Machine Learning for Defect Identification in Crystalline Materials

Michael D Skarlinski and David J Quesnel

University of Rochester, Hopeman Engineering Building, Rochester NY, 14627

E-mail: michael.skarlinski@rochester.edu

Abstract. Machine learning is a versatile tool for finding complex trends in large datasets. Recent developments in atomistic representations have resulted in rotationally, translationally, and permutationally invarient expressions for local atomic environments. Here we use these unique representations to collect a small database of standard crystalline defects, such as partial and full dislocations, surfaces vacancies, interstitials, and grain boundaries for both FCC and BCC materials. Both thermal perturbations and strains are included in the training data to make the final classification algorithm robust against erors in both. We develop a machine learning approach, called BiDef, to identify similar defects which were not present in the training data, which remains quite flexible for adding new defects or atomic structures. Error analysis indicates that the BiDef algorithm is more accurate than CSP and CNA methods in idenfying defects during tensile testing of nanowire samples, as well as vacancy diffusion experiments. However, the power and flexibility of the method comes at a relatively large comutational cost and the requirement of a robust training library.

1. Introduction

1.1. Defect Identification in Crystalline Simulations

The fact that mechanical, electrical and optical properties of materials are largely determined by crystalline defects is a fundamental tenet of materials science. For example, grain boundaries can act as dislocation sources or sinks, or serve to hinder dislocation motion through a crystal, causing Hall-Petch hardening. Vacancy clusters can collapse to cause stacking faults separated by partial dislocations, and interstitial atom strain fields can inhibit dislocation motion. Energy differences among atomic planes can facilitate or retard crack formation along those particular atomistic directions.

Molecular dynamics gives scientists an excellent tool to investigate these defect structures such as surfaces, grain boundaries, dislocations, interstitial atoms or vacancies. Since these simulations only track the trajectories of atoms, it is necessary to develop classification schemes which allow for the particular atoms contained within these defects to be identified. As simulations evolve into larger scales or complexities,

the identification of defects within crystals becomes a labor intensive and sometimes imprecise task. Currently, many defect identification methods exist with a variety of applicabilities for different scenarios. Comparisons between the most popular methods are given by Stukowski[21].

As noted his work, geometric indicators are often used such as inter-atomic distances, bond angles or bond network topology. Popular descriptor methods include Common neighbor analysis (CNA)[8], Centrosymmetry parameterization[10], and bond-order analysis[19]. We use these techniques (particularly CNA and CSP) as benchmarks for defect identification performance in this work. CNA and adaptive-common neighbor analysis [21] are defect identification methods based on variations in the topology of bonds in the neighborhood of a given atom. The method works by taking N nearest neighbors, where N is 12 for FCC and HCP, 14 for BCC, and 16 for cubic diamond, and comparing three different bond properties between each of them. The three properties are, n_{cn}, the number of neighbor atoms shared between the central atom and each atom its bonded to, n_b, the number of bonds between each of these common neighbors, and n_{lcb}, the lowest number of bonds possible in a chain which connects the common neighbors. This set of three descriptors is used for each of the N neighbors, and allows for unique identification of the local structure surrounding a particular atom. CSP[10] analysis provides another tool for defect identification, where local bond symmetry is used as a descriptor. The centrosymmetry parameter is defined as,

$$CSP = \sum_{i=1}^{N/2} |\mathbf{r_i} + \mathbf{r_{i+N/2}}|^2. \tag{1}$$

Where $\mathbf{r_i}$ and $\mathbf{r_{i+N/2}}$ are vectors between the atom of interest and a pair of opposite neighbors. The pairs are identified by summing the bond vectors for each of the $N(N-1)/2$ possible atom pairs, and finding the $N/2$ pairs with the smallest bond magnitude, $|\mathbf{r_i}+\mathbf{r_j}|^2$. When the atom of interest is in a perfect lattice site, only symmetric pairs of opposite atoms will be chosen, giving a CSP value near zero. The less symmetric the pairs of atoms are, the higher the CSP value. Bond-order analysis [19] identifies defects by projecting the vectors between neighbors onto a unit sphere, then defines a set of parameters based on rotationally invariant combinations of spherical harmonics, which are characteristic to each crystal structure. The technique was developed for identifying glass transitions in metallic glass, and it allows for characterization between amorphous and crystalline states.

While the primary advantage of the above methods is in their speed and simplicity, they suffer from being relatively inaccurate at high temperatures or crystalline strains. Furthermore, they have little ability to uniquely identify defective structures: CNA can only determine the presence of crystal structures and CSP analysis will give a scalar value which is difficult to correlate to a particular defect. Some more recent methods have made strides toward unique defect identification, including the dislocation extraction algorithm (DXA)[22] and Topological fingerprint based analysis[17]. DXA

allows dislocations to be completely identified within a crystalline structure. The algorithm is essentially split into three steps, firstly, ordered atoms are identified by CNA, then the defect atoms are taken and transformed into an *interface mesh*, where the perfect crystal is separated from the disordered portions. Finally, complete Burgers circuits are generated on the 2-d manifold where dislocations can then be transformed into one dimensional lines. The method is quite robust at identifying dislocations, and even pointing out areas where non-dislocation or unidentified defects are located in the crystal. However, the method lacks the ability to classify any type of defect, and will suffer performance degradation if the CNA analysis performed is flawed at high strain values.

While the previously mentioned techniques have been empirically successful at identifying structural defects, we find that improvements can be made in the unique identification of defect structures. A method capable of identifying individual defects would bypass costly manual investigations into the atomic structure using visualization or post-processing techniques. For particularly large structures of 10's of millions of atoms or more, it may not even be possible for a scientist to verify defect orientations within the crystal. The frequent use of post-processing techniques (such as slip-vector analysis [27]) is a time consuming process for complex configurations and structures. The ability to uniquely identify different atomic configurations would allow for the dynamic assignment of regions in which the chosen inter-atomic potentials are non-transferable. Potentials which were parametrized for a particular defect could be focused on atoms within that defect region. This concept could be expanded by the introduction of ab-initio methods for atoms in 'un-fitted' configurations. Here we present a technique based on the use of machine learning for the specific identification of crystal defects. We demonstrate low temperature and strain sensitivity in the atomistic identification, even when simulating non-trained data.

A similar technique has been previously explored in the literature, known as topological fingerprint analysis. [17] The method works by first identifying all atoms occupying the verticies of the coordination polyhedra that are formed around each atom of interest. Crystal structures will often form several characteristic coordination polyhedron structures, with different nearest neighbor counts. Coordination polyhedra for a given atom are identified by a best-fit type algorithm to known structures in a database. By examining relationships in viable pathways to travel from the initial atom of interest to atoms recursively located in the coordination polyhedra of neighbors, then back to the initial atom, many atomic structures can be identified. The number of 'hops' needed to travel from a subject atom to atoms within the coordination polyhedron and back define the topological order. The topological fingerprints are defined as a function of the topological order, where the fingerprint is a listing of the proportion of viable pathways which travel through each available combination of coordination polyhedron types in the crystal of interest. This fingerprint was shown to provide unique values for many defects such as FCC-BCC interfaces, vacancies and interstitials. The machine learning technique presented in this thesis differs in implementation and sensitivity as

detailed in the next section.

1.2. Machine Learning

Machine learning can be broadly defined as a set of algorithms that learns and make predictions based on a set of data. The algorithms build models using input data, and make predictions based on these models as opposed to following a set of pre-made instructions.[13] Supervised learning algorithms work with a vectorized set of input variables, known as *features*, that have been pre-matched to particular values or *classes*. A model is then created which classifies previously unseen features.

The features which represent atomic environments are critical for this method, and each of the previously presented defect identification methods essentially creates its own feature space for defective atoms. We were seeking a complete representation of an atom's local environment such that each defect would be able to be uniquely identified. The bond-order spectrum is a nearly complete representation of an atoms local environment, however, these components only provide the spherical symmetry around each atom, the radial information is not included. There exists a more thorough approach, developed by Albert Bartk-Prtay [2], where a rotationally, translationally and permutationally invariant representation is used, which utilizes an expansion in bi-spectrum coefficients to represent atomic environments. These coefficients were previously used to make a dynamically quantum-accurate classical model, known as Gaussian Applied Potentials (GAP) [1]. However, we use these same coefficients as features in a defect identification learning algorithm, rather than a force and energy determination algorithm as in GAP. The previously explained topological fingerprint method also provides a viable featurespace for machine learning of defect structures, even though many structures require higher order fingerprints which increase the cost. However, we have chosen to use bi-spectrum coefficients as features since they are highly sensitive to small changes in local atomic positions. These changes would be missed by the topological fingerprints if nearest neighbor ordering remained unchanged. Furthermore, we don't need to generate the coordination structures at the start of each analysis, as required in the topological fingerprint method. The features for this space are simply the list of bi-spectrum components for each atom in the system, and our classes are descriptions of the local atomic site. The classes may be defect-free sites, such as bulk FCC or bulk BCC, as well as particular defects, including grain boundaries, partial dislocations, interstitial sites, or vacancies. Using this classification scheme, the number of features varies from 8 to 55 depending on how many bi-spectrum coefficients are considered, and the number of samples in the training set is well above a million.

A classification algorithm must be then chosen after constructing a database of features and classes. Many different supervised classification algorithms exist such as support vector classification [18], nearest-neighbor approaches [3, 15], naive-bayes [25], and ensemble based methods such as random forests [4] or Ada-boost[26]. Each method differs in efficiency and applicability for a particular set of data, where consideration of

data-set size, variability, and number of classifications needs to be taken into account. in considering the large size of our training set, we chose to use a nearest-neighbor KD-Tree approach [3], as it scales well to many features. Random forest ensemble methods[4] are also evaluated in this work, despite the slow run-time these algorithms train quickly, and are able to dynamically differentiate the importance of each featured variable. Furthermore, random forest algorithms are capable of more accurate classifications than nearest-neighbor approaches, given that enough decision trees are used. Support vector machines were not investigated due to the increased training time required for such large data sets. A succinct summary of these classification algorithms are included in the Methods section.

