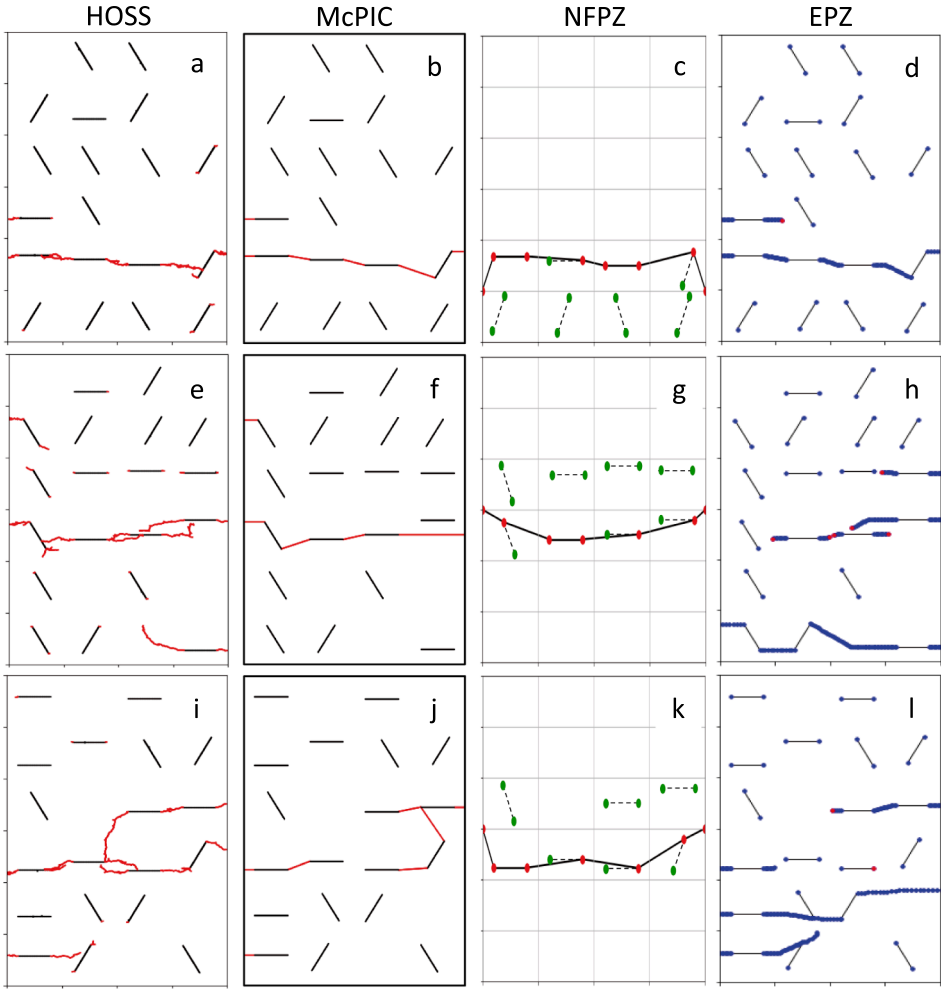

Fig. 6. Three representative simulation cases showing the HOSS results (Figures (a), (e), and (i)), and the results predicted with the McPIC (Figures (b), (f), and (j)), NFPZ (Figures (c), (g), and (k)), and EPZ (Figures (d), (h), and (l)) approaches. The first row of figures ((a)–(d)) shows an example of a simulation in which the material fully failed. Figures (e)–(h) shows an example of a simulation in which material failure was not achieved. Finally the third row, Figures (i)–(l), are representative of simulations in which the sample failed through crack branching.

Table 1
The number of correct failure path predictions for the OP, McPIC, NFPZ, and EPZ methods. A correct prediction of a HOSS failure pathway is defined when the initial cracks within the dominant crack spanning the sample are the same in the HOSS calculated failure path and the path predicted by the ML approaches. Only exact matches are considered as a correct prediction, although the McPIC and EPZ approaches in particular can also predict other crack propagation. Samples that failed with branching crack pathways are separated into a category since the were the most difficult pathways to accurately predict. When the HOSS simulation is not fully failed, the failure path is defined as the dominant crack, which is the crack pathway that spans the largest amount of the sample width.

