Additive Manufacturing 36 (2020) 101641

Contents lists available at ScienceDirect

Additive Manufacturing

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

Review

Invited review: Machine learning for materials developments in metals additive manufacturing

N.S. Johnson [a,b], P.S. Vulimiri [c], A.C. To [c], X. Zhang [a], C.A. Brice [a], B.B. Kappes [a,*], A. P. Stebner [a,*,1]

[a] Alliance for the Development of Additive Processing Technologies, Colorado School of Mines, Golden, CO 80401, USA
[b] Los Alamos National Laboratory, Los Alamos, NM 87544, USA
[c] Department of Mechanical Engineering & Materials Science, University of Pittsburgh, Pittsburgh, PA 15261, USA

A B S T R A C T

In metals additive manufacturing (AM), materials and components are concurrently made in a single process as layers of metal are fabricated on top of each other in the near-final topology required for the end-use product. Consequently, tens to hundreds of materials and part design degrees of freedom must be simultaneously controlled and understood; hence, metals AM is a highly interdisciplinary technology that requires synchronized consideration of physics, chemistry, materials science, physical metallurgy, computer science, electrical engineering, and mechanical engineering. The use of modern machine learning approaches to model these degrees of freedom can reduce the time and cost to elucidate the science of metals AM and to optimize the engineering of these complex, multidisciplinary processes. New machine learning techniques are not needed for most metals AM development; those used in other sects of materials science will also work for AM. Most prolifically, the density functional theory (DFT) community has used many of them since the early 2000s for evaluating numerous combinations of elements and crystal structures to discover new materials. This materials technologies-focused review introduces the basic mathematics and terminology of machine learning through the lens of metals AM, and then examines potential uses of machine learning to advance metals AM, highlighting the many parallels to previous efforts in materials science and manufacturing while also discussing new challenges and adaptations specific to metals AM.

1. Motivation

Metals additive manufacturing (AM) has created a paradigm shift in the way metal components are manufactured; materials and parts are fabricated simultaneously using a single machine, highly complex geometries are possible, and local variations of microstructure-property relationships may be realized through local process variations. Although decades of scientific and engineering work in industry, academia, and government have resulted in the commercialization of metals AM technologies, the consistency and quality of parts and materials are still open challenges for many applications. In recent decades, Integrated Computational Materials Engineering (ICME) approaches have proven to accelerate the development and adoption of materials technologies [1]. Traditionally, ICME approaches incorporate physics-based experimental data with simulations that span different length and time scales. However, for metals AM, much of the physics are still being discovered; hence, the development of comprehensive, computationally feasible physics-first approaches to ICME are still an open challenge. The diverse array of promises and problems in AM has resulted in a field of study that is rich with data – so much so that our ability to store and analyze the data is challenged. At the same time, this wealth of data is motivating a paradigm shift to incorporate machine learning into ICME approaches.

1.1. Background

The 20th century saw the maturation of materials science and engineering as a field of study, enabling targeted materials discoveries and innovations for specific applications. Over the past several decades, materials development cycles have greatly accelerated by formulating materials problems through the process-structure-property-performance paradigm [1,2].

The process-structure-property-performance (PSPP) paradigm is a core philosophy in materials science and engineering that governs how the manufacturing of a material determines its ability to be used in different engineering applications. The PSPP relationships break down

* Corresponding authors.
E-mail addresses: bkappes@kmmd.io (B.B. Kappes), aaron.stebner@gatech.edu (A.P. Stebner).
[1] **Present Address**: Woodruff School of Mechanical Engineering, School of Materials Science and Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA

https://doi.org/10.1016/j.addma.2020.101641
Received 1 June 2020; Received in revised form 20 August 2020; Accepted 23 September 2020
Available online 19 October 2020

materials development into four key areas of scientific and engineering interest [2].

In AM, the processing of a material is dictated by the thermal, mechanical, and chemical changes experienced during its manufacture. Controllable machine parameters like energy density of the heat source, the path in which material is deposited or fused, the order in which part layers are manufactured, or the location of parts on the build plate are determining factors of the material process history. Table 1 shows many of the controllable parameters common to laser-based additive manufacturing systems. The choice of these parameters largely impacts the processing history. The true processing history, however, is better described by the thermal history of the build volume, both during manufacture and post-processing, the mechanical forces it experiences, and any chemical reactions that occur in or on the part. Processing *routes* are often discussed in AM and typically refer to beneficial or detrimental processing histories that impact the part's structure.

The structure of a material is a wide-ranging concept that spans many length scales. Structure can refer to the crystallographic structure at the atomic scale, to the morphology and orientation of grains at the mesoscale, to the geometry being manufactured at the macroscale. Microstructure is a term often used in materials science referring to a specific subset of the material structure. Microstructure for metals most commonly refers to grain and sub-grain level information like material phases, grain morphologies, texture, and any defects like pores or dislocations that might be present. Microstructures are often considered in analysis of material structures because they fundamentally dictate a material's properties.

The properties of a material are characteristics that determine its qualities. Properties of metals AM parts that have been of interest are wide-ranging and they vary depending on the desired engineering application of the part. Mechanical properties are some of the most studied for AM metals since the majority of metals applications are structural. Other properties of interest include thermal conductivity, which determines the heat transfer through an AM part, chemical properties, like corrosion resistance, and optical properties, like reflectivity.

The performance of a part is its ability to be successfully implemented in an engineering application. Performance can be viewed through the lifetime of an AM part when subjected to the mechanical, thermal, chemical, etc., forces it will experience. Early additively manufactured alloys showed degraded-to-comparable static properties compared to traditionally manufactured alloys [3]. Further research and development improved the static properties of AM materials, yet high microstructure variability and defect density can still cause AM material to fail unexpectedly in fatigue limited applications [4,5]. Some recent AM developments have resulted in material properties that exceed those of traditionally manufactured materials [6–11]. Ultimately AM processes are unique relative to other metal fabrication techniques and it is difficult to make fair comparisons regarding performance across various manufacturing methods. When properly designed, AM parts can meet the intended performance needs in a wide variety of end-use applications. The large combinatorial space of manufacturing options in AM often obfuscates how proper design choices can be made.

Table 1
A possible design space for laser powder bed fusion additive manufacturing. There are over 10^4 possible combinations of machine inputs, based on the listed ranges and step sizes. Any possible combination of these parameters is a point in the design space.

Parameter	Range	Step size	Levels
Power	100–200 W	10 W	10
Scan speed	500–1000 mm/s	100 mm/s	5
Spot size	50–100 μm	10 μm	5
Energy density	1–5 J/mm^2	1 J/mm^2	5
Sample build direction	0–180°	90°	3
Amount of recycled powder	0–100%	10%	10
Hatch spacing	0.1–0.50 mm	0.1 mm	5

The materials scientist interacts with the process-structure-property paradigm in traditionally manufactured materials. Traditional material manufacturing can be phrased in a cause-and-effect relationship between process, structure, and property. Once the material has been developed and characterized by the materials scientist or engineer, another engineer then considers the property-performance linkage. Since material is made separately from an engineered part in traditional manufacturing, the PSPP paradigm can be broken up into these two separate sets of relationships. In AM, the material and the part are made simultaneously. Simultaneous material-part manufacturing motivates consideration of linkages across the entire PSPP paradigm. The ICME approach to materials science is focused on modeling, bridging, and predicting relationships throughout the PSPP paradigm.

Computational materials science and engineering has enabled the prediction of microstructure from processing and of properties from microstructure, reducing the need for costly and time consuming experimentation in discovering or developing a new material and/or its manufacturing. Today, ICME approaches tightly integrate physics-based computational models into the industrial design process, allowing the desired performance requirements of a part to guide the design of a material. Alloy specific examples include low-Rare Earth Ni superalloys for better turbine performance [12] and lower cost and radioactive element free Ferrium S53 alloy designed for corrosion-resistant landing gears [13]. Both cases reduced materials innovation timelines from decades to years, demonstrating the practical capability of designing and qualifying new materials within an industrial product development cycle. Generalizing and accelerating this capability across different industries and materials is a primary goal of the Materials Genome Initiative (MGI) [14].

Predicting PSPP linkages in metals AM is difficult with existing physics-based ICME approaches. The physics of AM processes are more complex than traditional fabrication methods, like casting, as they involve rapid solidification, vaporization and ingestion of volatile elements, and complex thermal history that consists of dozens of heating and cooling cycles, each one different. Furthermore, all of these additional complexities vary from one location to another within a part, and from part to part within a build volume. For AM, physics-based ICME tools have been mostly developed through attempts to adopt legacy manufacturing models to AM data, with some success. However, today's relatively low cost and time for performing AM processing experiments has led to metals AM development being largely combinatorial, with a chief strategy of adopting AM processing to legacy alloys that were developed for other types of manufacturing using extensive design of experiments.

It is with awareness of the large amounts of data being generated in AM through these combinatorial development cycles that machine learning (ML) has been targeted to accelerate AM innovations and their commercialization. Machine learning as a technology development accelerator has shown wide application in recent years across fields including finance [15], molecule design for genomics, chemistry and pharmacology [16], social networking [17] and, most relevant to this review, materials science and engineering [18–20]. Still, the use of ML in materials science was relatively limited for a variety of reasons, especially the lack of large curated datasets amenable to existing ML methodologies. Through the work done under the MGI, this data limitation was identified as a primary impediment to future materials innovations [14]. In response, there has been significant recent investment in materials database developments to better enable materials data informatics innovations. It is now recognized and accepted that ML frameworks can couple legacy physics-based ICME tools with experimental data to produce more accurate process-structure-property models and to automate the iteration of designed experiments for model improvement and optimized materials [21–24].

We proceed to review how the paradigm shift from purely physics-based to coupled physics-based/data-driven ICME approaches can be

made through solving metals AM challenges. We begin by phrasing terms and ideas from AM in ways that are compatible with machine learning. We provide a basic review of machine learning algorithms and how they can be applied to additive manufacturing. Following this introduction to using ML for AM problems, we review other uses of machine learning in materials science and engineering and state the uses of such approaches for solving AM challenges.

2. Phrasing additive manufacturing as a machine learning problem

While machine learning may seem abstract at first, it can be expressed and understood in plain terms. Many of the tenets and frameworks for machine learning are based in mathematical operations that are likely familiar to any scientist or engineer, but applied in new ways. In this section, we proceed to define the basic terminology and classes of machine learning and data. A list of machine learning algorithms used in the papers cited in this review can be found in Table 2. The following section details general terminology and intuition for the application of AM. Specific ML algorithms are then introduced specific to contextual AM examples in Section 3.

Machine learning algorithms are mathematical constructs that may be used as scientific and/or engineering tools when warranted. They are not appropriate for all science and engineering problems – just as finite element simulations should not be used to study the mechanics of discrete interfaces or single atomic bonds between two atoms or DFT should not be used to simulate mm-sized polycrystals, machine learning algorithms should not be used to model data that lack statistical correlations. Hence, the first questions every scientist and engineer considering the use of machine learning approaches should ask and answer is: "how are the data statistically distributed?", and "are there statistical correlations between the data features of interest?" Once this is complete, then a researcher can decide if ML is appropriate.

If the data lack clear statistical correlations using basic probability analyses, machine learning is not a "magic box" that can suddenly make such correlations evident. Similarly, if the statistical distributions of the data are featureless except for an occasional outlier, machine learning cannot meaningfully fit a model that is based on statistical distributions.

Today, many scientists and engineers are embracing the approach that "we will machine learn it," without understanding how to evaluate if machine learning is an appropriate tool to apply to a problem or not. One unpublished example in AM of a problem that ML is not well suited for is building a model to predict the location of a maximum pore within a powder bed laser fusion build. A maximum pore is a statistical outlier – usually one of thousands-to-millions, depending on the size of the part being built. Even though the pore may occur in the exact same position of the build volume if the same part is built over-and-over again (i.e., it is highly repeatable), the fact that it is a statistical anomaly means that nearly all machine learning algorithms are built to ignore it. Once this understanding is at hand, then a researcher can decide if ML is appropriate.