An interesting advantage of this learning method is that it is fundamentally capable of identifying any reproducible atomic defect–no matter how obscure. It is capable of identifying any class for which it has trained with similar features. This means that including the effects of thermal perturbations or strains is as simple as generating the strained or thermalized structures themselves, and then adding them to the current training set. The incorporation of a new defect into the present model will only make the identification algorithm more robust and thorough, giving it the ability to find the defect in any new set of atomic configurations presented to it. In the prototype version of the method, it is capable of handling thermal and strain-based fluctuations, as well as the identification of a few choice defects in FCC and BCC crystals. The method's performance and variability was evaluated in identifying a list of common defects in FCC and BCC systems, when tested in applications outside the training set.

2. Methods

2.1. Unique atomic representations: Bispectrum Coefficients

In this section, some background is given on the rotationally, translationally, and permutationally invariant bi-spectrum components is described as developed by Albert Bartk-Prtay [2] They have since been adapted for use in another automated interatomic potential algorithm, Spectral Analysis Method for Automated Generation of Quantum-Accurate Interatomic Potentials or SNAP. [23] The bi-spectrum coefficients are later used to train a machine learning dataset based on structural defects.

Firstly an atomic density function, $\rho_i(\mathbf{r})$ is established as a sum of δ functions in space:

$$\rho_i(\mathbf{r}) = \delta(\mathbf{r}) + \sum_{r_{ii'}<R_{cut}} f_c(r_{ii'}) w_{i'} \delta(\mathbf{r} - \mathbf{r_{ii'}}), \tag{2}$$

here $\mathbf{r_{ii'}}$ is the vector between the atom of interest, i to a neighboring atom, i'. The coefficients w_i are weighting factors to differentiate between atom types in multi-type systems. The sum occurs for all atoms within a cutoff R_{cut}, and $f_c(r_{ii'})$ is a smoothing function to avoid a discontinuity at R_{cut}. In a similar fashion to the bond-order analysis [19], the angular portion of the density function can be expanded in a basis of spherical

harmonic functions, $Y_m^l(\theta,\phi)$, for $l = 0, 1, 2, ...$ and $m = -l, -l+1, ..., l-1, l$. [24] Rather than expanding the radial component in a separate set of basis functions, Bartok et al. [1] had the insight to map the radial distance, r to a polar angle θ_0 defined as follows,

$$\theta_0 = \theta_0^{max}\frac{r}{R_{cut}} \tag{3}$$

θ_0 maps the points, (θ, ϕ, r) to a a set of points on a 3-sphere (sphere analog in 4-D), (θ, ϕ, θ_0). Some θ_0^{max} value is defined to limit points to be north of this latitude. The advantage of this trick is that 4D spherical harmonics functions, $U^j_{m,m'}(\theta_0, \theta, \phi)$, can be used as a basis set, where $j = 0, \frac{1}{2}, 1, ...$ and $m, m' = -j, -j+1, ..., j-1, j$. If the 4D harmonics functions are parametrized with Euler angles, then they become identical to Wigner D-functions from quantum angular momentum. The atomic-neighbor density function can now be represented as follows,

$$\rho(\mathbf{r}) = \sum_{j=0,\frac{1}{2},...}^{\infty}\sum_{m=-j}^{j}\sum_{m'=-j}^{j} u^j_{m,m'}U^j_{m,m'}(\theta_0,\theta,\phi) \tag{4}$$

The expansion coefficients are found by using the inner product of the local atomic density with the basis functions. Since the density is comprised of delta function sums, the coefficients can be written as a sum over local atomic locations within a cutoff radius,

$$u^j_{m,m'} = U^j_{m,m'}(0,0,0) + \sum_{r_{i,i'}<R_{cut}} f_c(r_{i,i'})w_{i'}U^j_{m,m'}(\theta_0,\theta,\phi) \tag{5}$$

The expansion coefficients are not invariant under rotation, so a scalar triple product is used to enforce rotational invariance. [2] Coupling coefficients, also known as the Clebsch-Gordan coefficients, can then be used to construct the bi-spectrum coefficients,

$$B_{j_1,j_2,j} = \sum_{\substack{m_1=-j_1,\\ m_1'=-j_1}}^{j_1}\sum_{\substack{m_2=-j_2,\\ m_2'=-j_2}}^{j_2}\sum_{\substack{m=-j,\\ m'=-j}}^{j}(u^j_{m,m'})H^{\,jmm'}_{\substack{j_1m_1m_1',\\ j_2m_2m_2'}}u^{j_1}_{m_1,m_1'}u^{j_2}_{m_2,m_2'} \tag{6}$$

These coefficients characterize the density correlation for points on the 3-sphere. The order of the expansion will determine the coarseness of the detail in the atomic description. In practice, the value of j that is is chosen for truncation will balance the detail and computational load necessary to compute the bi-spectrum. The number of bi-spectrum coefficients will also determine the size of the feature vectors which are used for training our classification algorithm, strongly affecting the classification's performance. Since j is represented as 1/2 integers, we will use $2j$ values as the descriptor for comparing performance when generating the bi-spectrum coefficients. The number of bi-spectrum coefficients generated for a given j value is a non-linear relation. By taking advantage of the symmetries in the bi-spectrum description, it has been shown[23] that there are $(j+1)(j+2)(j+\frac{3}{2})/3$ unique coefficients as a function of j.

The coefficients which are generated are a function of a nearest-neighbor radial cutoff parameter. The cutoff value must be large enough to capture the scale of the defects of interest while balancing the performance needed in the nearest neighbor based summed expressions. Here we used 2 times the equilibrium lattice parameter for each

of the crystalline systems of interest. We found that the total time to generate bi-spectrum coefficients scaled linearly with nearest neighbor size, as expected from Eqn.5. For defects with larger structures than two equilibrium lattice structures, large cutoffs will be necessary for unique identification, slowing down performance.

2.2. Nearest Neighbor Classification Algorithms

Our intention was to cluster and identify pre-labeled defect structures using bi-spectrum coefficients as features. Since the training set was large (about 5 million atoms), a nearest neighbor approach[5, 3, 15], which is one of the simplest classification methods, was evaluated for use in our learning algorithm. The conceptual basis behind nearest neighbor classification algorithms is quite simple, first a distance metric is defined in feature space (typically a euclidean distance). When new points are introduced to be classified, the algorithm calculates the distance metric between trained points in feature space, and choses the classification for which the majority of training points have the minimum distance. If there is a tie in the classifications of nearest-neighbors, then it is randomly chosen. An advantage of these methods is that they are purely memory based, so they do not require that any model be fit to the data.

The trade-off between variance and bias in the nearest-neighbor algorithm is dictated by the choice of the number, k of nearest neighbors. Having few neighbors (as few as 1, for instance) produces a model with extremely high variance, and low bias. It does not generalized well to unseen data. By increasing the number of nearest neighbors the variance decreases, as the bias increases.

The process of choosing nearest neighbors can present a computational burden, where a brute force approach would result in calculating the distance between all N training points to find the k closest neighbors. This results in a total search process which operates at $O(ND)$, where D is the number of features in the training set. There are several algorithms for reducing the computational load such as the KDTree [3] and BallTree [15], which are $O(D * log(N))$ algorithms under the correct circumstances. These algorithms work by separating the feature space into discrete regions such that data occupying different regions will need not be considered as a nearest neighbors. The algorithm used here was the BallTree [3] for $D > 20$ and the KDTree for smaller dimension ($2j < 4$ values for our bi-spectrum limits).

2.3. Random Forests Classification Ensembles

Random Forests [4] are an ensemble method, which implies that the classification decision is made on the basis of an ensemble of slightly-varying simpler methods. For the aptly-named random forest, the classification is based on an ensemble of decision tree classifiers. [5] A decision tree is a structure which divides up a feature space into rectangular regions on the basis of minimizing the node-impurity of the tree. The tree is constructed by recursively splitting the data into two parts, where each portion minimizes some function in analogy to the loss function for regression problems. The

splits are made on one dimension in feature space. Ultimately the tree is left with nodes from each split decision, which lead down to terminal or leaf nodes, where the splitting stops. Choosing when to stop the tree growth can control the variance and bias of the model. Here, we stop growth when the splitting decision would produce a leaf node with fewer members than some minimum value. By adjusting this value, we can control for over-fitting and bias within our model.

The node impurity being minimized is known as a Gini index. Given some set of classifications with values $k = 1, 2, ..., K$, and a particular tree node m, the Gini index is defined as follows:

$$Gini = \sum_{k \neq k'} \hat{p}_{mk} \hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_m k)$$
$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k) \tag{7}$$

Where $\hat{p}_{mk}$ is the proportion of each class, k, that is present in node m. R_m are all samples in the data set of class m, and I is a function which is unity when a feature's class is equal to k. The classifier works by assigning the majority class, $k(m) = argmax_k(\hat{p}_{mk})$, to each member of node m. The Gini index is sensitive to non-majority class probabilities, such that node splits which minimize the probability of the non-majority nodes will best minimize the index. The algorithm iterates through node choices in feature space (across the range of points in each feature dimension), and choses a split which minimizes the Gini index. This continues until the stopping criterion is met.

The final created tree is useful because it naturally splits the data into probabilistic categories, and it inherently finds feature importances. It is able to find feature importances because features which produce a lot of variance, or correlate well, with the classification data, will be chosen as splitting nodes much more frequently. Since the trees will continue to find nodes until some stopping criterion is reached, they are quite prone to over-fitting. They are also prone to instability, as small variations in data structure can cause entirely different node choices and trees to form.

Random Forests are a system where the disadvantages of the decision trees can be minimized by combining a large number of 'de-correlated' trees, and averaging the results. The de-correlation is enforced by choosing a limited subset of features for the construction of each tree. We used a typical choice in the Random Forest, where $\sqrt{n}$ features were used for each tree, where n is the total number of features. Classifications are chosen by running the entire ensemble of trees (here we used 1000) and choosing the class which was chosen by the majority.