Description	Number of simulations	OP	McPIC	NFPZ	EPZ
Failed samples without branched pathways	20	9	9	6	4
Failed samples with branched pathways	5	0	0	0	0
Samples that did not fully fail	10	2	5	2	1

failure path when the HOSS simulation shows that the sample has not yet completely failed. This can be seen in the example simulation comparisons for the McPIC, NFP, and EPZ methods in Fig. 6(e)–(h).

4.2. Predicting time to failure

Three of the methods, namely OP, McPIC and EPZ, can also predict the time at which failure occurs. Fig. 7 shows both a scatter plot and histogram of the predicted failure times in comparison to those calculated with HOSS. Results are shown for only the 25 simulations that fully failed in the dataset. As noted in the previous section, failure time was underpredicted in nearly all cases for the 10 simulations that did not fully fail.

Fig. 7(a) shows rather narrow bands for the failure time prediction from the OP and EPZ method. Essentially these two methods have learned a typical time for failure to occur in these types of tensile samples with these loading conditions. This is confirmed in Fig. 7(b), which shows that these two methods predict the failure time for nearly all simulations to be very close to the average failure time calculated by HOSS. It is worth noting that all of the OP and EPZ predictions slightly underpredict the average failure time, with the EPZ method showing a slightly broader spread than the OP method. The McPIC method shows a much wider spread, and only fails to capture failure times at the two extremes of the data, which is highlighted in the histogram in Fig. 7(b). Fig. 7(a) shows that the results produced by the McPIC method follow the parity line.

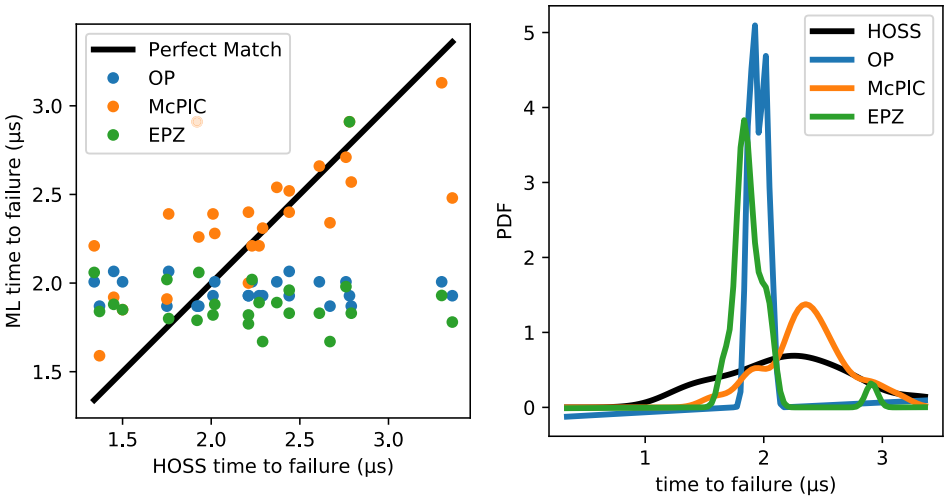

Fig. 7. Predicted failure times from the OP, McPIC, and EPZ approaches compared to the failure times calculated with HOSS shown as (a) a scatter parity plot and (b) a histogram. Note that the ML algorithms were not trained to predict the time to failure, so the accuracy is lower than what one would typically expect from an ML algorithm that *is* trained to predict time to failure. However, we use this as a sanity check that the algorithms are making reasonable predictions.

5. Discussion

Of the methods presented, the McPIC shows the most promising results for both failure path and time to failure predictions. This is the only method that considered crack pairs as the unit of interest, highlighting the importance of capturing these interactions. The method not only had a relatively high number of correct exact predictions, but also shows the ability to capture coalescence outside of the dominant failure pathway and crack branching behavior. This is the only method that did not purely consider geometric factors, but also included crack intensity factors as variables in the feature vectors for the ML algorithms to train with. This likely impacted both the crack propagation and pathway behavior and, subsequently, the time at which failure was predicted. We expect that the model would see improvements if tertiary (or higher order) crack interactions were considered rather than just pairwise interactions. The initial pre-existing cracks are relatively close to one another, hence higher order interactions may have a notable impact on the growth rate of the crack network. If this were explicitly accounted for in the algorithm, improvement in the accuracy of predictions would be expected. However, a larger training dataset may be required to train such a model effectively.