Still, most data of interest in metals AM have strong statistical features, as we will proceed to discuss in more detail in the examples given in this review. Once some basic statistical analysis of the data of interest has been performed and it has been determined that there are quantifiable correlations between the inputs and outputs, or across different inputs, and that there are also statistical features that describe the distributions of the data, then a researcher can proceed to consider data featurization and processing, and then tune and evaluate the performances of machine learning models to find the best performers. We proceed to describe these techniques in more detail, after defining some basic terminology used in this article.

Table 2
Several of the most widely used machine learning algorithms in materials science.

Class of algorithm	Examples	Applications	Strengths	Constraints
Weighted neighborhood clustering	Decision trees, Random forest, k-Nearest neighbor	Regression, Classification, Clustering and similarity	These algorithms are robust against uncertainty in data sets and can provide intuitive relationships between inputs and outputs. See Ref. [25] for a primer on clustering	They can be susceptible to classification bias toward descriptors with more data entries than others
Linear dimensionality reduction	Principle component analysis (PCA), Support vector regression (SVR), Nonnegative matrix factorization (NMF)	Experimental design, model dimensionality reduction, model or experimental input/output visualization, descriptor analysis, regression	This type of algorithm can produce orthogonal basis sets that reproduce the training data space. They can also provide quick and accurate regression analysis. For a primer on PCA specifically, see Ref. [26]	The relationships studied must be linear in nature, and these algorithms are susceptible to bias when descriptors are scaled differently
Nonlinear dimensionality reduction	t-SNE, Kernel ridge regression, Multidimensional metric scaling	Experimental design, model dimensionality reduction, model or experimental input/output visualization, descriptor analysis, regression	These algorithms are robust against nonlinear input/output relationships and can help visualize similarity in high dimensional relationships. For accessible examples, see Refs. [27,28]	Interpretation of high dimensional similarity can be difficult; while these algorithms are useful for visualizing relationships interpreting the *why* of the relationship found is difficult. Global relationships can also be lost when nonlinear dimensionality reduction results are projected onto lower-dimensional spaces
Search algorithms	Genetic algorithms (GA), Evolutionary algorithms	Alloy design (in conjunction with a material modeling approach), model optimization. topology optimization for AM	Search algorithms are intuitive for material properties that can be described geometrically, such as topology optimization for weight reduction. They are efficient at searching spaces with multiple local extrema, such as finding local maxima of quality in multidimensional design spaces. For a useful application of genetic algorithms to process characterization, see Ref [29].	These success of these algorithms are highly dependent upon selection and mutation criteria
Neural Networks & Computer Vision	Artificial neural networks, Convolutional neural networks (CNN), General adversarial networks (GAN)	Classification, regression, feature identification and extraction in images, simulation of atomic potentials, transfer learning, in situ process monitoring, feedback and control	Neural networks have successfully modeled processing and image data; the research and development surrounding NNs is among the most mature of any type of machine learning algorithm	Neural networks tend to require large training datasets, especially for image analysis applications; however, transfer learning approaches can adopt NNs to small datasets

2.1. The design space of additive manufacturing

The design space of metals AM is the set of all PSPP relationships. More specifically, the term "design space" will be used throughout this article in reference to the set of AM data that is used and calculated by machine learning algorithms. An example design space for laser powder bed fusion (LPBF) of metals, the most industrially prolific of current metals AM technologies, is graphically depicted in Fig. 1. A complementary example of a process design space of LPBF is given in Table 1. Observable process phenomena may link the manufacturing parameters to the resulting materials properties, hence they may also be used to augment the manufacturing parameters and material properties within the design space. Examples include melt pool morphology, temperature history, and cooling rates.

A single combination of process parameters, observed process phenomenon, measured material properties, and a part's performance can be considered as a coordinate, or point, in the design space. Single coordinates, defined this way, can sometimes lead to a multitude of material properties due to latent variables, unforeseen complications, and the stochasticity of the process. Explicit consideration of process phenomenon in the design space coordinate can be used to more accurately establish unique points within the design space. In summary, any part that is processed under a single set of conditions and is observed to have a set thermal history and set of material properties can be considered to be manufactured at that point in the design space.

While the design space of AM is vast, data cannot always be given to machine learning algorithms "as is." It is important to consider the sources of data in the design space and how they need to be changed or curated for use with ML.

2.2. Data sources

Data, as a materials scientist normally thinks about the term, encompasses a vast range of sources and formats. Some of the most common sources of data used by materials scientists for AM can be seen in Table 3.

The most obvious data that materials scientists interact with are scalar values like modulus, ultimate tensile strength, laser scan speed, laser energy, layer height, etc. Distributions of scalars are also used such as grain size distribution or particle size distribution of AM feedstock. Many materials scientists interact with series data that can be subdivided into several more categories. Times series data can include a temperature measurement from a thermocouple during an AM build. Other series data include X-ray diffraction histograms or X-ray fluorescence spectra.

Data can also take non-quantitative forms, often referred to as categorical data. These can include crystallographic structure, grain morphology, or the shape of an AM part. In many cases, these categorizations can be converted into quantitative data by measuring a feature such as the major and minor axis length of a grain. More difficult to quantify categorical data in AM includes melt pool morphology and track solidification defects like "balling" or "lack of fusion/delamination."

Images are some of the most commonly obtained data sources in materials science and are taken from a wide range of techniques. Light optical microscopy, scanning electron microscopy, and transmission electron microscopy images are all collected to study material structure. Materials processing images may include computed tomography radiographs and/or 3D reconstruction of a melt pool and thermal measurements using two-color pyrometry. Images can be treated as a data point on their own, but they are often analyzed to extract other data such as measuring grain size from light optical microscopy or categorizing crystal structure from a transmission diffraction pattern.

Data can also be esoteric, depending upon the problems within AM that are being addressed. For example, a vector field of particle flow from a computational fluid dynamics simulation can be considered data. The orientation distribution function of the material's texture can also be considered data. The 3D model and slicing path used to generate an AM part can be considered data. Limitations on what constitutes "data" in a materials science problem are not worth defining. Rather, it is more important to consider how data can be featurized for use with an ML algorithm, as this ability determines whether or not data are amenable to use for machine learning approaches.

2.3. The featurization and curation of AM data

Featurization involves extracting information from a data set such that a machine learning algorithm can interpret relationships between features themselves or between features and desired processing outcomes like mechanical strength, surface roughness, shape, etc. The preprocessing step of featurizing data is crucial for successful implementation of machine learning algorithms. Improper featurization of data can impact prediction and classification errors [30].

Scalar data are possibly the easiest to work with because they are features themselves; scalar values are also often referred to as descriptors in this context. Therefore, they do not necessarily require featurization but rather curation and organization for use with ML. In many cases, scientific and engineering studies of AM map scalar data related to machine parameters – like those in Table 1– to scalar measurements of material properties, like strength, modulus, surface roughness, density, etc.

It is important that machine learning models are trained on datasets with a certain volume of data collected i.e. datasets with statistical variability – in some cases this could be individual measurements of machine parameters and material properties distributed across the design space. In other cases, having repeated measurements at the same place in the design space can reveal variability in the manufacturing process. Scalar data can be featurized by conducting simple statistical tests to understand relationships that are already present in the dataset. Statistical information such as

- mean, median, and mode;
- standard deviation;
- the presence of outliers;
- correlation coefficients between parameters;
- type of distribution (Gaussian, Lorentzian, Weibull, etc.).

are fairly straightforward to assess. Understanding the basic statistical nature of the machine learning algorithms can prevent problems in the application of ML to AM. For example, heavily correlated inputs in a dataset impact the results of machine learning models. In the worst case scenario, having correlated inputs degrades the predictive capability of the algorithm being implemented; in the best case, it has no effect on model performance but slows down modeling and computation time by adding unnecessary computations [31].

The type of distribution that best describes the data may guide the underlying assumptions that some machine learning models make. For example, as further discussed in Section 3, Gaussian Process Regression assumes that a Gaussian distribution best describes the statistical variance of the data being modeled [32,33].

The removal of statistically correlated inputs (if necessary) combined with determining the statistical nature of the dataset is the curation of data.

Once it is determined that the data are properly featurized and curated, the next step is data organization. Collections of scalar values can be represented by several different mathematical tools before use with machine learning. Matrices are used in machine learning to represent multiple observations, typically stored in rows, of a set of features (columns), for example,

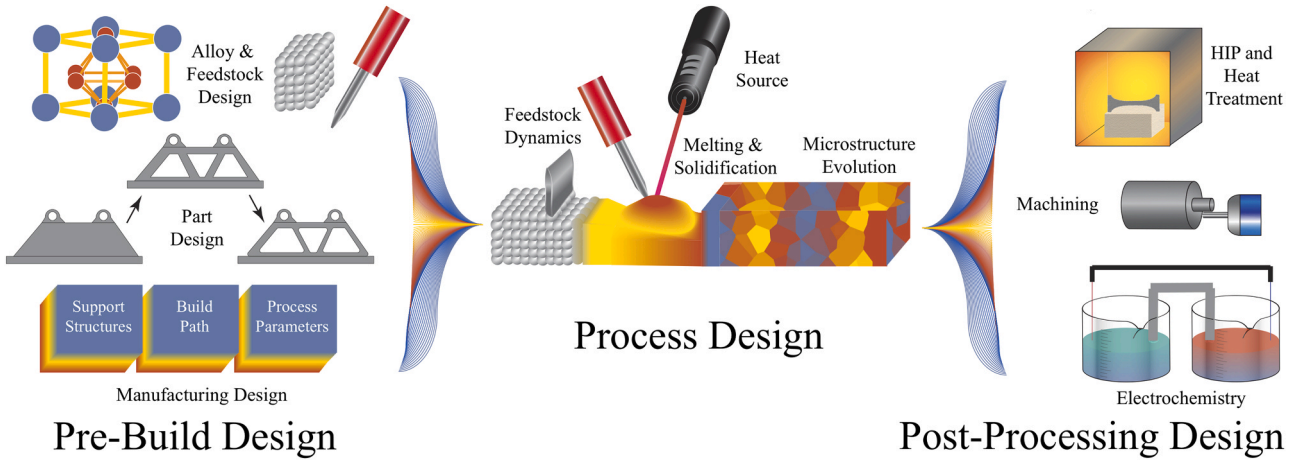


Fig. 1. The design space of metals additive manufacturing spans many engineering disciplines since the material and part are made at the same time. As shown in this schematic, alloys, parts, and manufacturing process designs are concurrently considered in the "pre-build" phase. The physics of the process itself may then be modeled, including feedstock dynamics, thermodynamics and kinetics of melting, solidification, and thermal histories, which dictate the final microstructure. Today, post-processing treatments are typically performed as secondary processes, though the future points to "hybrid manufacturing" processes where they are also incorporated at the point of fabrication.

Table 3
Types and sources of data common in materials science and, specifically, additive manufacturing. The entries under each vary from a source of data – like a characterization technique – to the data itself – like a single measured scalar value.

Scalar	Time series	Spectral	Images	Categorical	Spatial
Ultimate tensile strength	Stress-strain curve	X-ray diffraction	TEM	Composition	3D model and slicing path (e.g. STL file)
Hardness	Temperature gradient	X-ray Photospectroscopy	SEM	Quality	Scan path
Toughness	Pyrometry	X-ray dispersive spectroscopy	Optical metallography	Crystal structure	Part orientation in build chamber
Fracture Strength	Thermography		X-ray computed tomography	Melt pool morphology	Crystallographic texture
Density	Differential thermogravimetric analysis		High energy diffraction microscopy		
Solidification velocity	Differential scanning calorimetry				
Cooling rate	Chemorheology				
Solidus/Liquidus temperature	Magnetometry				
Enthalpy of formation/ melting					
Pore size					
Fatigue properties					

$$\mathbf{X} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,m} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,m} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,m} \end{bmatrix} \quad (1)$$

where the columns of **X** represent the different features, out of m, and there have been n repeated measurements of each. In some machine learning uses, the investigator wants to learn trends within the dataset. When varying dozens of parameters at a time, which is often the case in additive manufacturing, trends across multiple print parameters are not always obvious. In this case, a dataset of observations can be formatted into a matrix like in Eq. (1). This is referred to as unlabeled data. In other cases, the experimenter wants a predictive tool that allows them to ask: if I print at these specific conditions, what will be the result? In these cases, it is better to store the print parameters in a format like Eq. (1), but have the resulting properties stored in a separate vector object, like

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad (2)$$

In this case, the data has been separated into inputs and outputs. This is referred to as labeled data.