2.4. Training Set Construction

The learning algorithms required ample training data where crystalline defects were correctly identified such that they could be the proper features could be labeled. The

following crystal defects were created to train the algorithm: vacancies, interstitials, surfaces ((111), (112), (110), (100)), partial dislocations, full dislocations, and randomly oriented grain boundaries. Each of the defects were generated for both FCC and BCC systems. The effects of temperature were included by adding random displacements between 0 and 0.04 $\mathring{A}$ on each atom, and generating 20 copies of each defect with different random configurations. Normal strains were also included by straining the cell in all 1-d, 2-d, and 3-d combinations of $\hat{x}$, $\hat{y}$, and $\hat{z}$ directions at both 5% and 10% strains. Dislocations were created by applying burgers vectors within material slabs using the atomsk [7] code. The partial dislocations were of the $b = \frac{a_0}{6} < 112 >$ type for FCC. The full dislocations were of $b = \frac{a_0}{2} < 110 >$ and $b = \frac{a_0}{2} < 111 >$ for FCC and BCC cells, respectively. All bi-spectrum components were generated using atomic structures built using the equilibrium lattice parameter of copper ($3.597\mathring{A}$). The bi-spectrum nearest neighbor cutoff radius was two times this lattice parameter. Since the coefficients are sensitive to scaling, the BCC structures were also scaled to the same lattice parameter for training.

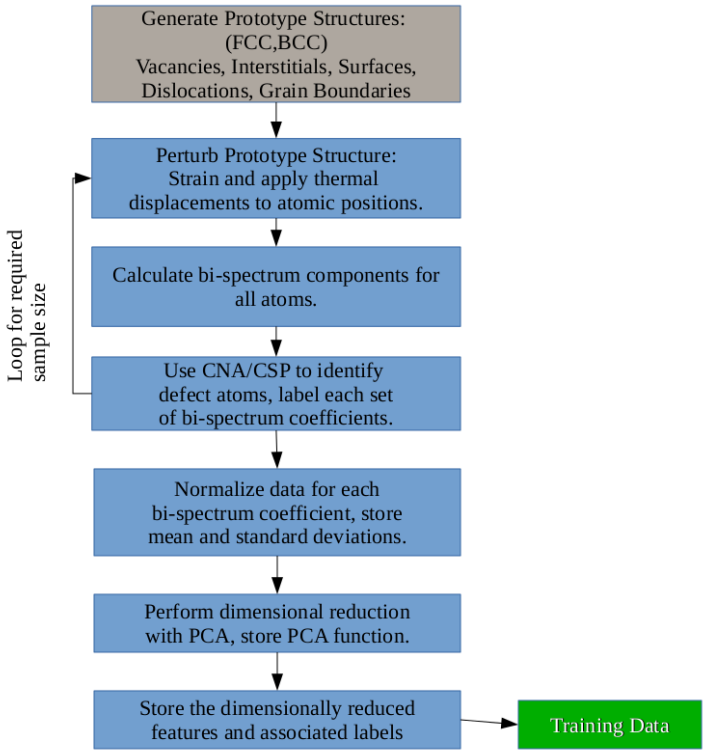


Figure 1: Algorithm for building the training data in defect identification.

The training process is featured schematically in Fig. 1. Each cell was generated as a prototype structure (before any strains or thermal perturbations), after which the defects were initially identified by use of either Common Neighbor Analysis (CNA)[21] or Centrosymmetry Paramter (CSP)[10]. The full description of each prototype training cell is included in the Appendix. CNA was used to capture the bulk regions of the

crystals, since an individual defect was present in each prototype structure, then the non-bulk atoms could be labeled with their proper class for each case. CSP was used in cases where only minor perturbations needed to be considered as part of the defect, such as for full-dislocation core atoms.

Each prototype was perturbed through strain and small random variations to recreate temperature, then the bi-spectrum components were computed for each atom using a dynamic library in the LAMMPS [16] code. The set of all bi-spectrum coefficients were then stored, along with each atom's associated zero-temperature and strain free defect labels. This procedure was repeated for each prototype until a sufficient spectrum of perturbations was acquired for each defect. The data were then transformed to have zero mean, and were scaled by the standard deviation of each feature (bi-spectrum component) such that the magnitude difference between bi-spectrum coefficients would not affect their significance in the machine learning algorithms. The complete training data set had approximately 5 million atoms with approximately 600,000 labeled defect atoms.

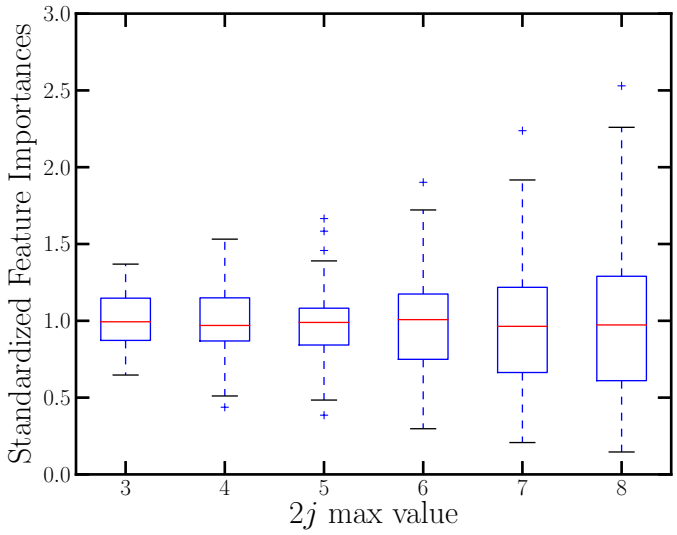


Figure 2: Distribution of feature importances calculated during random forest construction. The box represents the upper and lower quartile limits, with a line at the median. The whiskers show the total data range, with outliers represented by points.

Dimensionality reduction is difficult at this point, because while bi-spectrum components may provide minimal variance in the data for the currently chosen defects, these coefficients may be required to identify the specificities of new defects introduced into the data set. For this reason, we chose not to perform dimensionality reduction by means of eliminating the features which provided the least variance in classification outcome. The feature importances are implicitly calculated while building random

forests by determining the number of splits needed for each feature. In Fig. 2, the feature importances are plotted as a function of $2j$, the parameter in Eqn. 2.31 that determines the limits of the bi-spectrum expansion. The importances have been standardized such that they each have a mean of 1. The further the values are from one, the more or less important they are to identifying defects within the crystal. As expected, as the number of bi-spectrum components increases the relative importances begin to spread. At $2j = 8$, which results in a total of 55 bi-spectrum coefficients, even the least significant values are still about 10 % as important as the mean, leading us to believe that some particular defects may require higher order bi-spectrum expansions. It seems there are no negligible bi-spectrum contributors, even with 55 coefficients.

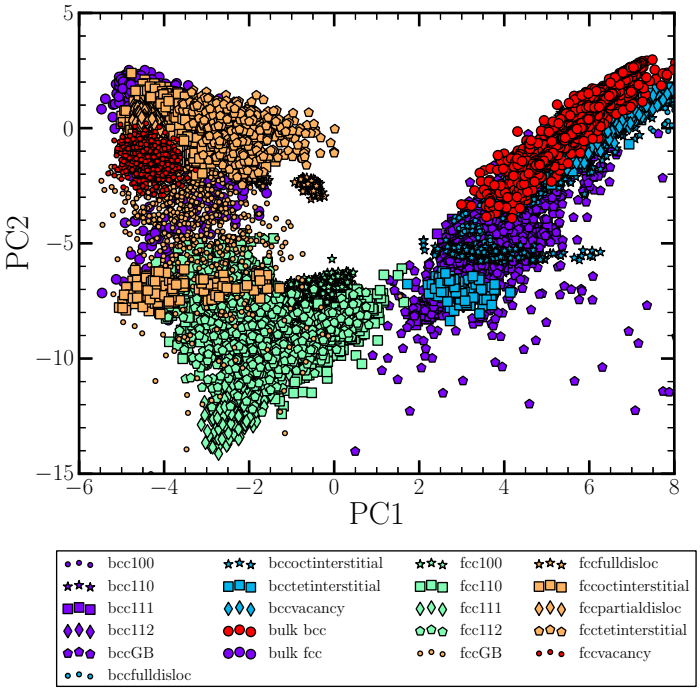


Figure 3: Two component PCA decpomposition of all defects considered in this study.

As an investigative tool into the efficacy of using the bi-spectrum coefficients for defect features, we performed a principal component analysis on the $2j = 8$ data sets, with 55 total features. The PCA was performed, and only the first two principal components were considered for the visualization shown in Fig. 3. It becomes immediately apparent that more than two components are required to uniquely identify the defects considered in this study. There is substantial overlap between defect regions which causes missed or erroneous classifications. The most relevant finding from the PCA in Figs. 3 and 4 is that small atomic position perturbations from strain or temperature will produce closely clustered (in terms of euclidean distance) points in bi-spectrum space. This means that using bi-spectrum coefficients as features will be a stable methodology for nearest neighbor or decision tree algorithms.

The spread in the PCA distribution for each defect results from the thermal and stress-based position fluctuations relative to each atom's nearest neighbors. It's interesting to note that a few defects have dense distributions relative to the bulk atom distributions, including the FCC vacancies, FCC full dislocations, and BCC tetrahedral interstitial sites. This implies that these defects have environments which are less susceptible to variation under thermal fluctuations when compared to bulk sites. A second plot of the defect PCA decomposition, with only bulk and surface defects is shown in Fig. 4. Even in this low dimensional representation, it can be seen that the different surface defects tend to separate in bi-spectrum coefficient space such that a dividing surface can be defined between bulk and defective regions. These encouraging visualizations lead us to push forward with the technique's implementation.