The OP method also produced good results, particularly in calculating the failure pathway. For this metric, the OP model was only outperformed by the McPIC method in the case of predicting the dominant crack pathway in simulations where the sample did not fail. It also only slightly underpredicted the time at which failure occurs. These results essentially show that accounting for the orientation dependence with varying crack lengths is a reasonable assumption, as long as average times for failure are adequate for the application. However, this assumption is only applicable for Mode I failure, and would not be acceptable for shear or combined loading conditions. If interactions with neighboring cracks above and below the horizontal growth path were accounted for in the crack growth rate, time to failure would likely be predicted more accurately.

The EPZ method produced interesting results. While this method did not outperform others in a significant way, it is able to capture coalescence of cracks outside of the dominant failure pathway. It was also able to capture reasonable crack curvature and potentially also crack branching behavior. These behaviors are likely a result of the elliptical PZs that exist at each crack tip and evolve in shape and orientation over time and with the location of the crack tip. This provides a large amount of variability in the direction in which cracks can grow, which is something that the other methods do not allow for. However in the EPZ method, it seems that much of the crack propagation is overpredicted in comparison to the HOSS results. In the EPZ images shown in Fig. 6, many of the cracks that are predicted to propagate also are shown to propagate in the HOSS results, however in the EPZ model they seem to propagate too quickly resulting in incorrect coalescence and sometimes incorrect failure path predictions. This perhaps reflects a dependence on one or more material factors within the posited crack growth factor (Eq. (2)) that is too strong. In addition, accounting for stress state information at the crack tips could also impact the crack growth rate improving results.

The NFPZ approach is somewhat a hybrid of the OP and EPZ models, and consequently it performs better than the EPZ method and not as well as the OP method for failure path prediction. This model does assume, based on Mode I failure arguments, that horizontal cracks will grow the fastest similar to the OP method. This idea is used to determine a zone where failure is likely. This method also assumes PZs present at crack tips, similar to the EPZ approach. The PZs were used to determine which cracks are likely to coalesce, and this also informs the likely location of failure. Unlike the EPZ method, the PZs are not recalculated with time and all crack tips have the same PZ size (which is true as long as $a_1 = a_2$). Adding some variability to the PZ calculation (e.g., dependence on orientation, stress state, etc.) could improve results. In addition, the likely region of failure maybe too narrowly defined. Other methods (specifically the OP and McPIC methods) that did not reduce the system to a local failure zone produce better results.

Comparison between these methods would be incomplete if it did not provide us with insights into the relative importance of using a combination of physics intuition and training data. Fig. 8 shows where the different methods fall on a conceptual spectrum of explicitly embedding physics into the model versus implicitly learning physics from the data. The HOSS model sits at the lower right portion of the plot, indicating that it has a great deal of physics incorporated into it, but is unable to learn from data. Of course, it is appropriate for a high-fidelity model like HOSS not to learn from data since it is not an ML model. The OP approach uses ML regression techniques and has a limited amount of physics embedded in it (i.e., the concept that Mode I failure will be dominant). However, it has somewhat limited trainability because of the simple feature set used in the ML method (i.e., only crack length is used to predict). The NFPZ and EPZ algorithms do not use traditional ML algorithms such as neural networks, but instead takes an approach where an ML model is a computer program with tunable parameters [54]. This approach enabled the models to incorporate more physical insights explicitly, but limited the potential to learn more physics from the data. The McPIC algorithm used standard ML models that are highly trainable and incorporated physics via feature engineering (i.e., things such as the stress intensity factor were added to the vector that can be used to make predictions) and emphasizing the importance of pairwise crack interactions.