A time series signal can be represented as a list of scalar values that are correlated in the time dimension. Indeed, it is possible to represent collections of time series signals using the mathematical form in Eq. (1), where each column is a time step and each row is a different measurement. For some applications, this data processing approach will result in unnecessary data being used for modeling. For example, if looking for indications of defect formation, much of the collected data can be ignored. It can be reasonably assumed that defect formation occurs when certain signals change from an expected mean value, like a rapid rise in temperature or energy density of the laser. In these cases, it is better to search for indications of these changes away from the expected mean value instead of using the entire signal.

Featurization in this case is searching for aspects of the series that are correlated with a desired process outcome. For a timeseries signal, useful features include the signal maximum, minimum, locations with sharp changes in curvature, sudden changes in absolute value, and more. For other series data, such as diffraction histograms or spectral data, other features need to be considered. For diffraction histograms values like peak position, peak breadth, peak intensity, etc., are useful. Values

measured in between peaks (i.e. the background noise) can likely be ignored. One of the major benefits of featurizing series data is removing unnecessary values – indeed, the field of compressive sensing is focused around removing redundant information from series signals [34]. This type of featurization is useful for formatting extracted scalar values into matrix and vector objects like Eqs. (1) and (2). Once features have been extracted from the series signal they can be represented as a collection of scalars. From there, the collection of features should be treated to the same statistical litmus tests described above for scalar values. It is worth noting that some machine learning algorithms use entire series for inputs and featurize them as part of the algorithm [35–37].

Featurization of images is an active area of research in computer vision, a subfield of machine learning. Images are characterized by a spatial correlation in intensity: discrete changes in intensity dividing regions/domains of comparable, or slowly varying, intensity. Images are also most often represented as matrices of spatially-correlated intensity. The image processing algorithms discussed elsewhere in this review rely on a matrix representation of images. There are many toolboxes available, both free and commercial, which can pre-process images for use in machine learning algorithms; for example, the MATLAB Computer Vision Toolbox and the C++/Python OpenCV libraries.

Featurization of images occurs in a wide variety of ways and will be discussed in-depth in Section 3. However, it is worthwhile here to discuss filters, one of the most common ways of extracting features from an image. A filter is a mathematical operation applied to a region of an image that changes or enhances that image. Filters can be used to remove noise or distortion from an image, blur the image, sharpen edges, and more.

Filtering an image is computing the product of a matrix $\mathbf{w}$ with a matrix $\mathbf{f}(x, y)$. The function $\mathbf{f}$ is the pixel value of an image $\boldsymbol{I}$ at location (x, y). The filter is applied as the product

$$\mathbf{g}(x,y) = \sum_{s=-m}^{m} \sum_{t=-n}^{n} \mathbf{w}(s,t)\mathbf{f}(x+s, y+t) \tag{3}$$

where $\mathbf{g}(x, y)$ is the matrix resulting from the operation .[2] The product in Eq. (3) is a convolution of $\mathbf{w}$ and $\mathbf{f}$. In certain cases a correlation operation is applied instead. More information on these operations can be found in the work of Szeliski et al. [38], or any online open resource discussing image filtering.

The product of filtering is another image $\mathbf{g}(x, y)$ that has been modified or enhanced to reveal aspects of the original image. The application of filters can identify edges, reveal bright spots, reduce noise, blur, and do more to an image. The filtered matrix $\mathbf{g}$ can also be used as input for a machine learning application, like regression. Some machine learning applications like convolutional neural networks (CNNs) actually learn filters $\mathbf{w}$ themselves that maximize prediction accuracy in regression applications.

While filters are perhaps the most common featurization tool for images, other featurization methods exist. N-point correlation functions have found extensive use in extracting features from materials microstructure data [40].

Many machine learning algorithms can operate directly on machine inputs and processing outputs; however, it can be equally useful to measure the relationship between data points instead of the values of the data points themselves, i.e. using the covariance. The covariance is measured between two data points $\kappa(\mathbf{x},\mathbf{x}')$, instead of being a property of a single data point. The function $\kappa(\cdot,\cdot)$ is called a kernel function. The covariance between data points encodes cross-correlated information within the design space. Kernel functions can be used to assess the similarity of design space coordinates or to transform the feature space – for example, from a linear to a logarithmic space, or from a continuous to a logistic space – to better suit the underlying physics of the feature–target relationship [41]. Ways of calculating covariance are many and varied and will be defined explicitly throughout this review as they are used.

Once data has been pre-processed and featurized it can be used in a machine learning algorithm, but first it is important to consider some of the underlying assumptions of machine learning and ensure that the dataset being used meets those assumptions.

2.4. The assumptions behind machine learning

Two fundamental assumptions underpin the use of machine learning:

.

- *The Relational Hypothesis*: a correlative relationship exists between the data input to the ML model and the response of the system.
- *The Similarity Hypothesis*: similar points in the design space will have similar properties.

The relational hypothesis is a foundation for predictive models: after all, no prediction is possible in the absence of a correlative relationship between input and response.

The similarity hypothesis supposes that data are comparable: that according to some measure of similarity, similar input will produce similar output.

There are two types of machine learning covered in this review: unsupervised and supervised. Unsupervised learning will find trends in a dataset that are indicative of the underlying behavior. Supervised learning will learn a function $f(\mathbf{x}) = y$ that encodes part of the PSPP relationship. We proceed to walk through toy examples of each type; keep in mind that these are simplified examples meant to provide intuition behind the uses of machine learning. Scientists and engineers should research machine learning models, their uses, and their specific underlying assumptions before applying them.

2.5. Unsupervised machine learning

Unsupervised machine learning algorithms are used to identify similarities or draw conclusions from unlabeled data by relying on the similarity hypothesis. Unsupervised approaches are useful for visualizing or finding trends in high dimensional data sets, screening out irrelevant modeling inputs, or finding manufacturing conditions that produce similar material properties.

Consider an experiment that varies three different manufacturing inputs x_1, x_2, x_3 and measures a single material property y. A distance metric can be defined between data points in the design space. For example, data can be collected at two points $\mathbf{a} = (x_1, x_2, x_3)$ and $\mathbf{b} = (x_1 + \delta, x_2, x_3)$. The ℓ_2 norm of $\mathbf{a}-\mathbf{b}$ yields

$$||\mathbf{a}-\mathbf{b}||_2 = \delta. \tag{4}$$

The value and magnitude of δ gives an inclination about how similar $\mathbf{a}$ and $\mathbf{b}$ are. If δ is close to zero, then a researcher can say that $\mathbf{a}$ and $\mathbf{b}$ are similar. As δ becomes larger a researcher can say $\mathbf{a}$ and $\mathbf{b}$ become more dissimilar. The concept of "similar" manufacturing conditions may be easy to assess by an experimentalist when tuning only a few parameters at a time. When taking into consideration tens or hundreds of design criteria, sometimes with correlated inputs, elucidating similar manufacturing conditions becomes difficult. This vector distance approach is a simple, yet effective first glance at similarity in a design space and is generalizable to n many design criteria.

Steps must be taken to ensure that $\mathbf{a}$ and $\mathbf{b}$ can be compared in a meaningful way. It would not make sense, for example, for the first entry of $\mathbf{a}$ to be in units of N while that of $\mathbf{b}$ is in MPa. Dimensional analysis provides a physical check on comparability; that the same property is

[2] The product between $\mathbf{w}$ and $\mathbf{f}$ is not valid in all locations due to mismatches in indices at the border of the image. There are special cases defined where either the weight matrix or the image needs to be modified; Szeliski [38] and MATLAB's documentation [39] provide more information.

being measured in both instances. But distances need not be limited to physical quantities or measurements. Distance is, more generally, a measure of the similarity of two objects and. The similarity, or dissimilarity, of two distributions P and Q may be measured by their Kullback–Leibler divergence, $(D_{KL}(P||Q) + D_{KL}(Q||P))$, although not a true distance metric because the KL divergence does not satisfy the triangle inequality. The distance between two sets, say two sets of categorical variables, could be measured by the Jaccard Distance or by the Tanimoto Similarity.

Let us say that δ is small and that **a** and **b** are similar manufacturing conditions. Now, consider a third point in the design space $\mathbf{c} = (x_1 + \delta, x_2 + \delta, x_3)$ that has not yet been measured.

Since **c** was manufactured at similar conditions to **a**, as measured by $||\mathbf{c} - \mathbf{a}||_2 = \sqrt{2}\delta$, then we may say that **a**, **b**, and **c** are all similar to each other. If the similarity hypothesis is correct then manufacturing with conditions **a**, **b** and **c** should yield similar measurements of y.

To better understand why unsupervised learning is desirable for AM R&D consider a research project with initial manufacturing inputs **a**, **b**, **c**, **d**, etc., and associated property measurements that have been tested. Measuring the remainder of all possible design space coordinates to map the process-structure-property-performance relationship quickly becomes prohibitive. Instead, researchers can use similarity metrics to determine whether or not a future test is worth running. Comparing the manufacturing inputs through vector distance gives a rough idea of the possible outcome before spending time and resources on running a test. If the intent is exploring design spaces then manufacturing at conditions furthest away from previously observed points may be the answer. If looking for local maxima of quality, an operator would want to manufacture at conditions *nearest to* the conditions currently known to have high quality.

Another common application of unsupervised learning is finding clusters in data sets that produce useful partitions of material behavior. Using vector distances as metrics of similarities can produce results that are analogous to creating process maps [42], which is further discussed in Section 3.2.2.

The following demonstration of unsupervised learning is based on k-means clustering, a commonly used unsupervised machine learning clustering algorithm.

A researcher has acquired the datasets in Eq. (1) and wants to partition $x_j \in \mathbf{X}$ into groupings of print parameters that produce similar results. However, there are several values of $x_j \in \mathbf{X}$ that lie between two extremes and the cutoff for "similar conditions" is not obvious. Similarity metrics can be used to find demarcations in the dataset that indicate regions of similarity. To begin, the data set is partitioned randomly into two groups, $\mathbf{X}_1$ and $\mathbf{X}_2$. The centroids m_1, m_2 (or centers of mass, in engineering) of each grouping can be calculated as

$$\begin{aligned} m_1 &= \frac{1}{|\mathbf{X}_1|}\sum_{x_j \in \mathbf{X}_1} x_j \\ m_2 &= \frac{1}{|\mathbf{X}_2|}\sum_{x_j \in \mathbf{X}_2} x_j. \end{aligned} \tag{5}$$

where $|\mathbf{X}|$ is the mean value of a grouping. The measurements were randomly partitioned at first; the goal is to re-partition each set so that similar measurements are in the same set. To do this, we can re-assign each set by

$$\begin{aligned} \mathbf{X}_1 &= \{x_i : ||x_i - m_1||_2 \leq ||x_i - m_2||_2\} \\ \mathbf{X}_2 &= \{x_j : ||x_j - m_2||_2 \leq ||x_j - m_1||_2\}. \end{aligned} \tag{6}$$

The re-assignment in Eq. (6) can be interpreted physically: if a measurement initially assigned to set $\mathbf{X}_1$ is closer in distance to the centroid of $\mathbf{X}_2$ then it is more similar to the other set. Thus, it is re-assigned. Measuring the similarity of each data point to the mean of the groupings re-classifies these outliers into groupings that are more reflective of the position in the high dimensional design space, giving manufacturing designers insight into how design parameters are distributed in that space.

Once re-assignment is complete the centroids in Eq. (5) can be re-calculated and updated. Then, data points are re-assigned once more based on how similar they are to the centroid of each partition. If the input settings (x_1, x_2, x_3) are partitioned along with their corresponding measurements, then we have lists of input settings that are likely to give good/bad quality parts. Further analysis can also be conducted, such as analyzing which regimes of inputs lead to good or bad quality – this is precisely what process maps represent. The difference in this case is that n many manufacturing conditions can be related to a quality metric simultaneously, with little to no human inspection or intervention. Additionally, a researcher can dig further and analyze *why* groups of input settings result in given quality for a material property.

2.6. Supervised machine learning

In a supervised machine learning algorithm the goal is to determine a functional relationship $f(\mathbf{x}) = \mathbf{y}$ based on previous measurements of **y** at points **x** in the design space. That is, supervised machine learning algorithms relate manufacturing inputs to labeled output data.