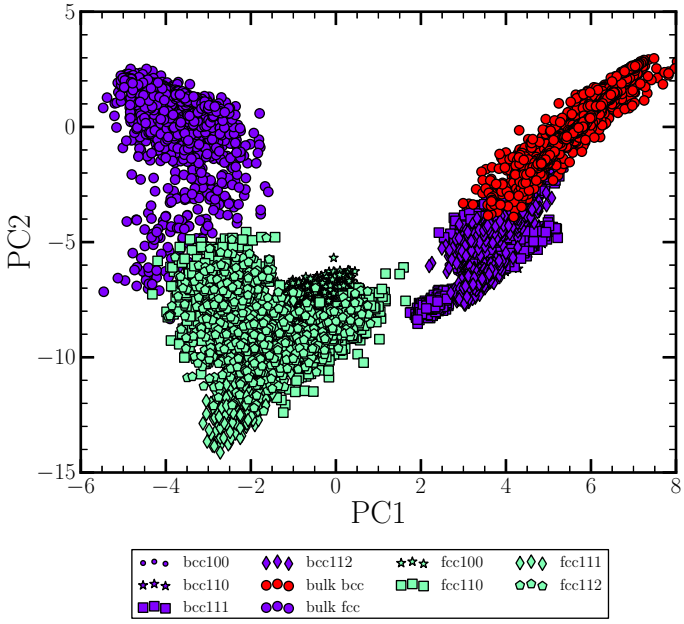


Figure 4: Two component PCA decpomposition for only surface defects and bulk atoms in FCC and BCC.

2.5. Algorithm Training and Optimization

Both a nearest neighbors (NN) and random forest ensemble (RF) algorithm[4] were trained with the bi-spectrum data for different maximum $2j$ values to evaluate performance of the bi-spectrum generation as a function of training set accuracy. Our final training data set resulted in 21 different labeled categories. Since some categories had significantly more members then others (bulk FCC, BCC for instance), we chose to evenly sample from each category to generate the final data which the algorithms were trained on. The timing and performance results for different numbers of bi-spectrum coefficients is shown in Fig. 5. The NN algorithm used $k = 5$ nearest neighbors, and the

RF algorithm used at least 10 samples per leaf with 1000 trees in the ensemble. We see that adding bi-spectrum coefficients progressively increases the training set accuracy, where values beyond $2j = 8$ see diminishing returns in terms of training set accuracy. The bi-spectrum generation algorithm seems to scale with $O((2j)^2)$, giving an optimal value for speed and accuracy between $2j = 3$ and $2j = 4$.

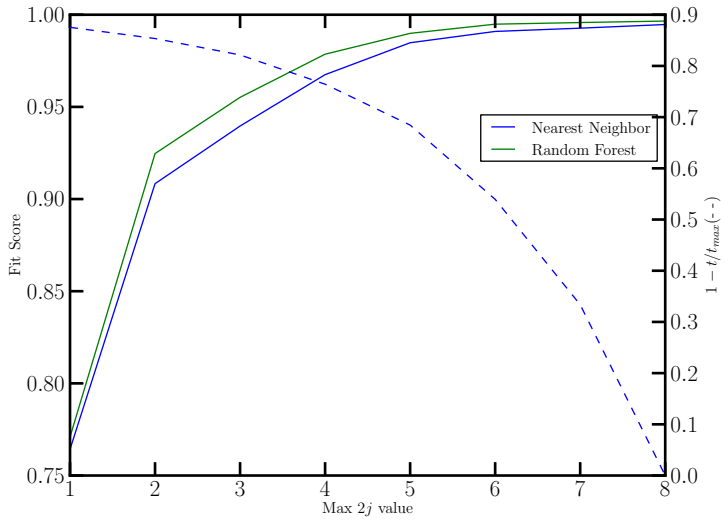
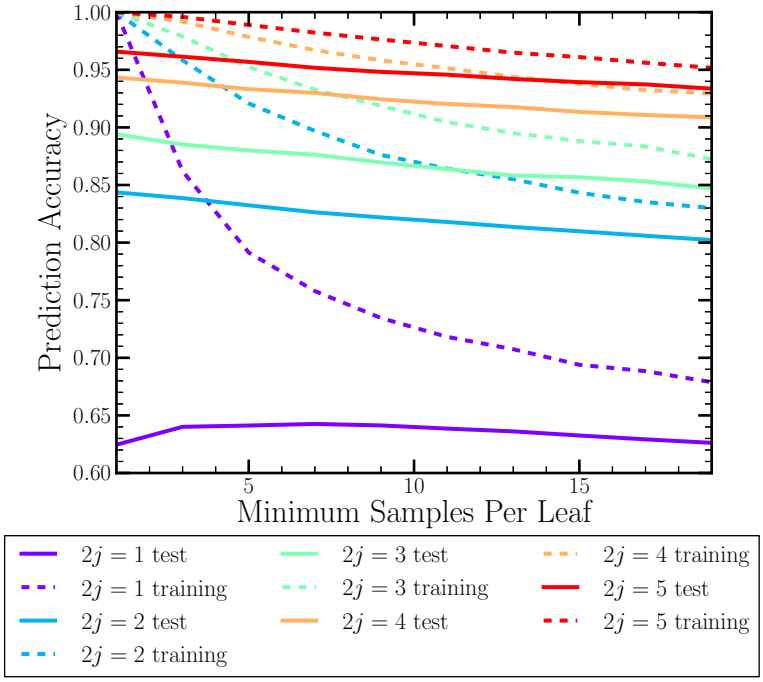


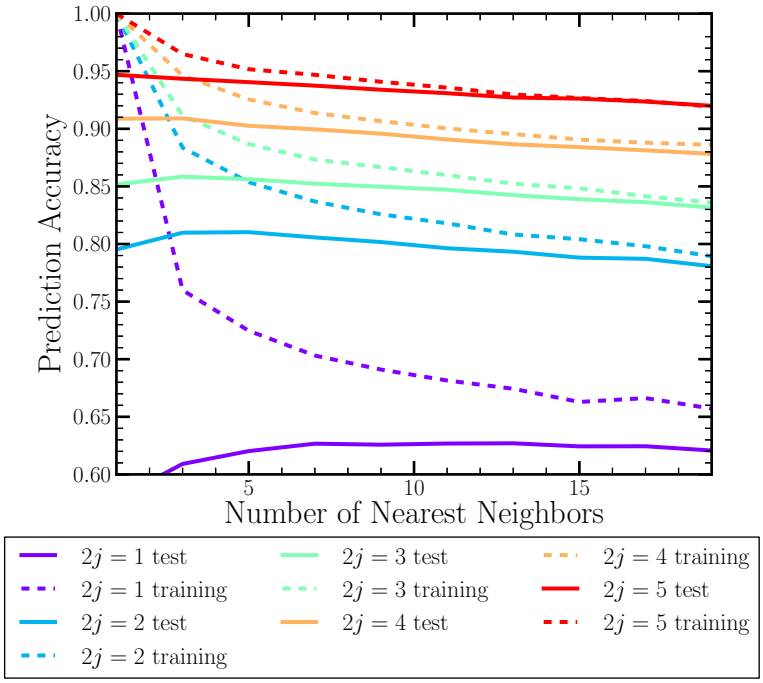
Figure 5: Performance information for BiDef algorithms as a function of the maximum $2j$ value used. Timing information is given as $1-t/t_{max}$ such that 1 would be the fastest time of the $2j$ values tested. Fitted Score is a indicator of how well the training set was fit.

The way the defect identification code is currently implemented, it needs to be run as a post processing step from a LAMMPS dump file. The atomic positions are read from the LAMMPS file, and the bi-spectrum components are then generated in a manner similar to building the training set. Since the bi-spectrum components are sensitive to scaling, the input system must also provide a scaling factor to match input features with the training features that were generated using the equilibrium lattice parameter of copper. The system is scaled by this factor prior to generation of the bi-spectrum coefficients. The set of bi-spectrum coefficients must be normalizing using the same mean and standard deviation data as gathered from the training samples, so the stored values are used to normalize the current bi-spectrum components being evaluated. The classifications are then predicted using either the nearest neighbor or random forest algorithms, and an object is returned which has the unique identification for each atom. These results can be visualized in Ovito[20] as shown in the next section. This algorithm is referred to from this point as BiDef, a shortening of bi-spectrum based classifier for defect identification.

Statistical verification methods like cross validation are quite difficult for this



(a)



(b)

Figure 6: Prediction accuracy for both the training and testing set data as a function of a)the maximum leaves per node and b) the number of nearest neighbors. The data is represented for 5 different bi-spectrum coeffcient cutoffs, $max2j$ values.

algorithm since there is no objectively correct way to identify defective atoms in previously unseen data. One limited method of evaluating error is to verify correct predictions within the training data. Portions of the pre-generated training data are left out of the training procedure and then compared after training, and the left-out data referred to as the testing set. This approach is known as a bootstrapping to evaluate error. The model is trained repeatedly on random samples from the total training set, and the error is evaluated for both the training set and the testing set. The errors are averaged as the model parameters are varied to find globally optimal model parameters. The model accuracy is calculated as the fraction of correctly identified samples, the error can then be viewed as $1 - accuracy$.

The results of the bootstrapping test are shown in Fig. 6. As expected, the training data prediction accuracy decreases with both increasing number of neighbors and minimum samples per leaf node. The models are over-fit when few neighbors are used, or when a small number of samples per leaf node is required. Fig. 6 shows the results for $2j$ values from 1 to 5, which gave the model between a 2 and 21 dimensions feature space. Increased numbers of bi-spectrum coefficients give both models greater prediction accuracy.

Interestingly for the bootstrapped accuracy of the test data set, the accuracy monotonically decreases with increased numbers of nearest neighbors or samples per leaf. We had expected to see a maximum, followed by a decrease in accuracy caused by over-simplification in the models at high numbers of nearest neighbors or leaf-node members. We think that this behavior is due to the tight clustering and relative uniqueness of the defects in feature space. The defects in bi-spectrum space are so tightly clustered that we are essentially fitting the model to 21 different classification points in feature space. Therefore, a relative high variance model was necessary given the large feature space and small dispersion in training data. Since we don't have any untrained defects which fall in the 'spectrum' between two other defects, the generalization, or the bias, in the model is not too important for our application.