This research exercise further confirms our notion that there exists a

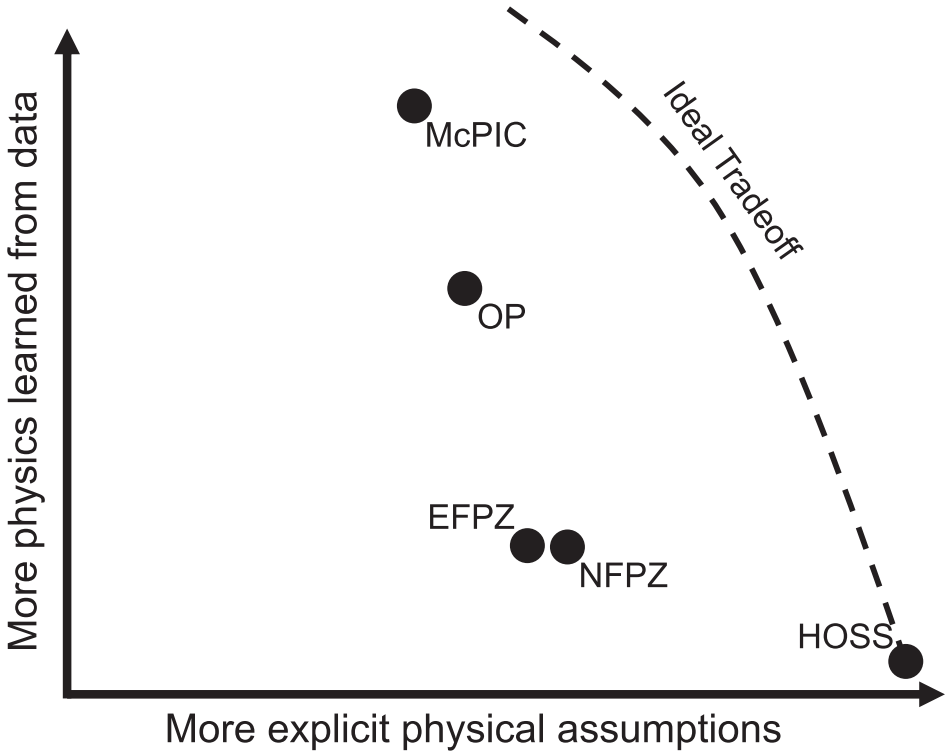

Fig. 8. The four models presented in this study as well as HOSS represented on a conceptual plot that visualizes the extent to which the models have built in more physics and how trainable the model is. The dashed line conceptualizes the ideal trade-off between embedding physics explicitly and learning physics implicitly from data. Note that in our conceptualization there is room for further movement toward the dashed line in all cases.

"sweet spot" or region on this conceptual plot, where the judicious use of physics-informed features in combination with training data is more amenable to learning the underlying complex physics of the crack propagation problem. This combination used by McPIC resulted in the optimal results seen here and provides a path forward for future modeling of this sort. As has been noted in a broader context, "feature engineering is the key" [55] to success with ML models. The results obtained here are not an exception to this rule. Going forward, smart feature engineering combined with flexible ML models is needed to further improve ML models that aim to predict the behavior of materials undergoing brittle crack. This approach was best exemplified by the McPIC model here which had the minimal physical assumptions baked in and used neural networks to learn from the data.

6. Summary

In summary, we have presented four different approaches (OP, McPIC, NFPZ, and EPZ models) that could be used to develop reduced-order models for informing larger length-scale simulations. The problem of interest in this work is crack network evolution in brittle geomaterials. In particular, we narrow the problem to address Mode I failure in a 2D concrete sample under pure tensile loading conditions. HOSS, the high-fidelity model chosen for this work, produced a training dataset of 150 simulations where the orientations and locations of the 20 pre-existing cracks in the sample are randomly chosen for each simulation. The OP, McPIC, NFPZ, and EPZ methods all rely on ML algorithms to help approximate the system by attempting to learn the dominant trends and effects that can determine an overall material response. All of these methods used the same 150 simulation training set to inform the ML algorithms. Two metrics were chosen for comparison: the failure pathway and the failure time. An additional 35 HOSS simulations were used for validation of the methods. Of the methods, the McPIC approach produced the most promising results for both metrics used in the comparison. This method is specifically focused on accounting for pairwise crack interactions. Future work includes incorporating higher order crack interactions into this approach and finding additional physics-based features that could help the model learn more from the data. The method could be used to produce probability density functions of how the overall damage in the cell is changing with time. Such information can be used to directly inform macro-scale continuum constitutive models through degrading elastic moduli, producing an overall material response that includes crack interaction effects in a model that cannot explicitly evolve discrete cracks. The key finding in this research study is that a combination of physics-informed and learned features are necessary for emulating the physics of crack propagation, interaction and coalescence.

Data availability

Data from Figs. 3 and 7 is in the supplementary information. The data is available from the authors upon request.