Functional mappings of input data **x** to process outcomes **y** can take the form of either regression or classification. In a regression problem, the goal is to find mappings between inputs **x** to continuous values of **y**. An example includes predicting mechanical strength from processing conditions, where the process conditions can be continuous or discrete, like those in Table 1, and the output **y** can be any reasonable value of strength. A classification problem sorts inputs **x** into categories with associated labels. These classifications can be binary or one-of-many classes. An example would be training an algorithm to answer the question "Will the build fail?" based on processing inputs, with the possible class labels being "Yes" or "No."

Functional relationships can take many forms, depending on the specific supervised ML algorithm being used. One method is to model the relationships as a vector product

$$\mathbf{X}\beta = \mathbf{Y}. \tag{7}$$

where β is a vector of coefficients that weight the machine inputs to approximate an entry in **Y**.

A researcher usually seeks this relationship through the measurements they have observed; in this case, the measurements are stored in the matrices of Eqs. (1) and (2). A common method to find a vector representation of β, and a critical element in most machine learning algorithms, is through least squares regression. Least squares regression finds β through a minimization problem, given by

$$\min\left|\left|\mathbf{X}\beta - \mathbf{Y}\right|\right|_2^2. \tag{8}$$

Eq. (8) can be interpreted analogously to similarity measurements for unsupervised algorithms: the closer that $\mathbf{X}\beta - \mathbf{Y}$ is to zero, the more similar $\mathbf{X}\beta$ is to $f(\mathbf{x})$.

The methods of solving Eq. (8) are many and varied; indeed, much of this review will focus on finding solutions to Eq. (8) for various problems throughout additive manufacturing. The result is an approximation to the functional relationship $f(\mathbf{x}) = \mathbf{y}$. A new point of interest in the design space $\mathbf{x}'$ can be chosen and its associated material property $\mathbf{y}'$ can be predicted by computing

$$\mathbf{x}'\beta = \mathbf{y}'. \tag{9}$$

This simple example demonstrates how functional relationships can elucidate more information about design spaces from previously generated data.

2.7. Error metrics

Models that are used to predict values, whether numerical regression or classification algorithms, must have metrics to assess success. There are a multitude of error metrics that are used in the machine learning community. Different error metrics provide different information about the model, such as its ability to predict mean values, its robustness against outliers, and uncertainty in predictions, amongst other information. Many different error metrics have been formulated by the statistics community and used by the ML community [43]. Here, we review many of the most commonly used error metrics. For readers interested in more in depth discussion and examples, the website DataQuest provides an open source article about common error metrics [44], explanations of commonly used error metrics and their benefits/drawbacks can be found in Table V of Shan et al. [45], and Botchkarev wrote a review article detailing different error metrics used by the machine learning community over time [46].

The following parameter definitions are used in the ensuing basic introduction of common error metrics and the remainder of the article.

- $\hat{y}$ – the value predicted by a regression algorithm $f(\mathbf{x}) = \hat{y}$.
- y – the actual value of a material process/structure/property at input location $\mathbf{x}$.
- n – the sample size used to train a machine learning algorithm.

The mean absolute error (MAE) assesses the absolute residual between the predicted value of a regression problem and the actual value. It is calculated as the absolute difference between predicted value $\hat{y}$ and the actual value y, normalized by the sample size. Stated mathematically, the mean absolute error is

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{10}$$

MAE penalizes error linearly. The MAE penalizes outliers in the data with the same magnitude as data points lying close to the mean. The mean absolute error can also be changed into a percentage, the mean absolute percentage error (MAPE) by normalizing each individual error measurement against the actual value y. Stated mathematically,

$$\text{MAPE} = 100 \times \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{11}$$

Other error metrics highlight the impact of outliers on the dataset. The mean squared error (MSE) squares the difference term in Eq. (10) to produce

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|^2. \tag{12}$$

The MSE penalizes error quadratically. Outliers in the dataset will have a much larger impact on MSE than they will on the MAE. A downside of the MSE is that the errors are reported as the square of the units being predicted by the model. Some users wish to have an error with the same units as the value being predicted; thus the root mean squared error (RMSE) adds a square root such that

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right|^2}. \tag{13}$$

All of the above metrics produce a measure of the absolute value of the error in the model. In some cases it is useful to know if a model is *over* predicting the value (negative error) or under predicting the value (positive error). In these cases, the MAE can be modified to the mean percentage error (MPE), given as

$$\text{MPE} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{y_i}\right) \tag{14}$$

The MPE can reveal if a machine learning prediction algorithm is skewed towards certain types of values.

All of the above error metrics are suitable for regression problems with continuous value of $\hat{y}$. In the case of a classification problem, where the outputs are non-numerical, a non-numerical method of measuring error must be defined. While several methods have been developed [47,48] a common method is to use a confusion matrix. A confusion matrix displays the percentage of classifications that were correctly identified, as well as the percentage of classifications made to the wrong class. An example confusion matrix can be seen in Fig. 2. In this figure, the main diagonal of the figure displays the percentage of data points that were correctly identified by class. The off-diagonal components display when a certain class was mis-identified as another class and how often it occurred.

2.8. The bias-variance tradeoff and model validation

Now that basic methods of machine learning and associated error metrics have been defined we proceed to introduce how machine learning models are fit and validated. The following discussion focuses on finding parameters to fit a machine learning algorithm, how those parameters are validated, and common obstacles that arise in validating the model.

A cost function[3] $C(\mathbf{x}; \boldsymbol{\theta})$, is a metric that quantifies the fit of a particular model parameterization. A cost function quantifies the error in a prediction and can be any non-negative, scalar function that increases monotonically away from a target value. [50]. That is, for every input dataset $\mathbf{x}$ there is an associated set of parameters $\boldsymbol{\theta}$ for the machine learning model that best fit $\mathbf{x}$ to corresponding $\mathbf{y}$. The training step is

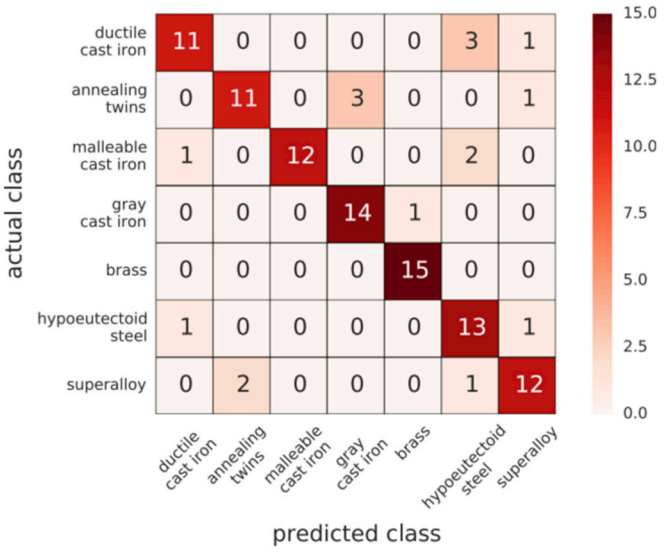


Fig. 2. A confusion matrix used in a study by DeCost and Holm [49]. The goal of the study was to classify materials based on images of their microstructures. The main diagonal of the matrix represents correct classifications. In the case of the upper-leftmost entry, 11 images of ductile cast iron were correctly identified as ductile cast iron. The upper-rightmost entry indicates that 1 image of ductile cast iron was incorrectly classified as a superalloy.

[3] Also sometimes called the loss function or reward function depending on if the objective is to minimize or maximize the value [50].

concerned with finding the model parameterization that minimizes the cost. There are many different choices for cost function but the best known cost function is the squared loss, given in Eq. (8).

The cost function is used to optimize a parameterization of a machine learning algorithm. For example, a least squares regression algorithm is parameterized by β_i,

$$\hat{y} = \beta_0 + \beta_1 x + \beta_2 x^2 + \ldots + \beta_n x^n. \quad (15)$$

Optimal values of β_i minimize the value of $|\mathbf{X}\beta - \mathbf{Y}|$ in Eq. (8). The actual method of performing this optimization can take on many forms and is discussed in-depth elsewhere. The scikit-learn package, part of the Python scipy library, provides many methods for optimization of cost functions [51]. Gradient descent is a common method for performing cost function optimization [52]. Brochu et al. discuss optimization of cost functions using Bayesian optimization, an important topic in modern statistics [53].

Most machine learning methods – such as neural networks, decision trees, and ridge regression – also have model hyperparameters. These parameters define aspects of the model itself, not aspects of a specific parameterization of the relationships between **x** and **y**. For linear regression of a polynomial function to a dataset the weights β_i are model parameters and the order of the polynomial is a model hyperparameter. Hyperparameters will be discussed more in-depth later as specific machine learning algorithms are introduced in Section 3.

All machine learning models follow a basic training and validation process:

- Divide data into training, test, and validation data: $\{\mathbf{X}, \mathbf{y}\} \rightarrow \{\{\mathbf{X},\mathbf{y}\}_{\text{train}}, \{\mathbf{X},\mathbf{y}\}_{\text{test}}, \{\mathbf{X},\mathbf{y}\}_{\text{validate}}\}$.
- Estimate the model parameters, $\hat{\boldsymbol{\theta}}$, using $\{\mathbf{X}, \mathbf{y}\}_{\text{train}}$ using an appropriate cost function.
- Adjust the model hyperparameters using $\{\mathbf{X}, \mathbf{y}\}_{\text{test}}$ based on the accuracy of the best fit parameterization of $\boldsymbol{\theta}$.
- Validate the best parameterization and check against over- or underfitting by evaluating the model on the validation set $\{\mathbf{X}, \mathbf{y}\}_{\text{validate}}$.

These steps are repeated until the model performance, as measured by the model error estimate, converges.

As the complexity of the model increases – such as the complexity of a polynomial in a linear regression problem – so does the tendency of that model to overfit to the training data and generalize poorly to unseen inputs, leading to an increase in the out-of-sample error. This balance between the ability of the model to represent the inherent complexity between the input and output spaces (i.e., reduce the model bias) while minimizing the out-of-sample error (i.e., reduce the model variance) is the basis for the bias-variance tradeoff that is central to all machine learning models. Visual examples of overfitting, underfitting, and proper fitting can be seen in Fig. 3. The goal in validating a machine learning model is to find a balance between overfitting the training dataset and underfitting the testing dataset, as shown in Fig. 4. While the RMSE shows a decrease in the training dataset as model complexity increases, the RMSE of the testing dataset increases significantly.

Overfitting and selection bias can be sussed out through use of cross-validation. Cross-validation is the process of training machine learning models on subsets of the training set and evaluating with the remaining data to see how sensitive the model performance is to the choice of

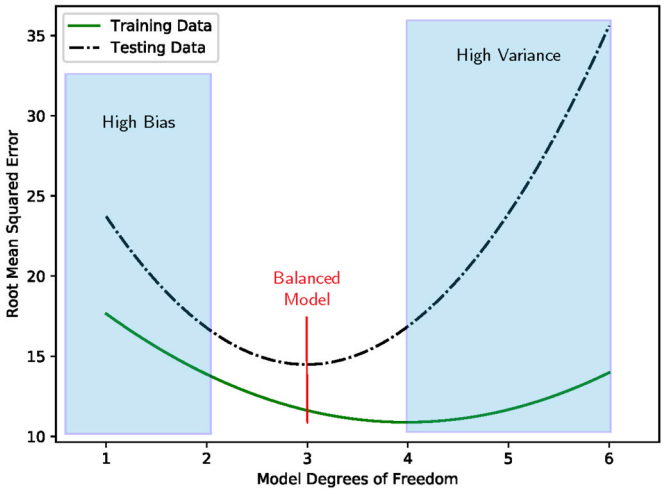


Fig. 4. The calculated root mean squared error for six different models fit to the dataset shown in Fig. 3. The training data continues to decrease as the models become more complex, demonstrating overfitting. However, when the model is evaluated against a test dataset the RMSE increases significantly with more complex models.