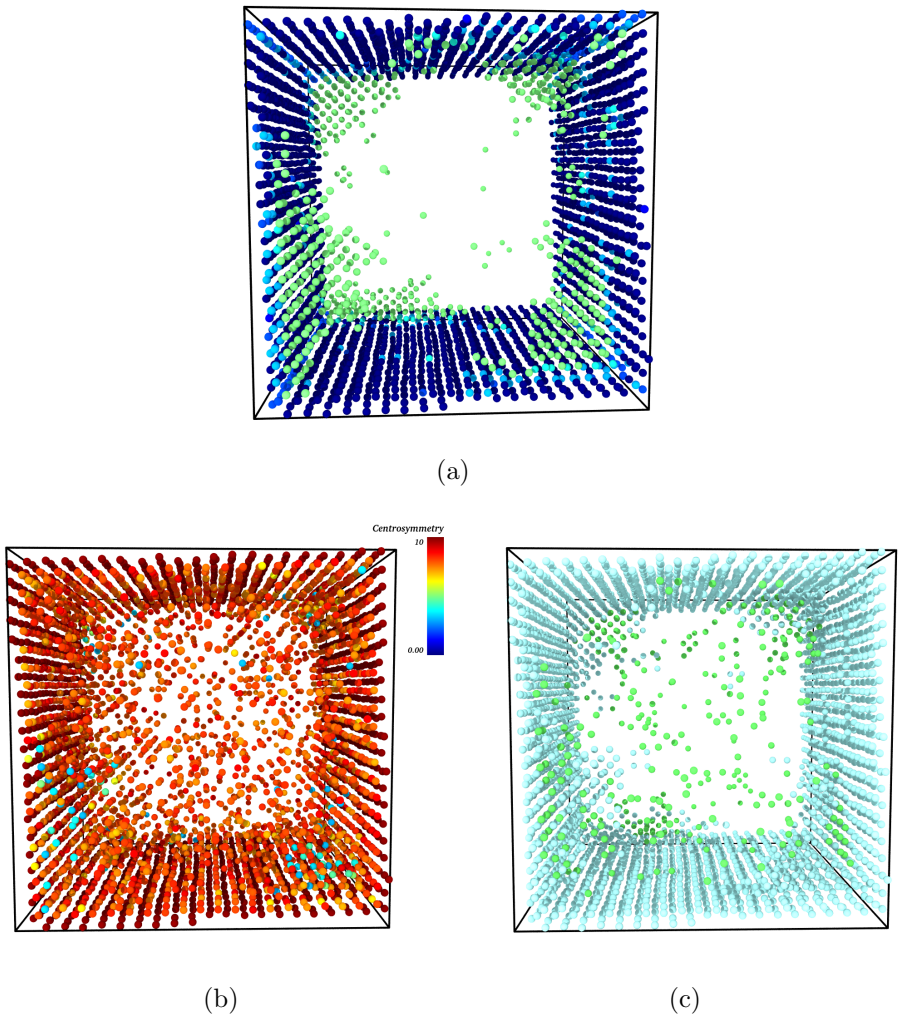
3. Results and Discussion

A qualitative comparison between CSP, CNA and bi-spectrum with random forests defect identification algorithms is shown in Fig. 7. The results are shown for a 1-d, square nano-wire of BCC iron, which was subjected to a tensile test in a direction normal to the plane of the page (along the nano-wire axis). A Nose-Hoover thermostat [9, 14] was used to control the temperature at 300K during the tensile test. A Finnis−Sinclair EAM potential[11] was used for the iron interactions. The crystals were oriented such that the free surfaces were of the $< 100 >$ type, and the crystal was strained along the [001] axis. The sample images are taken just as the yield (slip) process started. It's important to note that none of the atoms in this tensile test were used in training the machine learning algorithm, the algorithm made all classifications based on previous data of a different geometry and size scale, as detailed in the Appendix. The most

obvious difference between the machine learning algorithm, Fig. 8a, and the CSP/CNA algorithms is that the defects were able to be specified. In Fig. 8a we see that the surfaces are primarily $< 100 >$ type surfaces (dark blue), as expected from the starting cell configuration, with the other two methods the only obtainable information is that the atoms along the surface are defective relative to the bulk. Along the surfaces where the $b = \frac{a_0}{2} < 111 >$ dislocations leave the bulk, the surface atoms have shifted such that they are identified as (112) surfaces, indicating surface slip relative to the other surface sites. The corner atoms are identified as (111) surfaces, which isn't physically accurate, but indicates that the algorithm is sensitive to the reduction in coordination. It was not trained to identify anomalies beyond simple flat surfaces, so this misidentification is to be expected at this stage. A desirable quality of the machine learning algorithm is that it is evidentially resilient to noise in defect identification. When comparing the machine learned algorithm to Fig. 8b or Fig. 8c, we can see that many of the interior atoms, particularly with the CSP algorithm, are identified as defects. In reality these atoms are simply oscillating around the lattice minima due to thermal vibrations and local variations in strain, but both CNA and CSP algorithms have mis-identified many atoms.

In terms of the structure of the slipped regions in Fig. 7, some clear planes can be identified as slipped regions in the CNA and machine learned algorithm. The CSP has too much noise near the surfaces to determine where the slipped planes are located. One difficulty in identifying defects within BCC crystals is that CNA is unable to differentiate slipped regions with any specific description. In FCC systems, leading partial dislocations will leave behind HCP slip systems, which gives CNA more utility in defect identification. This disadvantage is overcome with the BiDef algorithm as it is capable of indicating that the slipped regions are BCC dislocations. Its interesting that the DXA was unable to identify the slipped regions as dislocations, it is possible they haven't slipped a full burgers vector, but they were still identified by BiDef. This implies that by adding training sets with partially slipped regions (plane misalignment similar to a stacking fault), the algorithm may be sensitive enough to identify regions which are on the cusp of becoming partial dislocations. This would open the possibility of predicting where yield begins in a simulation, and assisting in a subset of atoms to be chosen for nudged elastic band calculations[6], for instance.

Fig. 8 shows another snapshot of the same simulation after significant strain (10%) has been applied. Here we see that the BiDef algorithm identified similarly defective atoms to the CNA method, however it was able to retain the proper identification of the BCC dislocations (which now cross the entire crystal). BiDef also correctly identifies that the surface atoms have switched orientation from (100) to (112) oriented surface atoms across the twinned boundary that forms. This information is all lost on the CNA method, as all of these defective atoms are simply identified as "other". This figure demonstrates that the BiDef method is fairly resilient to large strains. This is expected as the algorithm was trained to incorporate strained defects. The CSP method has a large amount of noise at these strain values, while the slipped regions can still be



(a)

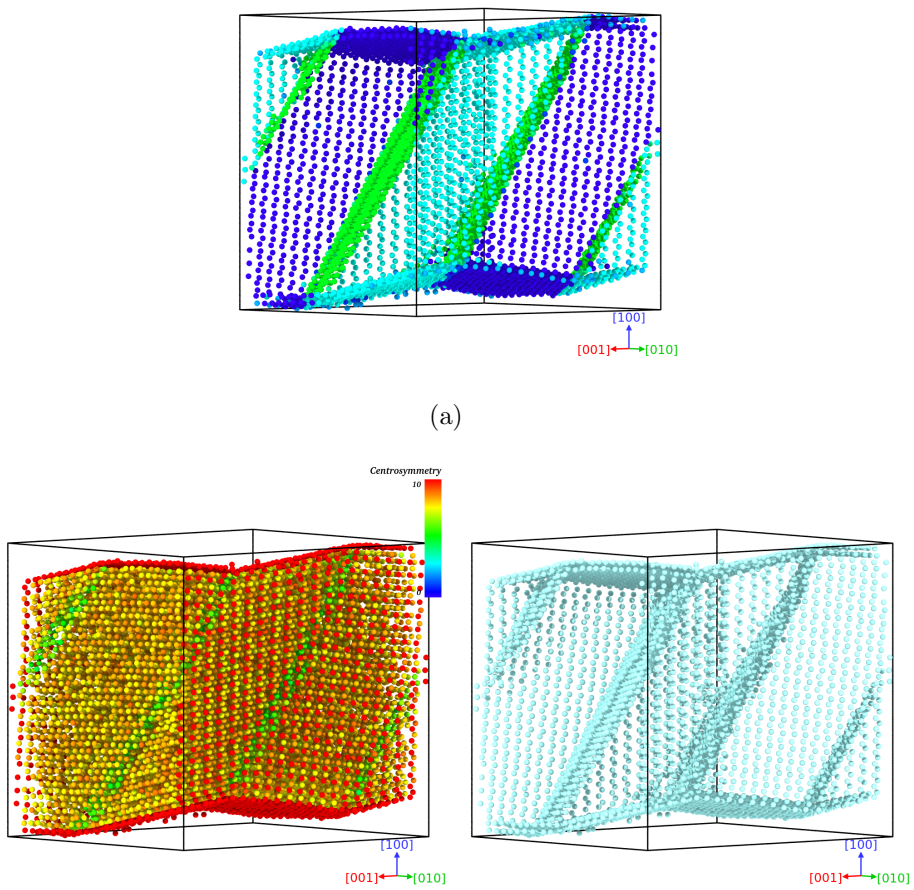
(b)

(c)

Figure 7: Example structure of tensile-strain test yeilding in BCC iron. a) Bispectrum coefficient based method, green atoms are full-bcc dislocations, dark blue are (100) bcc surfaces, medium-blue (atoms along corners) are identified as (111) BCC atoms, and the light blue atoms are (112) BCC surface atoms. b) Same cell color coded by CSP, with all atoms where CSP < 3 are removed. c) CNA colored cell, where green atoms are FCC sites, light blue atoms are unidentified sites, and all BCC atoms have been removed.

differentiated, many of the 'bulk' atoms are misidentified as defects.