CRediT authorship contribution statement

Abigail Hunter: Conceptualization, Writing - original draft, Writing - review & editing, Visualization, Supervision, Funding acquisition. **Bryan A. Moore:** Methodology, Software, Investigation, Data curation, Writing - original draft, Visualization. **Maruti Mudunuru:** Methodology, Software, Investigation, Data curation, Writing - original draft. **Viet Chau:** Methodology, Investigation, Data curation. **Roselyne Tchoua:** Methodology, Software, Investigation, Data curation, Writing - original draft. **Chandramouli Nyshadham:** Methodology, Software, Investigation, Data curation, Writing - original draft, Visualization. **Satish Karra:** Conceptualization, Writing - review & editing, Supervision. **Daniel O'Malley:** Conceptualization, Writing - review & editing, Visualization, Supervision. **Esteban Rougier:** Conceptualization, Writing - original draft, Supervision. **Hari Viswanathan:** Conceptualization, Supervision. **Gowri Srinivasan:** Conceptualization, Writing - review & editing, Supervision, Project administration, Funding acquisition.

Acknowledgements

The authors would like to acknowledge support from the Los Alamos National Laboratory Directed Research and Development

(LDRD) Program through the Directed Research Project # 20170103DR. MKM and SK authors also thank the support of the LANL LDRD Early Career Award 20150693ECR. MKM gratefully acknowledges the support of LANL Chick-Keller Postdoctoral Fellowship through Center for Space and Earth Sciences (CSES). The authors thank the LANL Institutional Computing program for their support in generating data used in this work. The authors thank an anonymous reviewer whose comments substantially improve the manuscript.

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.commatsci.2018.10.036.