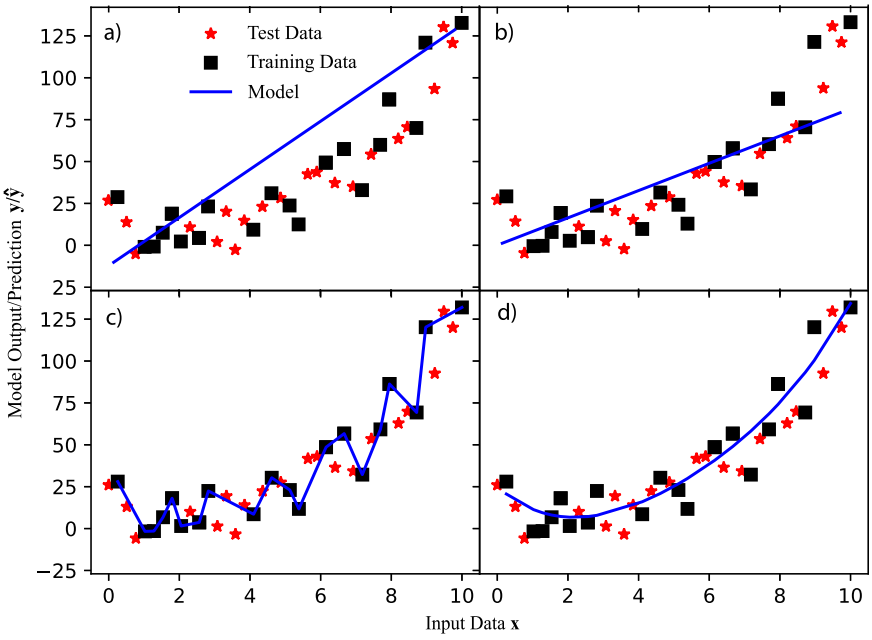


Fig. 3. Illustrations of high bias and high variance models. A toy dataset was generated from the polynomial $y = 5 + 0.1x + 0.1x^2 + 0.1x^3 + 0.002x^4 +$ Random Noise. The fits in (a) and (b) are both parameterizations of a model. Each model (line) in both fits has approximately the same error but does not accurately capture the behavior of the data due to poor model assumptions; in this case, fitting a first order polynomial to a dataset generated from a fourth order polynomial. This is an example of high bias models. A twentieth order polynomial was fit to a subset of the full dataset in (c), shown in blue. While the model has very good predictive error for the training dataset it will not extrapolate well to the data in the testing set; this is overfitting or high variance. A third order polynomial was fit to the data in (d) demonstrating a good balance between bias and variance. The model accurately captures trends in the data while not overfitting the training dataset.

different inputs. Cross-validation is often referred to as k-fold cross validation because the machine learning model is trained on k different subsets.

In k-fold cross validation the training data set is randomly split into k different groups or folds. The machine learning model is trained on $k-1$ of the folds and tested on the kth fold. For each training and testing set the fit model parameters and associated error should be kept in order to assess how each parametrization of the model performs.

Randomizing, training, and validating on multiple subsets of the data elucidates the model's ability to perform on new datasets. If a model is suffering from overfitting, or high variance, then it will have very low predictive error on the training set but perform poorly on the testing set. If the model is suffering from high bias then it may demonstrate similar performance metrics between fitting of each kth set but has high prediction error in general. High bias often results from improper assumptions in the machine learning algorithm or a poor choice of model hyperparameters. Cross-validation reveals these behaviors in machine learning models by providing error metrics for models trained on many different subsets. Necessary changes to the model hyperparameters, or even changes in machine learning modeling used, can be discovered from cross-validation.

One specific case of cross-validation where $k = n$ is called leave one out cross-validation. In this method the models are trained on all data points except one, then tested on the remaining data point. Leave one out cross-validation is especially useful for assessing the impact on outliers of the model performance.

Another method of cross-validation called leave-one-cluster-out (LOCO) cross-validation was introduced by Meredig et al. [54] for materials science applications. LOCO CV was introduced to highlight problems in the distribution of data in materials datasets. Often, datasets from materials science are limited around specific clusters of material compositions or properties. An example for AM is that most datasets generated focus around weldable alloys like 300 series steels, superalloys, and titanium alloys. As a result the prediction performance of machine learning algorithms may be biased toward these clusters of materials. LOCO CV uses a nearest-neighbor clustering approach – akin to the example given in Section 2.5 – to evaluate the impact of clustering of material types on prediction performance.

The above methods are for the validation of individual machine learning models. In many cases it is worthwhile to train several different machine learning models on the same problem and assess the best model. As is shown in Table 2, several different machine learning algorithms can often be applied to the same task. Because each algorithm has different assumptions, one type of ML model may perform better on a dataset than others. Thus, it is worthwhile to use tools that can compare the performance of different ML models for the same application.

2.9. Comparison across machine learning approaches

The validation of a single machine learning model can be addressed by the methods presented in Sections 2.7 and 2.8. Finding the best possible parameterization of an individual model does not guarantee that a researcher has found the best possible solution to their specific problem. It is generally good practice to evaluate several machine learning approaches to a problem and choose the best approach across all algorithms that may be reasonably expected to perform. Table 2 shows that many different algorithms can be used for the same types of problems. Different algorithms may have vastly different performance even for the same problem or dataset.

For example, Principal Component Analysis (PCA) and kernel ridge regression (KRR) can both be used as regression tools; PCA relies on the assumption of linearity between inputs and outputs while KRR does not. Often, a researcher might not know the if the relationship being studied is linear or not and therefore should try both options to see which produces a better result.

In general, researchers can follow a few steps to determine which model is best for their additive manufacturing problem:

- Evaluate if there are statistical correlations in the data of interest.
- Pre-process and featurize data for use with a machine learning algorithm.
- Tune the model parameterization and hyperparameterization through error analysis and cross-validation.
- Compare error metrics across several algorithms and select one algorithm as the best performer.

Regression models can be validated against each other using the error metrics in Section 2.7. It is important to use multiple error metrics for comparison because different machine learning algorithms handle outliers and statistical correlations differently. For classification problems, a graph called a receiver operating characteristic (ROC) curve has been developed to compare the classification success of different algorithms. An example ROC curve can be seen in Fig. 5. The ROC curve compares the true positive and false positive classification rates for a binary classifier, a group of problems whose solution can take one of two outcomes. To ensure that Type I error (false positive) accurately reflects the performance of the model, the less common outcome should always be taken as the True condition, and the more common outcome as the False condition.[4] More information on ROC curves can be found at Google's developers page [55].

Tools to compare across machine learning algorithms are invaluable and should be considered as a mandatory part of any machine learning approach. It is often the case that evaluating many machine learning

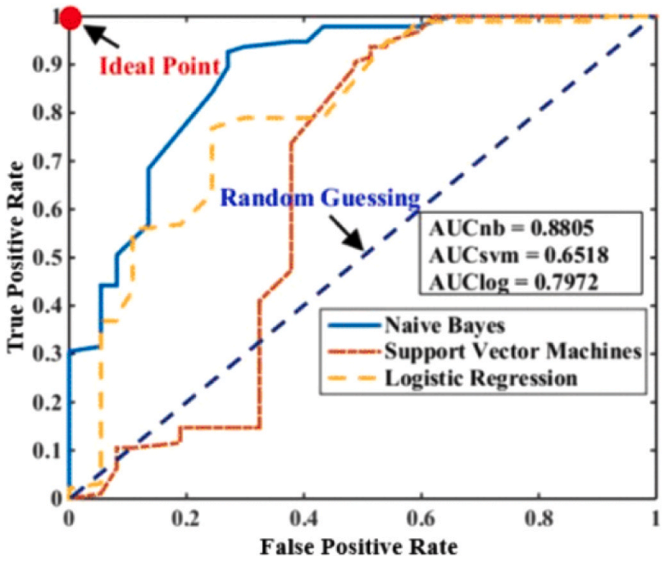


Fig. 5. An example receiver operating characteristic curve from the work of Liu et al. [56]. The goal of the study was to class material properties based on additive manufacturing machine inputs. The classes were regimes of material quality like "high density" or "low density." The dataset was built by mining data from literature on additively manufactured metals. The area under the curve (AUC) shows the integrated area under each algorithm's ROC curve; a perfect classifier has AUC = 1. In the example shown, Naïve Bayes significantly outperforms the other two algorithms and thus is the best choice of machine learning approach for this problem.

[4] Although restricted to binomial classification, the ROC curve may be extended to multinomial classification by recursion. That is, A or not A; and if not A, then B or not B; and if not B, then C or not C; etc. where A, B, C, etc. are all potential outcomes in order of increasing frequency.

algorithms against each other will lead to better overall performance because the best approach can be chosen from many. The ML packages listed in the next section all contain tools for comparing machine learning algorithm performance.

2.10. Machine learning toolboxes

Most of the machine learning algorithms and approaches discussed in this review are, in some form, free and openly accessible. Many machine learning packages exist across many different programming languages and platforms. Table 4 highlights a variety of computational tools and packages and their relevance to AM synthesis optimization.

3. Current ICME tools are well equipped to integrate with an ML framework

The following section discusses how machine learning approaches can be used in current R&D efforts in AM. This discussion includes how physics-based analyses, characterizations, and simulation methods may connect with different machine learning algorithms. Overall, the discussion is aimed at conveying how ML can be used to automate the generation of AM PSPP knowledge. Still, this article stops short of providing an exhaustive review of either machine learning algorithms or additive manufacturing. Instead, the intent is to introduce how ML approaches can be connected to AM research. The algorithms that are discussed were chosen because they were previously demonstrated in a materials science and engineering application or because the possible application of an algorithm to AM was clear and immediate. Similarly, the additive manufacturing problems addressed are not all-encompassing; they are merely a few that may be immediately addressable with machine learning approaches.

Table 4
Commonly-used machine learning, statistical analysis, and computer vision toolboxes. Some toolboxes listed are open source, while some are packaged with commercial software like MATLAB.

Language/ Platform	Package	Applications
Python	scikit-learn [57]	General data mining toolbox; packages for classification, regression, clustering, dimensionality reduction, model selection, and data pre-processing
	tensorflow [58]	Machine learning toolkit for data mining and data flows; specifically focuses on the use of neural networks and deep learning for model building and problem solving
	keras [59]	Deep learning-specific machine learning toolbox; designed for intuitive building of neural network systems
	OpenCV [60]	Algorithm toolbox for machine learning and computer vision; contains wide range of tools for image processing including image pre-processing, template matching, object identification, and convolutional neural networks
MATLAB	Statistics and Machine Learning Toolbox [61]	Commercial data analysis and machine learning toolbox with a wide range of applications in data analysis including clustering, classification, regression, and dimensionality reduction
	Computer Vision Toolbox [62]	Algorithm toolbox for machine learning and computer vision; contains tools for a wide range of image analysis including pre-processing, object identification, template matching, and convolutional neural networks
C ++	OpenCV [60]	Algorithm toolbox for machine learning and computer vision; contains wide range of tools for image processing including image pre-processing, object identification, template matching, and convolutional neural networks
	tensorflow [58]	Machine learning toolkit for data mining and data flows; specifically focuses on the use of neural networks and deep learning for model building and problem solving
R	Machine Learning in R (MLR) [63]	Infrastructure for incorporating common machine learning functions in R in an easy way; provides robust packages for a wide range of machine learning-based tools including regression, classification, clustering, sampling methods, model optimization and more; has built in parallelization methods

3.1. Experimental methods and manufacturing design

3.1.1. Alloy design and feedstock selection

Choice of alloy impacts the physics of AM from start to finish, ranging from the interactions of energy sources with material feedstocks to the performances of the final parts. For example: the reflected vs. absorbed intensity of lasers on powder beds is determined by the powder's composition [64,65]; the density of feedstock, both intra- and inter-granular density, plays a role in final part density [66]; conduction modes in the melt are partially determined by the thermal properties of the alloy [8]; and different alloys exhibit different solidification kinetics, which can lead to drastically different microstructures after manufacture [67]. Problems in the additive process can also be linked to composition such as vaporization of constituent elements due to rapid thermal fluxes, impacting the stoichiometry of melt pools and, ultimately, quality [68]. These can be different for different feedstock types (e.g., wire vs. powder), even for the same alloy choice. Wysocki et al. discuss the differences between different additive manufacturing processes for titanium alloys: electron beam, laser based, powder, wire, etc [69]. Some studies have also investigated the impact of feedstock properties like particle size distribution and morphology on process quality [65,70,71], although the direct impacts have not been fully resolved.