Since the slip and surface defects in the yielding sample tests were planar in character, a useful way to characterize noise can be to count the number of "defective" nearest neighbors to a given atom. This way atoms with two or less defective neighbors are highly unlikely to be a legitimate defect, as their presence on a plane would indicate more than two defective neighbors. We can quantify 'noise' in these samples using by using this method. A series of tensile yielding simulations were performed at different temperatures as a more quantitative evaluation of noise with the BiDef method. Both



(a)

(b) (c)

Figure 8: Large-strain (10%) structure of tensile-strain test yeilding in BCC iron. a) Bi-spectrum coefficient based method, green atoms are full-bcc dislocations, dark blue are (100) bcc surfaces, medium-blue (atoms along corners) are identified as (111) bcc atoms, and the light blue atoms are (112) bcc surface atoms. b) Same cell color coded by CSP, with all atoms where CSP < 4.5 are removed. c) CNA colored cell, where light blue atoms are unidentified sites, and all BCC atoms have been removed.

FCC and BCC cells were constructed and pulled under tensile strain rate of $10^{10}\frac{1}{sec}$. The surfaces normal to the tensile direction were left as free surfaces, giving the cell a nanowire geometry. Samples were tested at 300 K, 400 K, 500 K and 600 K using a Nos-Hoover thermostat. Both FCC and BCC nano-wires were in 20x20x20 simulation boxes with 1-d periodicity, oriented with [100],[010], and [001] surfaces corresponding to the $\hat{x}$, $\hat{y}$ and $\hat{z}$ directions. The FCC sample was copper with the popular Mishin et al. EAM potential[12] and the BCC was iron, where we used the same interatomic potentials as previously indicated.[11]

Fig. 9 shows the percentage of erroneously identified atoms comparison between

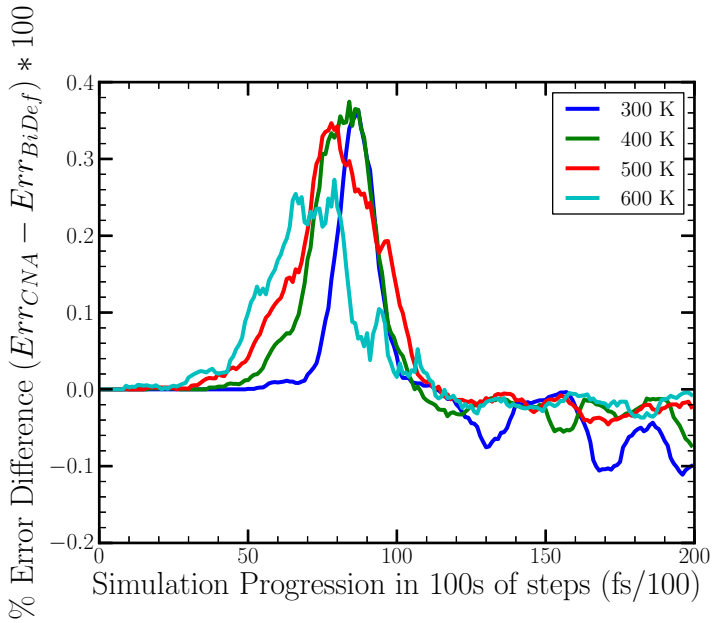


Figure 9: Error comparison between CNA and BiDef algorithms for different temperature tensile tests in a BCC sample.

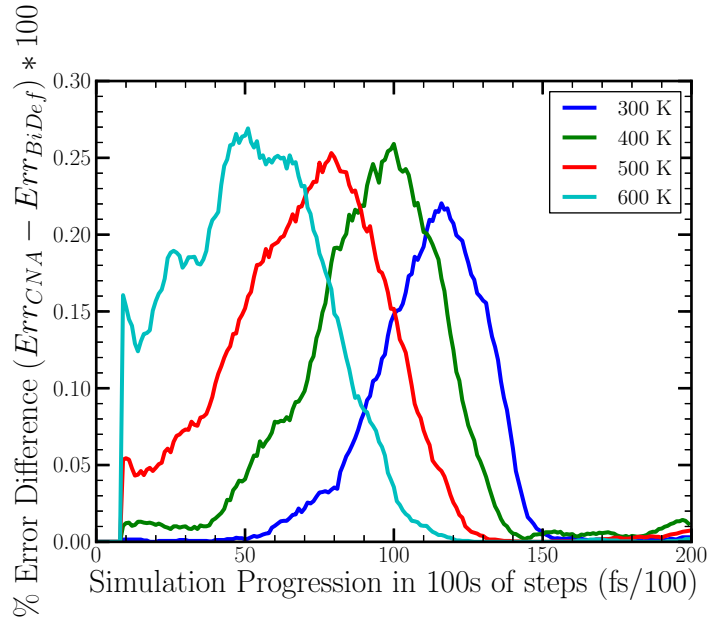


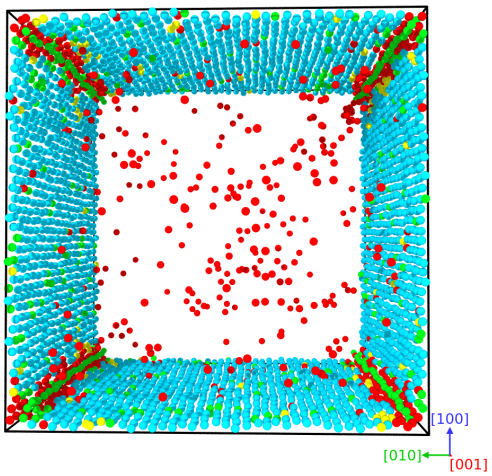
Figure 10: Error comparison between CNA and BiDef algorithms for different temperature tensile tests in an FCC sample.

CNA and BiDef for the BCC tensile test. Portions of the curve where the error percentage is positive indicate that CNA mis-identified more defect sites than BiDef, and the negative portions indicate the opposite. Prior to yielding, there is a large spike in misidentification errors in the CNA method relative to the BiDef algorithm, implying that our algorithm is more reliable at defect identification before yielding. After yielding occurs, portions of the simulation have more mislabeled sites in the BiDef algorithm than in CNA. We attribute these sites to 'holes' along planes in the material, where the algorithm was not trained at large enough strain values for identifying BCC dislocations. From visual inspection of the simulation, the mislabeled defects are almost all false-bulk assignment rather than false-defect assignments for the BiDef algorithm. Further training is likely needed for BCC defects with the BiDef method.

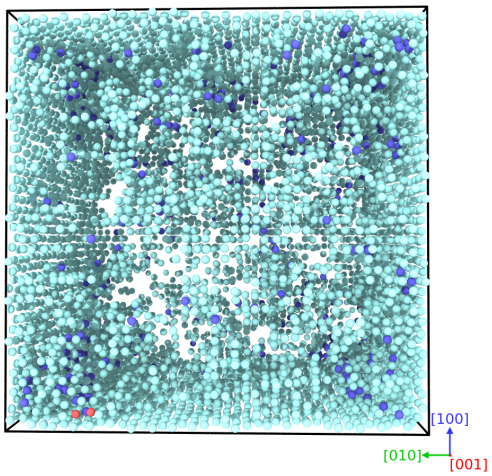
Fig. 10 gives an error comparison between CNA and the BiDef algorithm for the FCC tensile test. Using our error identification method, the misidentification error is exclusively higher when using the CNA method as compared to the BiDef algorithm. After yielding the algorithms appear similar in Fig. 10. However, when examining the defect structures, we found that CNA identified many regions as 'other' where no apparent slip had occurred. A disadvantage of our error evaluation method is apparent when large neighboring areas of atoms are mis-identified. Since they are surrounded by defect atoms, they won't be labeled as erroneous. Fig. 11a shows a snapshot of the FCC cell, for both algorithms, at 500K. It is clear from this figure that regions which were identified as 'other' sites in the CNA method, are actually bulk locations at large strain (the yield stress had yet to occur). The BiDef algorithm appears to be superior at avoiding the misclassification of these sites.

Another verification study was performed using vacancy diffusion at relatively high temperatures in a square 72 $\mathring{A}x\mathring{A}x\mathring{A}$ FCC copper nanowire simulation (1 periodic dimension). 106 vacancies were inserted randomly into the lattice, and a simulation was performed for 100 picoseconds at 900K to allow the vacancies to equilibrate. Qualitative differences in defect identification error can be observed in Fig. 12. Since non-defective (FCC) atoms are removed in Fig. 12, it is clear that significantly more defective sites are identified using the CSP (with a cutoff of CSP=5) and CNA methods than the BiDef algorithm. As with the previous simulations, the BiDef algorithm was capable of differentiating surface atoms, bulk atoms, and vacancy adjacent atoms within a margin of error.

An error calculation was necessary to quantitatively compare the algorithms. It is possible for these algorithms to falsely identify defective (false positive) and non-defective sites (false negative), so we used coordination data to evaluate the error for each algorithm. False positive sites were determined by first examining coordination, where a site labeled as defective would be incorrectly identified if the coordination number remained at 12, and then testing for the number of nearest neighbors which were also identified as defective. Defective sites without any defective nearest neighbors were likely mislabeled as a vacancy would be surrounded by atoms with a lowered coordination. False negative sites were found by examining the number of 'bulk' sites



(a)