References

[1] S. Meyer, E. Diegele, A. Brückner-Foit, A. Möslang, Crack interaction modelling, Fatigue Fract. Eng. Mater. Struct. 23 (2000) 315–323.
[2] J. Escobedo, C. Trujillo, E. Cerreta, G. Gray III, E. Brown, Effect of shock wave duration on dynamic failure of tungsten heavy alloy, Journal of Physics: Conference Series, vol. 500, IOP Publishing, 2014, p. 112012.
[3] D.J. Lucia, P.S. Beran, W.A. Silva, Reduced-order modeling: new approaches for computational physics, Prog. Aerosp. Sci. 40 (1) (2004) 51–117.
[4] I.H. Witten, E. Frank, M.A. Hall, C.J. Pal, Data Mining: Practical Machine Learning Tools and Techniques, Morgan Kaufman, 2016.
[5] E. Alpaydin, Introduction to Machine Learning, MIT press, 2014.
[6] G. Weikum, Foundations of statistical natural language processing, ACM SIGMOD Rec. 31 (3) (2002) 37, https://doi.org/10.1145/601858.601867 arXiv:arXiv:1011.1669v3 http://portal.acm.org/citation.cfm?doid=601858.601867.
[7] P. Larrañaga, B. Calvo, R. Santana, C. Bielza, J. Galdiano, I. Inza, J.A. Lozano, R. Armañanzas, G. Santafé, A. Pérez, V. Robles, Machine learning in bioinformatics, Briefings Bioinform. 7 (1) (2006) 86–112, https://doi.org/10.1093/bib/bbk007 arXiv:arXiv:1412.3919v1.
[8] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, Adv. Neural Inform. Process. Syst. (2012) 1, https://doi.org/10.1016/j.protcy.2014.09.007 9arXiv:1102.0183.
[9] T.K. Ho, Random decision forests, Proceedings of the Third International Conference on Document Analysis and Recognition, 1995. Vol. 1 IEEE, 1995, pp. 278–282.
[10] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. 20 (3) (1995) 273–297.
[11] W.S. McCulloch, W. Pitts, A logical calculus of the ideas immanent in nervous activity, Bull. Math. Biophys. 5 (4) (1943) 115–133.
[12] K. Rajan, Materials informatics, Mater. Today 8 (10) (2005) 38–45, https://doi.org/10.1016/S1369-7021(05)71123-8.
[13] K. Takahashi, Y. Tanaka, Materials informatics: a journey towards material design and synthesis, Dalton Trans. 45 (26) (2016) 10497–10499, https://doi.org/10.1039/C6DT01501H.
[14] A. Jain, S.P. Ong, G. Hautier, W. Chen, W.D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder, et al., Commentary: the materials project: a materials genome approach to accelerating materials innovation, Apl Mater. 1 (1) (2013) 011002, https://doi.org/10.1063/1.4812323.
[15] T.T. Tran, M. Obrovac, Alloy negative electrodes for high energy density metal-ion cells, J. Electrochem. Soc. 158 (12) (2011) A1411–A1416, https://doi.org/10.1149/2.083112jes.
[16] I.E. Castelli, T. Olsen, S. Datta, D.D. Landis, S. Dahl, K.S. Thygesen, K.W. Jacobsen, Computational screening of perovskite metal oxides for optimal solar light capture, Energy Environ. Sci. 5 (2) (2012) 5814–5819, https://doi.org/10.1039/C1EE02717D.
[17] M. Meinert, M.P. Geisler, Phase stability of chromium based compensated ferrimagnets with inverse Heusler structure, J. Magn. Magn. Mater. 341 (2013) 72–74, https://doi.org/10.1016/j.jmmm.2013.04.025.
[18] L.J. Nelson, G.L. Hart, F. Zhou, V. Ozoliņš, et al., Compressive sensing as a paradigm for building physics models, Phys. Rev. B 87 (3) (2013) 035125, https://doi.org/10.1103/PhysRevB.87.035125.
[19] K. Takahashi, Y. Tanaka, Role of descriptors in predicting the dissolution energy of embedded oxides and the bulk modulus of oxide-embedded iron, Phys. Rev. B 95 (1) (2017) 014101, https://doi.org/10.1103/PhysRevB.95.014101.
[20] E.E. Knight, E. Rougier, Z. Lei, A. Munjiza, User's manual for Los Alamos National Laboratory hybrid optimization software suite (HOSS) - educational version, Tech. Rep. LA-UR-16-23118, Los Alamos National Laboratory, 2016.
[21] G. Srinivasan, J.D. Hyman, D.A. Osthus, B.A. Moore, D. O'Malley, S. Karra, E. Rougier, A.A. Hagberg, A. Hunter, H.S. Viswanathan, Quantifying topological uncertainty in fractured systems using graph theory and machine learning, Scient. Rep. 8 (1) (2018) 11665, https://doi.org/10.1038/s41598-018-30117-1.
[22] C.T. Sun, Z.H. Jin, Fracture Mechanics, Elsevier Inc., Oxford, UK, 2012.
[23] G. Irwing, Analysis of stresses and strains near the end of a crack traversing a plate, J. Appl. Mech. 24 (1957) 361–364.
[24] E.E. Knight, E. Rougier, Z. Lei, Hybrid optimization software suite (HOSS)-educational version, Tech. Rep. LA-UR-15-27013, Los Alamos National Laboratory, 2015.
[25] E. Rougier, E.E. Knight, A. Munjiza, LANL-CSM: HOSS - MUNROU technology overview, Tech. Rep. LA-UR-13-23422, Los Alamos National Laboratory, 2013.
[26] E.E. Knight, E. Rougier, A. Munjiza, LANL-CSM: Consortium proposal for the advancement of HOSS, Tech. Rep. LA-UR-13-23409, Los Alamos National Laboratory, 2013.