As such, alloys developed for traditional metals manufacturing techniques such as casting, rolling, extrusion, etc. sometimes need to be altered to improve AM processing. In the best cases, alloys developed for AM may outperform traditionally manufactured alloys. For example, unique strengthening mechanisms can result from AM processing [8,7,68,72]. Designing alloys for AM – either altering the chemistries of known alloys or discovering new alloys – requires considering the implications of the physical properties of alloys with AM processing. An understanding of what trend in a physical property is "better" or "worse" for AM processing is still an open area of research. Hence, while information about the physical properties of different alloys has been collated into databases that are compatible with design for AM, models and optimization targets for mining those databases to extract candidate alloys for AM are still being developed and verified.

Existing databases contain alloy properties ranging from the reflectivity to the mechanical properties. The International Crystal Structure Database (ICSD) contains the crystal structures of millions of compositions. The Linus Pauling files contain a range of material information, from atomic properties like radius and electron valency to crystallographic level information [73]. More modern databases such as AFLOWLib [74] and the Materials Project [75] allow users to interactively search across different types of alloy information. Searching through large databases of information to find optimal compositions for manufacturing is actually one of the earliest materials informatics problems ever addressed. Methods exist to perform these searches in a fast, automated way. These methods are referred to as data mining, a data-driven materials design approach.

Data mining has been demonstrated to be useful for AM alloy development. Martin et al. used such an approach to modify the

chemistry of aluminum alloys to make them process better during LPBF [8]. The first step in a data-driven design process is to identify which alloy properties are important to the desired application. Laser powder bed fusion of Al alloys had been plagued by sparse nucleation of grains. The result was that large grains formed during AM together with large intergranular stresses, the combination of which resulted in hot-cracking. To overcome this problem, Martin searched for candidate grain inoculant compounds that could form through chemical reactions during LPBF. Searching for grain-refining nanoparticles has improved solidification properties [76]. For example, silicon and carbon could react to form SiC particles that would force more homogeneously, densely packed grain nucleation throughout the material. However, if such compounds had lattices that were dissimilar to those of the aluminum alloy, large stresses could form at the interface of the inoculants and the alloy matrix, still leading to cracking. Hence, they searched not only for potential inoculants, but more specifically for inoculants with crystallographic lattice parameters that closely matched those of the base aluminum alloy. Martin's study employed a search algorithm to search through 4500 different possible nucleants and identify those with the closest-matching parameters. Ultimately, hydrogen-stabilized Zr was found to be the best candidate.

The same database mining process employed by Martin – identify the target properties, then search for the closest match – can be extended to many AM problems as well. Database mining was first introduced in materials science to predict stable compositions, or estimate material properties from composition. Database mining has been successfully implemented to predict stable crystal structures [77–79] and predict material properties as a function of composition [80–84]. Some specially designed search algorithms have also been designed for improved speed in automated searches [85]. Successes have been found in designing Heusler compounds using high throughput search methods [86]. Several reviews exist detailing early high-throughput searches for compositions with ideal properties [87,88]. The same search algorithms employed in these studies can be extended to AM cases.

A limitation of database mining is that searches are limited to previously measured and/or calculated properties. Generally, information about the vast space of all possible materials is unknown. Traditional materials science and engineering approaches would turn to explicitly calculating or measuring the unknown points of interest, one at a time. Searching through compositions may be accessible for manufacturing processes like thin-film deposition where the composition can be adjusted continuously and with several species at once using well established methods. A combinatorial study of compositional changes for AM feedstock is hindered by the difficulty and expense of producing feedstock.

For example, consider the cost of combinatorially alloying Ti with alloying elements {Al, V, Zr, Cr, Hf} and then testing printability. Explicitly creating all possible combinations of {Ti, Al, V, Zr, Cr, Hf} is feasible if using a coarse set of level choices for additions of alloying elements, but undesirable. There are 15,503 alloy combinations if alloying in steps of 1 wt% up to 15% total alloying elements from the choices above.

However, using machine learning methods, the process of combinatorial exploration to find an optimal composition can be achieved without explicitly modeling each combination. For example, genetic algorithms (GA) can be used to augment many physics-based models. Genetic algorithms have been one of the most-used data driven approaches in materials science over the past few decades [79,85,89–93]. The principle of genetic algorithms is to evaluate the fitness of a population of candidate alloys against a fitness function. The fitness function $f(\cdot)$ is a method of evaluating how well a candidate alloy meets a criteria. Often in materials science the fitness function is evaluated by running models that can measure a material property based on composition. Examples include identifying stable crystal structure of a composition using DFT [77,79] and evaluating thermomechanical properties of an alloy using ThermoCalc [94]. Some additive-specific models include the model of Tan, which predicts dendrite arm spacing from composition [95]. The calculation of thermodynamic properties relevant to AM – such as vaporization temperature, coefficient of thermal expansion, solidus and liquidus temperatures – using the CALculation of PHAse Diagrams (CALPHAD) method [96] can also be a fitness function. For the sake of alloy design a model must be able to predict a material's properties based on composition. In reality, however, models must also consider additional physics related to the composition, such as crystal structure, thermodynamic properties, interatomic potentials, and more.

In using a GA for alloy design, a desired target property value must be identified. This value P_{target} is then formulated as a function of composition and process variables. Additionally, a method of measuring the property value as a function of composition and process variables $\mathbf{X}$ is needed; the models proposed previously (ThermoCalc, DFT, etc.) can serve as the evaluation step $f(\mathbf{X})$. The goal is to find a material whose measured property closest matches the desired target property, or

$$\min\left|\left|f(\mathbf{X}) - P_{\text{target}}\right|\right|. \tag{16}$$

As a thought experiment, consider various amounts of {Al, V, Zr, Cr, Hf} alloyed into Ti. These are the *genes* of the genetic algorithm. This is similar to a study completed by Li et al. [97]. Once a fitness function has been identified, the next step in a genetic algorithm is to represent candidate alloys as a chromosome.

We can represent a chromosome as

$$\mathbf{X} = [\chi_1, \chi_2, \ldots, \chi_n]$$

where χ_1 is the species and weight percent of the first element (titanium, in this example), χ_2 is the species and weight percent of the second element, up to n elements. For example, Ti-6Al-4V would be represented as

$$[0.9\ \text{Ti}, 0.06\ \text{Al}, 0.04\ \text{V}]$$

The goal is to find the alloy with optimal dendrite arm spacing. First, a population of candidate chromosomes needs to be generated, either randomly or by design. Two examples from a starting population may be

$$\text{Alloy}\ \ 1 = [0.9\text{Ti}, 0.05\text{Al}, 0.05\text{V}]$$

$$\text{Alloy}\ \ 2 = [0.9\ \text{Ti}, 0.1\ \text{Zr}]$$

The chromosomes produced from this initial population will serve as inputs to the fitness function.

Genetic algorithms select chromosomes out of the current population – called the parent generation – to proceed to another generation of model assessment – called the child generation. Selection consists of keeping the best performing compositions, say the top 10%, and discarding the rest, as determined by Eq. (16). Genetic algorithms find optimal locations in the design space by relying on the similarity hypothesis. If one alloy is in the top 10% of chromosomes then it is possible that a similar alloy will also be high performing – it may even perform better. Once selection is done, the next step is to search the space near the best performing alloys from the parent generation.

Genetic algorithms generate similar compositions from those selected in the parent generation by making alterations to genes. One operation is mutation, whereby genes are changed. For example, we could mutate alloy 1 by changing the composition:

$$\begin{array}{lll} \textbf{Parent Generation :} & \text{Alloy} & 1 = [0.9\ \text{Ti}, 0.05\ \text{Al}, 0.05\ \text{V}] \\ \textbf{Child Generation :} & \text{Alloy} & 1 = [0.9\ \text{Ti}, 0.02\ \text{Al}, 0.08\ \text{V}] \end{array}$$

where in the child generation the amount of V was increased, while the amount of Al was decreased. Another operation that may be performed is crossover where genes are added or interchanged. For example, one crossover operation may look like

Parent Generation : Alloy 1 = [0.9 Ti, 0.05 Al, 0.05 V]
Alloy 2 = [0.9 Ti, 0.1 Zr]
Child Generation : Alloy 1 = [0.9 Ti, 0.05 Al, 0.05 Zr]
Alloy 2 = [0.9 Ti, 0.1 V]

where in the second generation V and Zr have been interchanged.

Selection, mutation, and crossover followed by model assessment and further selection, mutation, and crossover continues until the design criteria is met. A schematic of the GA process can be seen in Fig. 6.

The similarity hypothesis supposes that similar points in the feature space have similar response values. That is, the similarity hypothesis is a tacit assumption of continuity. Therefore, the similarity hypothesis appears to be violated, and indeed, may be violated during perfunctory transformations–for example, first order phase transitions, such as melting, solidification, or allotropic phase transformations. But while first order phase transformations are discontinuous in the first derivative of energy with another thermodynamic variable (in the aforementioned cases, temperature), the energy continues to change continuously as the transformation proceeds. Therefore, if the target variable in an ML model is the thermodynamically stable phase, but the feature space includes temperature and excludes thermodynamic free energy, then a discontinuity will exist at the transformation temperature, similarity is violated, and the model will be unreliable near the transformation temperature. This failure of the similarity hypothesis is even more obvious in the presence of thermal hysteresis, where the model will become unreliable over a range of temperatures. However, both cases reveal a shortcoming in the formulation of the model, not in the model itself. In the former, the thermodynamic free energy, not the temperature, is the relevant variable, but the free energy is not tracked. The latter is further confounded by nucleation thermokinetics: surface energy and competing rates of nucleation formation and dissolution.

Behavior like phase transformations bring good opportunities for researchers to introduce physics-informed features in their problems. If the ML algorithm is not informed of sigmoid-like behavior, or Dirac delta function, or other nearly singular points in the design space then tens of thousands of data points and very high dimensional models would be needed to ever figure out the nonlinearity itself. However, since many of these nonlinearities are known from established physics, researchers can inform the ML by applying nonlinear mathematical transforms that linearize the data.