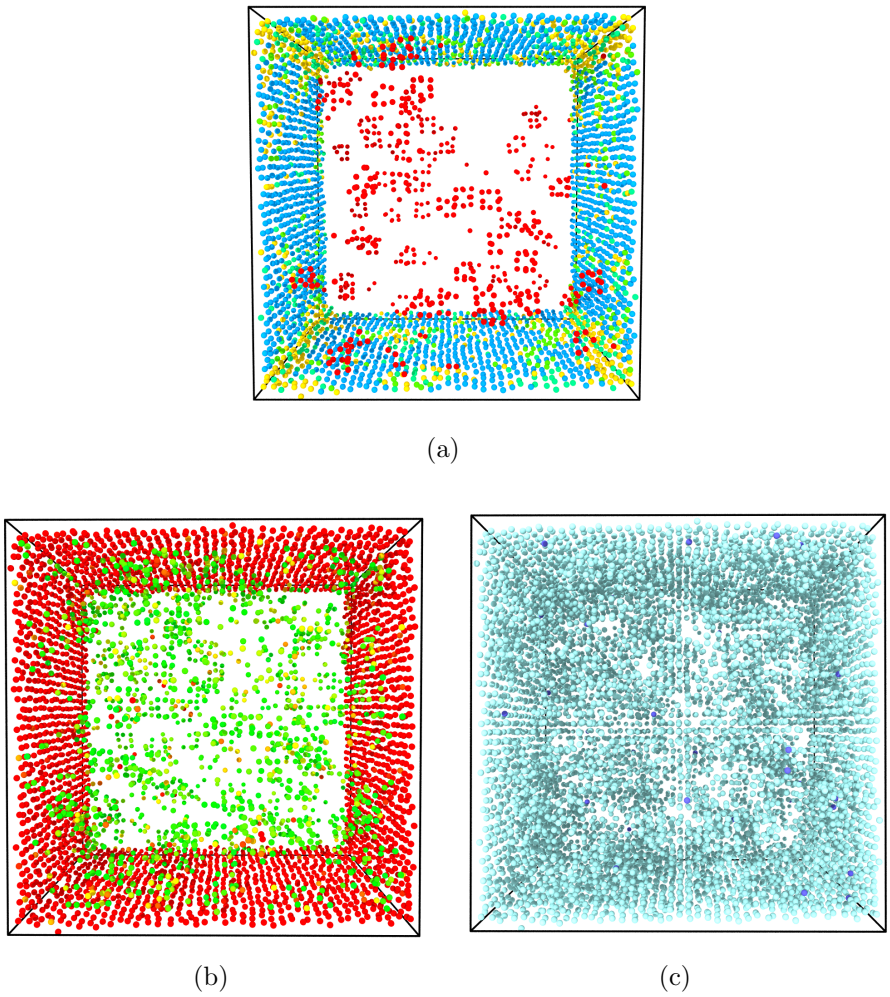


(b)

Figure 11: Pre-yield tensile FCC cell noise comparision between a) BiDef and b) CNA. This sample was at 500K and compared before slip had occurred. Atoms identified as bulk sites (FCC) are removed from the figure to observe noise in identification.

that had lowered coordination (< 12), while also having a defective nearest neighbor. Surface sites were eliminated from the error calculations for the CSP and CNA method by not counting atoms with coordination numbers less than 9 in these simulations. This actually lowers the calculated error for the CNA and CSP algorithms as surface atoms typically have lower coordination numbers than the chosen cutoff.

False negative and false positive defects were summed for each simulation technique and normalized to the number of sites in the simulation such that they could be directly compared with an error percentage, as shown in Fig. 13. While the system equilibrates



(a)

(b) (c)

Figure 12: Defect identification differences for vacancies inserted into copper FCC nanowire at 900 K. a) BiDef method, colored such that red atoms are an FCC vacancy, and yellow and blue atoms are (112) and (100) surfaces, respectively. All sites identified as FCC have been removed. b) CSP colored scheme where all atoms with a CSP value less than 5 have been removed. c) CNA colored scheme with all atoms identified as FCC removed, leaving the eggshell color as non-identified lattice structure.

($<$ 20 ps), we see that the errors are very low, indicating that thermal fluctuations produce a majority of the erroneous classifications in all three methods. Furthermore, while the temperature is increasing, both CNA and CSP methods produce less erroneous classifications than the BiDef method, up until about 7 ps. After this point, the thermal fluctuations produce much larger errors in the CNA and CSP methods. A plateau is reached upon equilibration, near 30 ps, where the errors remain constant as the simulation progresses. At this point the BiDef method had an error of $2.58 \pm 0.16\%$, the CNA method had an error of $9.91 \pm 0.23\%$, and the CSP method had an error of $8.29 \pm 0.23\%$. Even when considering both false negative and false positive defect

identification, the BiDef method was considerably more accurate at finding atoms adjacent to vacant sites. This is compounded with the advantage that the BiDef method is able to determine these atoms as a separate classification from the surface and edge atoms, without further coordination analysis. This would allow for tracking of just the vacancy adjacent atoms during a simulation.

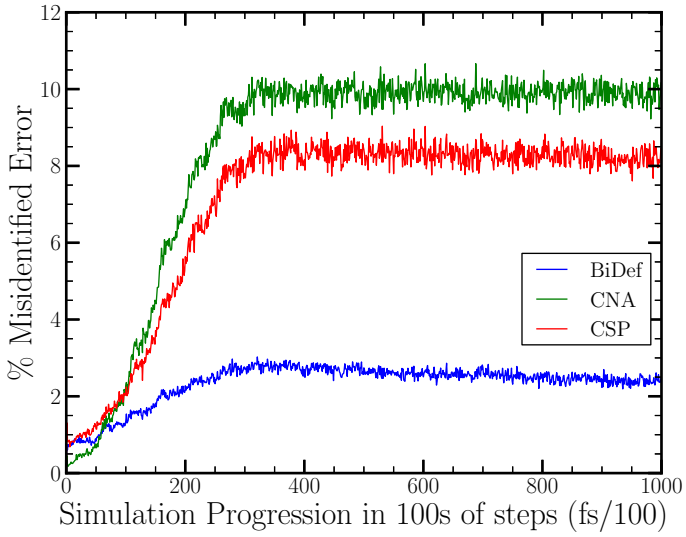
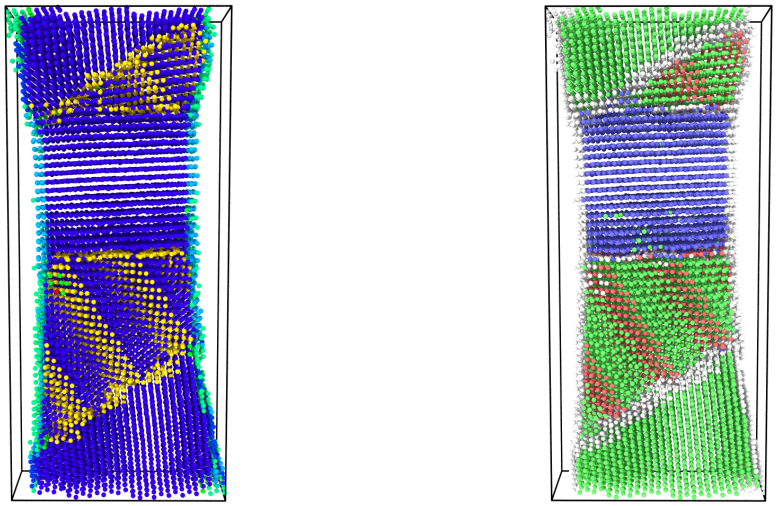


Figure 13: Misidentification error comparison between CNA, CSP, and BiDef algorithms in a vacancy filled copper nano-wire.

The BiDef method has several notable disadvantages for general use. Firstly, as expected, it will struggle with samples which show behavior totally unseen in the training data. Fig. 14 shows the results of BiDef identification as compared to CNA for the interior of copper nano-films showing an FCC-BCC interface yielding structure. As can be seen in Fig. 14a, the method succeeds in finding some bulk FCC sites, the surfaces and stacking faults, however it misses the entire BCC region found by CNA in Fig. 14b. Even though the method was trained to find both FCC and BCC structures, this failure is likely due to the scaling differences between the two lattices in the same simulation. Both FCC and BCC defects were normalized to the equilibrium lattice parameter of copper in the training data. Since the bi-spectrum is sensitive to lattice scaling, it is unable to identify BCC and FCC structures in the same simulation with different scales. The un-scaled BCC atoms represent an unmapped portion of feature space at the interface. Training data would need to include BCC-FCC interface atoms to be properly classified.

The generation of bi-spectrum components is quite costly, and scales poorly with the $2j$ bi-spectrum limits. On a reasonable machine (8-cores) with a $2j$ value of 1, the LAMMPS library call to generate the components takes 25.8 seconds for 350,000 atoms, and the generation takes 206.8 seconds for the same system with a $2j$ value of 8. All simulations performed here were done with a nearest neighbor cutoff of two times the



(a) (b)

Figure 14: Copper tensile test resulting in an FCC-BCC interface structure for a) BiDef with dark blue as bulk FCC atoms, yellow as partial dislocations/stacking fault atoms, and blue/green for [100]/[112] surfaces b) CNA with green as bulk FCC atoms, red as HCP atoms, blue as BCC atoms and white as 'other' atoms.

lattice parameter, and this produces an additional limitation in the method. Defect structures must be contained within the cutoff radius, and training libraries must be created for new cutoff values should they be used. Furthermore the larger cutoff radii will slow the generation of the bi-spectrum down linearly.

4. Conclusions and Outlook

From the preliminary tests presented here, it appears that using machine learning for defect identification is a promising method which avoids several issues presented in other methods. Specifically, it's capable of uniquely identifying defects, has relative intolerance to temperature and strain, and can be modified to examine any new type of defect. The power of this method does come with some disadvantages, it is quite slow in comparison to other methods. This is primarily due to the time it takes to generate bi-spectrum components. Although this process can be parallelized to increase performance. The method is also limited to studies where the defective sites are similar to previously seen training data. It will not generalize to new systems without additional data. We examined both random forest and nearest neighbors as learning algorithms, and found random forests had superior performance, while being slower to execute. In our case studies, we found that the BiDef method was less noisy than both CNA and CSP for identifying tensile defects during yielding. However, the algorithm slightly underperformed CNA in identifying all of the slipped regions after yielding in BCC

systems. Hopefully, the flexibility of this methods encourages others to contribute more complex defects to its training library such that it becomes more capable with time.

5. Appendix

Included here is a description of each of the training prototype cells used for the nearest neighbor and random forest machine learning algorithms. Each of these prototype cells were thermalized (random displacements added to the atomic positions) and strained to produce distributions in bi-spectrum space. All cells were generated using the lattice parameter for copper, 3.597 $\mathring{A}$, later inputs were scaled to this value to avoid issues with lattice parameter differences. The cutoff used for bi-spectrum generation was two times this lattice parameter. All systems were initially generated in LAMMPS, where the bi-spectrum components were generated using a LAMMPS dynamic library. The components were then parsed and used in python code for the rest of the analysis. For all systems listed, no relaxation was allowed to occur before calculating bi-spectrum components.