[27] A. Munjiza, Discrete elements in transient dynamics of fractured media, Ph.D. thesis, Swansea University, Swansea, Wales, 1992.
[28] A. Munjiza, The Combined Finite-Discrete Element Method, Wiley, New York, 2004.
[29] A. Munjiza, E.E. Knight, E. Rougier, Computational Mechanics of Discontinua, Wiley, New York, 2012.
[30] A. Munjiza, E. Rougier, E.E. Knight, Large Strain Finite Element Method: A Practical Course, first ed., John Wiley and Sons, London, 2015.
[31] E. Rougier, E.E. Knight, S.T. Broome, A. Sussman, A. Munjiza, Validation of a three-dimensional finite-discrete element method using experimental results of the split Hopkinson pressure bar test, Int. J. Rock Mech. Min. Sci. 70 (2014) 101–108.
[32] A. Munjiza, D.R.J. Owen, N. Bicanic, A combined finite-discrete element method in transient dynamics of fracturing solids, Eng. Comput. 12 (2) (1995) 145–174.
[33] E. Rougier, A. Munjiza, N.W.M. John, Numerical comparison of some explicit time integration schemes used in DEM, FEM/DEM and molecular dynamics, Int. J. Numer. Meth. Eng. 61 (2004) 856–879.
[34] A. Munjiza, K.R.F. Andrews, J.K. White, Combined single and smeared crack model in combined finite-discrete element analysis, Int. J. Numer. Meth. Eng. 44 (41-57) (1999) 791–797.
[35] T. Nicholas, Tensile testing of materials at high rates of strain, Exp. Mech. 21 (5) (1981) 177–185.
[36] B. Ycart, A case of mathematical eponymy: the vandermonde determinant (2012), https://arxiv.org/pdf/1204.4716.pdf.
[37] 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, E. Duchesnay, Scikit-learn: machine learning in python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[38] A.A. Griffith, The phenomena of rupture and flow in solids, Philos. Trans. R. Soc. Lond. 221 (1921) 163–198.
[39] P. Paris, F. Erdogan, A critical analysis of crack propagation laws, J Basic Eng 84 (4) (1963).
[40] B.A. Moore, E. Rougier, D. O'Malley, G. Srinivasan, A. Hunter, H. Viswanathan, Predictive modeling of dynamic fracture growth in brittle materials with machine learning, Comput. Mater. Sci. 148 (2018) 46–53.
[41] V. Veselý, L. Ro˘util, Z. Keršner, Structural geometry, fracture process zone and fracture energy, in: Proceedings of Fracture Mechanics of Concrete and Concrete Structures, Catania, Italy, 2007.
[42] X.Z. Hu, F.H. Wittmann, Fracture energy and fracture process zone, Mater. Struct. 25 (1992) 319–326.
[43] Z.P. Bažant, P.A. Pfeiffer, Determination of fracture energy from size effect and brittleness number, ACI Mater. J. 84 (1987) 463–480.
[44] Z.P. Bažant, M.T. Kazemi, Determination of fracture energy, process zone length and brittleness number from size effect, with application to rock and concrete, Int. J. Fract. 44 (1990) 111–131.
[45] P.-E. Petersson, Crack growth and development of fracture zones in plain-concrete and similar materials, Ph.D. thesis, Lund Institute of Technology, Lund, Sweden, 1981.
[46] Z. Brooks, Fracture process zone: Microstructure and nanomechanics in quasi-brittle materials, Ph.D. thesis, Massachusetts Institute of Technology, Massachusetts, USA, 2013.
[47] Y.-Z. Wang, J.D. Atkinson, R. Akid, R.N. Parkins, Crack interaction, coalescence, and mixed mode fracture mechanics, Fatigue Fract. Eng. Mater. Struct. 19 (1995) 427–439.
[48] L.B. Freund, Dynamic Fracture Mechanics, Cambridge Monographs on Mechanics and Applied Mathematics, Cambridge University Press, Cambridge, UK, 2012.
[49] S. Skiena, Dijkstra's algorithm, Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica, Reading, MA: Addison-Wesley, 1990, pp. 225–227.
[50] M. Mudunuru, N. Panda, S. Karra, G. Srinivasan, V. Chau, E. Rougier, A. Hunter, H. Viswanathan, Estimating failure in brittle materials using graph theory, arXiv preprint arXiv:1807.11537.
[51] T. Anderson, Fracture Mechanics: Fundamentals and Applications, CRC press, 2005.
[52] A. Hagberg, P. Swart, D. Chult, Exploring network structure, dynamics, and function using NetworkX, Tech. Rep. LA-UR-08-05495, Los Alamos National Laboratory, 2008.
[53] E. Jones, T. Oliphant, P. Peterson, SciPy: open source scientific tools for Python, [Online, accessed 2017-06-11] (2001–). <http://www.scipy.org/>.
[54] M. Innes, D. Barber, T. Besard, J. Bradbury, V. Churavy, S. Danisch, A. Edelman, S. Karpinski, J. Malmaud, J. Revels, V. Shah, P. Stenetorp, D. Yuret, On machine learning and programming languages (2017), https://julialang.org/blog/2017/12/ml&p.
[55] P. Domingos, A few useful things to know about machine learning, Commun. ACM 55 (10) (2012) 78–87.