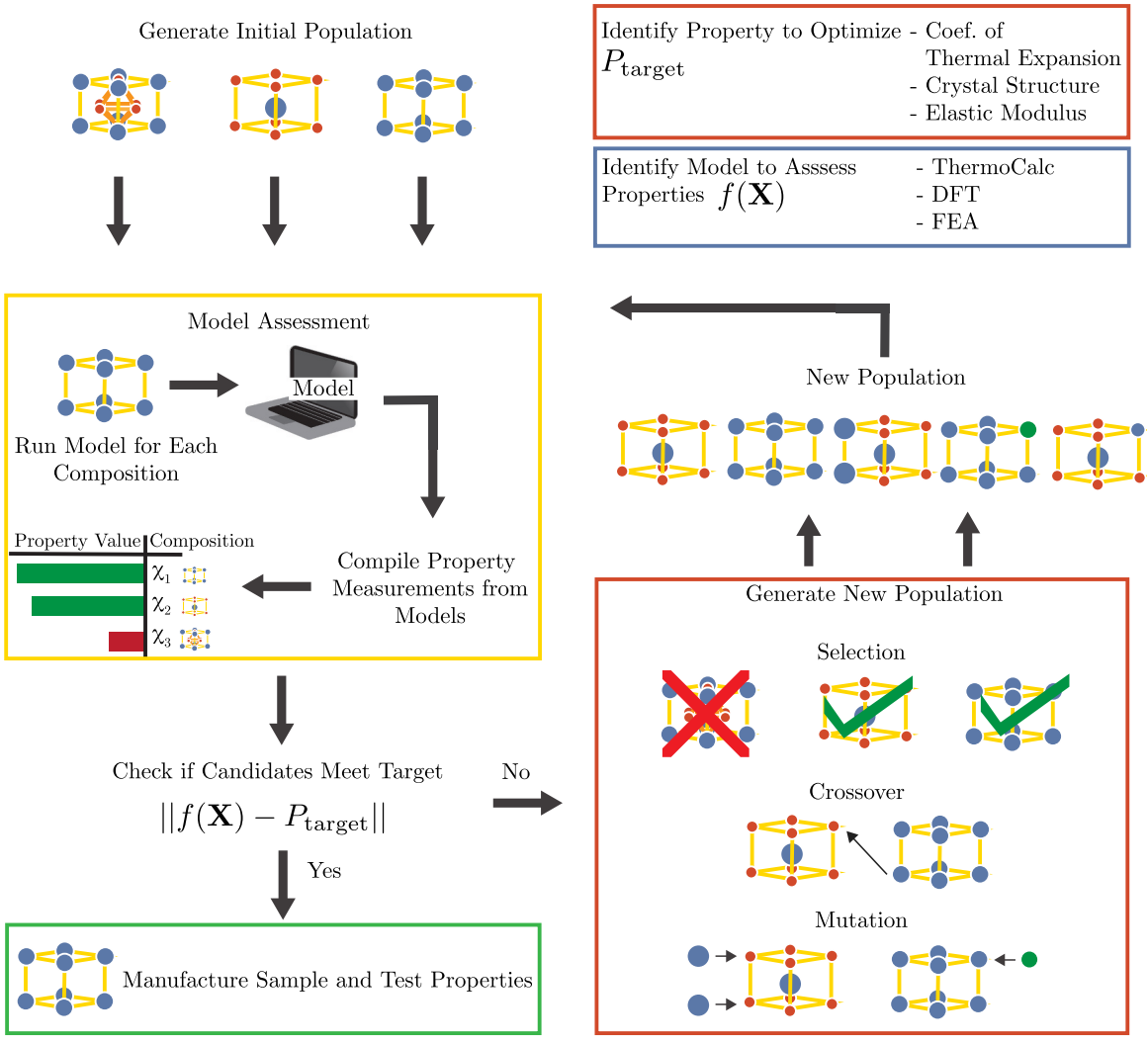


Fig. 6. An illustration of the alloy design process using a genetic algorithm. First, a target property P_{target} and an evaluation method $f(\mathbf{X})$ for the alloy $\mathbf{X}$ are chosen. The evaluation method is most often a material modeling approach that can predict material properties based on composition. Then, a population of starting compositions are made. The model is run for each composition and an associated material property is measured. The predicted values are compared against the target value. If no material matches the target, then the genetic algorithm begins. The closest-matching compositions are selected to create a child generation. Crossover and mutation occurs for those compositions that were selected. In this way, a new population of compositions are created that are similar to the best-performing compositions from the previous generation. Model assessment and the genetic algorithm are then run again until a composition is found that meets the target property value.

A study by Liu et al. explored exactly this problem [98]. In constructing Gaussian Process Regression (GPR) ML models for the design of alloys and their processing, it was found that first informing the ML models of known nonlinear, physical mathematical relationships, such as sigmoid and logarithmic functions that describe the time and temperature effects upon phase transformations that occur during precipitation and/or going to solid solution. The GPR algorithm used to predict properties of the alloys - in their work, shape memory effect properties of shape memory alloys - had difficulty accurately modeling these boundaries when times (t) and temperatures (T) were featurized as they would be in the laboratory notebook. However, using physics-informed nonlinear transformations of the process features t and T, such as $\ln(t)$ and $T \times (1 - e^{-(T-\theta)})^{-1}$ where θ is the critical temperature for a phase transformation allowed the GPR model to work well on relatively few (several hundred) datasets. With such physics-informed approaches to ML, simpler ML algorithms can work lesser amounts of data because the ML algorithm is informed of the known nonlinearities in the physics of the design space a priori, in effect linearizing the relationships that the ML needs to model. The similarity hypothesis also needs to be considered when selecting the features and targets of the model.

Genetic algorithms have been applied to alloy design for low and high temperature structural materials [80,99], ultra high strength steels [94], specific electronic band gaps [100], minimum defect structures [101], exploring stable ternary or higher alloys [91,102], and more. Chakraborti et al. wrote a review on the application of GA's to alloy design through the early 2000s [103].

In addition to genetic algorithms, other machine learning algorithms have also been applied to classify and optimize alloy compositions. Anijdan used a combined genetic algorithm–neural network method to find Al–Si compositions of minimum porosity [101]. Liu et al. applied partial least squares to data mining of structure-property relationships across compositions [104]. Decision trees, which are discussed in the next section, have been implemented for a number of different alloy optimizations, such as predicting ferromagnetism [105] and the stability of Heusler compounds [106]. In the search for new alloys, a wide range of machine learning algorithms can be implemented to guide the entire experimental design process so that an optimized property is found as quickly as possible. In the next section, we focus on using ML in design of experiments.

3.1.2. Design of experiments

Design of Experiments (DOX) is the design of task(s) aimed at performing parametric analysis [107]. Parametric analysis, broadly defined, is a method of mapping independent variables to corresponding dependent parameters. In materials science and engineering, process-property relationships are typically assessed using parametric analysis. Machine learning can reduce the number of experiments (i.e., tasks) needed to perform parametric analyses sufficient to characterize process-property relationships. Approaches such as sequential learning model relationships in parametric studies to discover regions of the parameter space that produce the most information about process-property relationships.

In additive manufacturing research, process parameters such as laser energy, speed, build direction, composition, and layer height are varied to study their impact on material properties. Examples include relating build geometry to microstructure or surface roughness [108,109], temperature history to microstructure [110,111], substrate temperature to residual stress development [68,112], or even entire manufacturing processes to microstructure [113]. Other commonly performed parametric analyses in AM relate heat source parameters to part temperature history [114,115], microstructure [116,117], mechanical properties [118,119], and residual stresses [120,121].

Information in AM research is any observation of process-structure-property relationships. For example, observing that a set of laser parameters results in an equiaxed microstructure can be considered information because the researcher has gained an idea of the structure to expect from set processing conditions. Therefore, information gain is any experiment that reveals a previously unobserved process-structure-property relationship. Rigorous mathematical definitions of information and information gain have been defined, typically referencing back to Shannon's original formulation of information theory [122].

Both engineering and scientific investigations of AM utilize parametric analysis. In science, tasks designed for information gain are performed until parametric analysis results in a theory or model for a process-structure-property relationship. In engineering, tasks designed for information gain are performed until an optimality criterion is met, such as maximum strength or minimum porosity. Both disciplines vary independent parameters and measure dependent responses until enough information about the underlying phenomenon is known to complete the parametric analysis with some predetermined level of certainty, variance, and/or precision.

Traditional DOX approaches maximize information gain from performing tasks by subdividing the design space a priori to maximize the likelihood of information gain from task to task. In these approaches, all pre-determined tasks are performed before parametric analysis is attempted. In machine learning DOX approaches, parametric analysis is performed after each individual task, and the next task to perform is determined based upon a statistical metric of the parametric analysis - as such, the likelihood of information gain incrementally improves as each task is carried out, and usually only a fraction (20–60%) of the number of tasks need to be performed to reach the established success criterion for the parametric analysis relative to the traditional DOX approaches [123,124].

For ML-based DOX, the first step is still to identify process-structure-property parameters of interest and to classify them as either inputs or outputs relative to the desired relationship that is to be determined, as is done in traditional DOX. As more parameters are added, the size of the design space grows. Once the scope of the design space has been defined, the next step is to generate an initial dataset (i.e., initial information). The first tasks can be designed with traditional DOX methods – often, an approach as simple as selecting an initial uniform sample from the design space. In addition to generating an initial dataset, a response function must be defined to interpret the relationship between the inputs and outputs. One example is a regression model of the process parameters (inputs) and material properties (outputs). A random forest algorithm trains many regression algorithms, each on a subset of the experimental data.

Random forest algorithms are ensembles of a type of simple regression algorithm called a classification and regression tree or a decision trees. Decision trees can be used for both classification and regression. Consider a design space that an engineer wishes to explore represented as a matrix, such as

Feature 1	Feature 2	Feature 3	Property 1
$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	y_1
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$x_{n,1}$	$x_{n,2}$	$x_{n,3}$	y_n

where $x_{1,1}$ is the first parameter setting for feature 1, $x_{1,2}$ is the first parameter setting for feature 2, and y_1 is the first property measurement for the associated position in the design space, out of n total measurements. This design space could be represented as a matrix by

$$\mathbf{B} = \begin{bmatrix} x_{1,1} & x_{1,2} & x_{1,3} & y_1 \\ \vdots & \vdots & \vdots & \vdots \\ x_{n,1} & x_{n,2} & x_{n,3} & y_n \end{bmatrix} \tag{17}$$

The rows of $\mathbf{B}$ represent different observations in the design space and the columns of $\mathbf{B}$ are different parameters or properties. The goal of parametric analysis is to map different values of x to a property y.

Decision trees begin by taking a samples from the design space – rows in $\mathbf{B}$ – and computing a split in one of the features (columns) that best classifies the data point. Consider a set of three experiments that has

feature-property (x, y) pairings of $(0.1, 0)$, $(0.2, 0)$ and $(0.3, 1)$. The decision tree computes every possible partitioning of x and computes a misclassification error called the Gini impurity, defined as

$$I_G(x) = \sum_i^J p_i(1 - p_i) \tag{18}$$

where p_i is the percentage of samples classified into class i for each split out of J classes. For our fictional example, $J = 2$. Consider a split along the value $x = 0.1$ where values less than or equal to 0.1 are predicted to have $y = 0$ and values of x greater than 0.1 are predicted to have $y = 1$. The Gini impurity can be calculated for each side of the split. For the case of $x \leq 0.1$, all the samples provided (only one sample, in this case) have an associated y value of 0. Therefore, the Gini impurity would be

$$\begin{aligned} I_G(x \leq 0.1) &= \frac{1}{1}\left(\frac{1}{1} - 1\right) + \frac{0}{1}\left(\frac{0}{1} - 1\right) \\ &= 0. \end{aligned} \tag{19}$$

The value 0 is the lower bound for the Gini impurity, thus this split produces perfect classification for values sorted into $x \leq 0.1$. However, the Gini impurity for the remaining values becomes

$$\begin{aligned} I_G(x > 0.1) &= \frac{1}{2}\left(\frac{1}{2} - 1\right) + \frac{1}{2}\left(\frac{1}{2} - 1\right) \\ &= 0.5. \end{aligned} \tag{20}$$

This higher value of the Gini impurity indicates that splitting feature x along the value 0.1 produces an imperfect classification. If the split was chosen along 0.2 instead, the Gini impurity for both sides would be 0, a perfect classification. The Gini impurity can be extended to an arbitrary number of classes, allowing decisions trees to behave as regression algorithms as well as classification tools.

Decision trees compute every possible partition for each feature in the dataset such that the misclassification error, as defined by the Gini impurity, is minimized. However, decision trees are highly prone to overfitting. Random forests overcome this overfitting problem by training many different decision trees, each on a subset of the total dataset. A random sampling, with replacement, of design space coordinates (rows of **B**) are chosen, known as bootstrap aggregating, or bagging, and a decision tree is trained. Alternatively, or in addition to bagging, jackknifing selects a subset of features (columns of B) to prevent overfitting to specific features.

Training many different decision trees in this way allows a user to calculate uncertainty metrics for each prediction. The method of calculating uncertainty depends on how the random forest is being applied [124]. Once the random forest has been trained on the initial dataset, new points in the design space are given to the algorithm and the expected result is predicted.

The predictions made for new points in the design space can be characterized by several different response functions. A study by Ling et al. employed three response functions: the maximum likelihood of improvement (MLI), maximum expected improvement (MEI), and maximum uncertainty. Each response function has its own benefits. The MEI selects the best experiment for maximizing (or minimizing) a target value. The MU, as the name implies, selects the experiment with the highest uncertainty in predicted result. The MLI chooses the experiment most likely to have a higher (or lower) target value compared to the best previously observed value.

Often, parametric analysis is concerned with either exploring relationships in the design space or optimizing on a property (either minimizing or maximizing the property). The random forest can be trained on m many subsets of the n rows of **B**. Then, new points in the design space are chosen and their associated property y_{n+1} is predicted. If the goal is to maximize a property, then the next experiment to run can be chosen by the MEI or MLI. If the goal is to explore the design space then it is useful to choose the design space coordinate based on the MU.