Firstly, surface prototypes were made for both FCC and BCC structures, one with $[1,1,1]$, $[\bar{1},1,0]$, $[\bar{1},\bar{1},2]$ orientations along the $\hat{x}$, $\hat{y}$, and $\hat{z}$ axes and another with $[0,1,0]$,$[0,0,1]$,$[1,0,0]$ alignment. The periodicity was simply moved to each surface to capture the changes in coordination. The cells were built with 6 unit cells in each direction, scaled such that directions with different line-densities still produce complete unit cells.

Vacancy defect prototypes were made by constructing FCC and BCC systems with 6 unit cells in the $\hat{x}$, $\hat{y}$, and $\hat{z}$ directions. All boundaries were periodic. One atom at random was removed from each since the defective sites are symmetric within the crystal. Interstitial sites were made in the same sized system, however both octahedral and tetrahedral sites needed to be considered for both FCC and BCC cells. To construct these, one atom was chosen at random, and a new atom was generated on a vector representing an interstitial site for that crystal structure.

The partial dislocations were of the $b = \frac{a_0}{6} < 112 >$ type for FCC. The full dislocations were of $b = \frac{a_0}{2} < 110 >$ and $b = \frac{a_0}{2} < 111 >$ for FCC and BCC cells, respectively.

The dislocation prototypes were initially made with the atomsk software. For the full ($b = \frac{a_0}{2} < 110 >$) dislocations in the FCC cell, two edge dislocations were inserted into a repeating slab (periodic in the direction of the dislocation line), as shown in Fig. 15. The cell was oriented with a non-repeating $[0\bar{1}1]$ and $[100]$ surfaces, and a periodic $[011]$ direction. The atoms near the free surfaces were masked to avoid complicating the training data. Having multiple defects would have made it very difficult to use CSP or CNA as an initial labeling tool. DXA properly identified the defect cores, but CSP was used for labeling the defect since it was sensitive enough to find more core atoms, as opposed to the CNA identified atoms. The full BCC dislocation was created in the same way, however it was oriented with non-periodic $[111]$ and $[\bar{1}\bar{1}2]$ directions and a

periodic [110] direction. Once again the free surfaces were masked. The BCC prototype is shown in Fig. 16 Both dislocations were 20 x 15 x 4 unit cells in each direction.

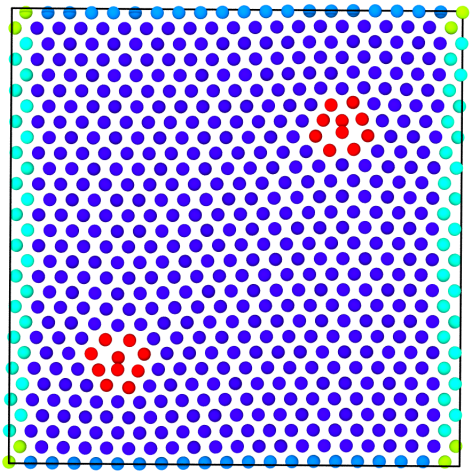


Figure 15: Full dislocation in a FCC crystal.

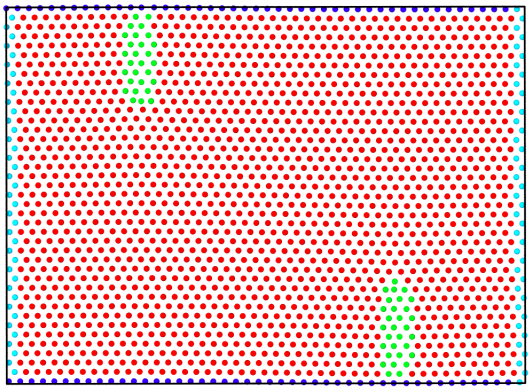


Figure 16: Full dislocation in a BCC crystal.

Partial dislocation structures were made by deforming two slabs and placing them next to each other using atomsk. The cell was periodic in the [110], [$1\bar{1}1$], and [$1\bar{1}\bar{2}$] directions, and an upper and lower slab was created where each was deformed by a half burgers vector ($b = \frac{a_0}{6} < 112 >$) in compression and tension. Then the two slabs were combined, creating two opposite partial dislocations in the middle, separated by a stacking fault. The crystal size was 35 x 4 x 3 unit cells.

To expose the training set to partial dislocations when nucleated from surfaces, a nanowire tensile test was performed, and the defect structures after yielding were used as training data. Only an FCC cell was used with 13000 atoms in a nano-film configuration. The $\hat{x}$ direction was 81 $\mathring{A}$, the $\hat{y}$ direction was 22.3 $\mathring{A}$ and the $\hat{z}$ direction was 129 $\mathring{A}$ oriented with the [100], [010], and [001] direction respectively. The $\hat{x}$ direction was the only non-periodic direction. The sample snapshots used were at 8, 10 and 15 % strain. CNA was used to label the defective structures, where HCP atoms were considered to be stacking fault atoms.

Grain boundaries were incorporated by making one large polycrystalline cell, which contained a total of 12 grains. Atomsk was used to generate the initial cell, which was 150 $\mathring{A}$ long on each side. The cell was periodic in all three dimensions, such that the only defective structure were the grain boundaries within the system. CNA was used to label defective atoms in this configuration. No distinction was made between grain boundary types such as low-angle, or twin boundaries. This is an area where improvements could be made, as some grain structures are quite similar to dislocation arrays, which could make overlap in bi-spectrum space when training the algorithm.

6. Bibliography

[1] Albert P Bartok, Mike C Payne, Risi Kondor, and Gabor Csanyi. Gaussian Approximation Potentials : The Accuracy of Quantum Mechanics , without the Electrons. *PRL*, 104(April):136403, 2010.

[2] Albert Bartók-Pártay. *The Gaussian Approximation Potential*. Springer-Verlag, Berlin, Heidelberg, 1 edition, 2010.

[3] Jon Louis Bentley. Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*, 18(9):509–517, 1975.

[4] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, October 2001.

[5] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[6] Graeme Henkelman, Blas P. Uberuaga, and Hannes Jonsson. A climbing image nudged elastic band method for finding saddle points and minimum energy paths. *The Journal of Chemical Physics*, 113(22):9901, 2000.

[7] Pierre Hirel. Atomsk: A tool for manipulating and converting atomic data files. *Computer Physics Communications*, In Press:1–8, August 2015.

[8] J Dana. Honeycutt and Hans C Andersen. Molecular dynamics study of melting and freezing of small Lennard-Jones clusters. *The Journal of Physical Chemistry*, 91(19):4950–4963, 1987.

[9] William G Hoover. Canonical dynamics: Equilibrium phase-space distributions. *Physical Review A*, 31(3):1695–1697, 1985.

[10] Cynthia Kelchner, S. Plimpton, and J. Hamilton. Dislocation nucleation and defect structure during surface indentation. *Physical Review B*, 58(17):11085–11088, November 1998.

[11] M. I. Mendelev, S. Han, D. J. Srolovitz, G. J. Ackland, D. Y. Sun, and M. Asta. Development of new interatomic potentials appropriate for crystalline and liquid iron. *Philosophical Magazine*, 83(35):3977–3994, December 2003.

[12] Y Mishin, M J Mehl, D A Papaconstantopoulos, A F Voter, and J D Kress. Structural stability and lattice defects in copper: \textit{Ab initio} , tight-binding, and embedded-atom calculations. *Phys. Rev. B*, 63(22):224106, May 2001.

[13] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*.

Adaptive computation and machine learning series. MIT Press, Cambridge (Mass.), London, 2012.

[14] Shichi Nosé. A molecular dynamics method for simulations in the canonical ensemble, 1984.

[15] Stephen M Omohundro. Five Balltree Construction Algorithms. pages 1–22, 1989.

[16] Steve Plimpton. Fast Parallel Algorithms for Short Range Molecular Dynamics. *Journal of Computational Physics*, 117(June 1994):1–42, 1995.

[17] T Schablitzki, J Rogal, and R Drautz. Topological fingerprints for intermetallic compounds for the automated classification of atomistic simulation data. *Modelling and Simulation in Materials Science and Engineering*, 21(7):075008, October 2013.

[18] Alex J Smola and Bernhard Scholkopf. A tutorial on support vector regression. *Statistics and Computing*, 14:199–222, 2004.

[19] P. J. Steinhardt, D. R. Nelson, and M. Ronchetti. Bond-orientational order in liquids and glasses. , 28:784–805, July 1983.

[20] Alexander Stukowski. Visualization and analysis of atomistic simulation data with OVITOthe Open Visualization Tool. *Modelling and Simulation in Materials Science and Engineering*, 18(1):015012, January 2010.

[21] Alexander Stukowski. Structure identification methods for atomistic simulations of crystalline materials. *Modelling and Simulation in Materials Science and Engineering*, 20(4):045021, June 2012.

[22] Alexander Stukowski and Karsten Albe. Extracting dislocations and non-dislocation crystal defects from atomistic simulation data. *Modelling and Simulation in Materials Science and Engineering*, 18(8):85001, 2010.

[23] A P Thompson, L P Swiler, C R Trott, S M Foiles, and G J Tucker. A Spectral Analysis Method for Automated Generation of Quantum-Accurate Interatomic Potentials. *arxiv*, 2014.

[24] D.A. Varshalovich, A.N. Moskalev, and Khersonskiĭ. *Quantum Theory of Angular Momentum: Irreducible Tensors, Spherical Harmonics, Vector Coupling Coefficients, 3nj Symbols*.

[25] Qing Zhang, Tahir Çan, Adri van Duin, William Goddard, Yue Qi, and Louis Hector. Adhesion and nonwetting-wetting transition in the Al/α-Al2O3 interface. *Physical Review B*, 69(4):045423, January 2004.

[26] Ji Zhu, Hui Zou, Saharon Rosset, and Trevor Hastie. Multi-class AdaBoost. *Statistics and ITS Interface*, 2:349–360, 2009.

[27] J. A. Zimmerman, C. L. Kelchner, P. A. Klein, J. C. Hamilton, and S. M. Foiles. Surface step effects on nanoindentation. *Phys. Rev. Lett.*, 87:165507, Oct 2001.