Ling et al. trained a random forest to maximize the fatigue life of steel as a function of composition (among other test cases presented in the article) [125]. The features used in Ling's study included composition as a function of nine different alloying elements (C, Si, Mn, P, S, Ni, Cr, Cu, Mo) as well as thirteen different processing steps such as heat treatment temperature. The total dataset used had 437 tests of steel fatigue as a function of the features. The random forest algorithm was used to choose experiments to run balancing maximum predicted fatigue life with uncertainty in the prediction. Ling's random forest approach found the composition and processing combination with the best fatigue life in fewer than 50 experiments out of the 437 possible options when using the MLI. The sequential learning workflow used by Ling, as well as the performance of differently trained random forest algorithms and response functions is shown in Fig. 7.

Random forests have been applied successfully to a range of applications in materials science. They have been used to discover new thermoelectric materials [127]. They have also been used to model material properties such as thermal conductivity in half-Heusler semiconductors [128] and to break down fields for dielectrics [129]. A review article detailing many optimization algorithms for design of experiments can be found in Shan et al. [45]. Adoption of machine-learning assisted design of experiments algorithms can rapidly increase the rate at which the relationship between AM process parameters and material properties are understood.

3.1.3. Topology optimization and generative design

Alloy design and process design are based upon process-structure-property relationships of materials, independent of part geometries. These optimizations reduce manufacturing costs and times and help attain targeted properties. Analogous optimization approaches can be applied to design the geometry-material-performance relationships of AM parts. Such approaches are called topology optimization methods. For structural materials and their parts, a common goal is to optimize the geometry to maximize the load bearing capacity, stiffness, or lifetime while minimizing the mass of the part. The ability to manufacture the unique, complex geometries determined by topology optimization algorithms for (nearly) the same cost as simple geometries designed for subtractive manufacturing processes is one of the greatest promises and appeals of additive manufacturing. One of the frontiers in research driven by AM processing, in which materials and part topologies are simultaneously manufactured, is to integrate alloy processing optimization with topology optimization to create concurrent optimization methods. Thus, part performance becomes integral to the material manufacturing optimization process in AM. Hence, we proceed to introduce topology optimization to the materials researcher while also discussing potential uses for machine learning to advance topology optimization.

Topology optimization can be applied for several optimization objectives, including compliance minimization, stress constraint or natural frequency maximization. Manufacturing constraints such as overhangs and support structures found in AM have also been added to the optimization process and improve the applicability of the result [131–135]. Support structure optimization that minimizes the amount of material used in supports has also been researched [136–138]. Other additive specific algorithms have been designed for optimizing density of parts [130]. By determining the ideal material layout, the final design can maximize performance for a given weight, minimize weight for an objective function, or reduce manufacturing costs by reducing the material used. However, TO is a local optimization method. Global design optimization usually requires statistical analysis of many TO simulations.

ML can help reduce the computational time necessary for a TO analysis. This approach allows for faster convergence to a TO result, as well as produces multiple designs efficiently for a researcher to better explore the possible design options. The process of producing many outputs for a given set of conditions is known as generative design.

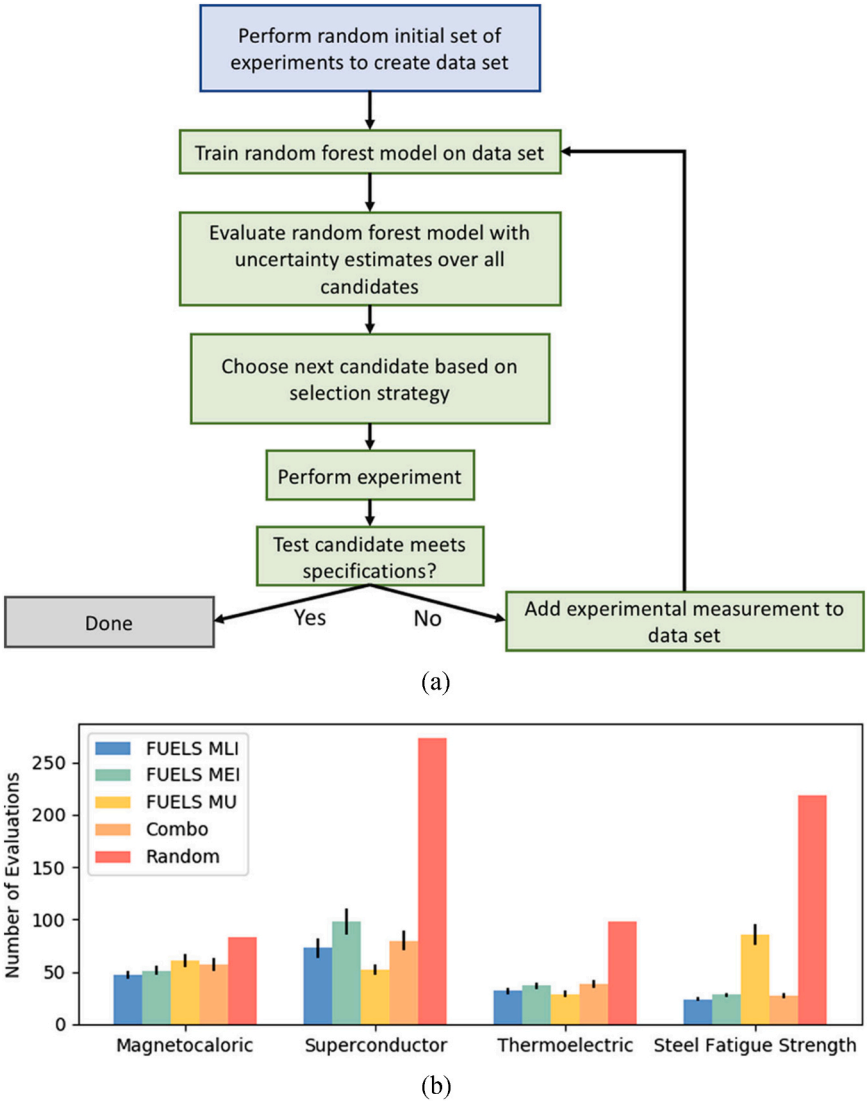


Fig. 7. Application of a random forest algorithm to find optimal material candidates for four different datasets: magnetocaloric materials, superconducting materials, thermoelectrics, and steels. A random forest algorithm was used with four different response functions: maximum likelihood of improvement (MLI), maximum expected improvement (MEI), maximum uncertainty (MU), and the COMBO Bayesian optimization approach [125,126]. The algorithm in (a) was used to speed up the experimental design process. In every case, the optimal material for the application in the dataset was found more quickly through sequential learning than through random guessing. The random forest approach was compared against COMBO, another sequential learning tool. The figure in (b) demonstrates how much more quickly the random forest algorithm was able to find an optimized state than random sampling of experiments to perform.

One approach is to change the topology of a region of a part by applying local filters to CAD models; mathematically, these filters are the same ones shown in Eq. (3). A physical representation of a filter matrix in 2D and 3D can be seen in Fig. 8. Topology optimization proceeds by generating a CAD model of an AM part and modeling its performance, such as testing performance under mechanical load through an FEA simulation. Filters are applied to the CAD mesh that selectively remove material from the part. Then, the mechanical performance of the new part is modeled, followed by further material removal. This process proceeds until either a minimum weight/volume condition is met or the mechanical performance of the part is degraded.

These filters, filters that identify which material to remove, can be learned by a type of machine learning algorithm called a convolutional neural network (CNN). Convolutional neural networks have been found to be well-suited for data containing multiple arrays, especially for image recognition tasks [142]. The input is separated into different channels, such as RGB for three channels of a color image input, and manipulated through different stages of the network, called layers. Commonly used layers in these networks include convolutional, pooling, and fully connected. Convolutional layers are divided into varying feature maps that abstract the input to smaller, localized regions for analysis. Pooling layers clusters the outputs from the previous layer and outputs either the maximum or average value from the cluster, reducing the dimensionality of the problem. Fully connected layers connect the outputs from the previous layer with the inputs of the next.

Using a convolutional neural network, Cang et al. and Banga et al. present similar approaches to produce "one-shot" tools for two- and three-dimensional TO, respectively [139,143]. One-shot tools produce an optimized structure directly from a starting topology, as opposed to iterative tools that require multiple passes of the algorithm to reach an optimized state. The inputs of the CNN were aspects of the initial part geometry and expected loading conditions. Features given to the model included the force experienced by the part, fixed boundary conditions, minimum mass and density values, and the locations of mass in the part. Their training and validation databases a dataset of optimized topologies generated via traditional TO. The goal of the CNN was to predict an optimized geometry for a starting structure in one pass through the model using knowledge of the loading conditions and part geometry. The results from both works show similar accuracy between the "one-shot" result and the ground truth found from traditional non-ML methods. Such methods greatly reduce the computational time, allowing for greater design exploration before finalizing the result. The architecture for the CNN used in Banga et al. can be found in Fig. 9a.

An extension of the convolutional neural network is the generative adversarial network (GAN). The methodology for GANs involves two neural networks in competition with each other: a generator and a discriminator. The generator attempts to create data similar to that of an existing database. The discriminator has access to the database and

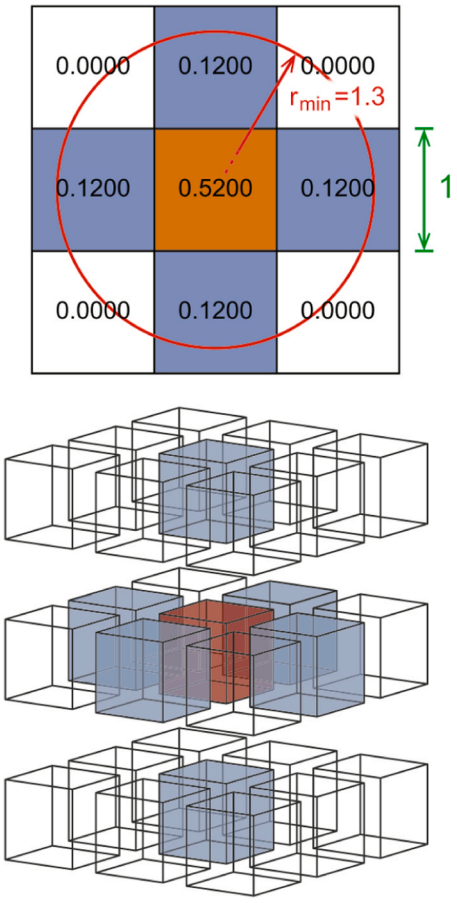


Fig. 8. Examples of filters that are applied to CAD meshes to change the geometry of the part [130]. The filters can be applied to remove material for weight reduction or add material to prevent part warpage during manufacturing.

discerns which data samples are from the database and which are from the generator [140]. The goal of the generator is to generate data that the discriminator cannot accurately classify into database or generated datasets. As the discriminator improves at discriminating between artificially generated data and user-provided data, the generator learns to produce better and better artificially generated data. The ultimate goal is a generator network that learns to produce high-quality optimizations of an input topology. A flow diagram of a GAN from Crewswell et al. can be found in Fig. 9b. Further information about the applications of GANs, including image synthesis and superresolution, can be found in Crewswell et al. [140].

Yu et al. uses a combined CNN-GAN to perform a superresolution for TO, upscaling a coarse mesh result to a higher resolution without the added computational time to directly compute the high resolution result [144]. First, a CNN was trained to predict low-resolution optimal geometries based on provided boundary conditions. The CNN used information such as minimum mass fraction, location of applied load, and fixed boundary conditions to predict the best topology at a low resolution. Then, a GAN was trained with random sampling of low and high resolution TO results as inputs and ground truth database, respectively. The GAN was trained to generate high resolution topologies from low resolution inputs. The low resolution output of the CNN was given to the generator; the generator then produced a high resolution topology from the given low resolution input [144]. The results showed very high agreement between the generated structures and a set of training structures generated using an open source code [145]. The result from this combined network was within 3% of the expected ground truth pixel values and produced it in 0.06% of the time compared with traditional TO [144].

For generative design, genetic algorithms and GANs are well suited as both architectures are designed to produce multiple optimal designs. Lohan et al. and Zimmerman et al. use the genetic algorithm to effectively search for optimal solutions for heat transfer and fluid optimization [146,147]. Using the genetic algorithm, high performing designs were iterated upon in subsequent steps, producing multiple optimal designs for the researcher to choose. As an example of a GAN used in generative design, Oh uses data mining to collect wheel examples to train a GAN and generate unique designs [141]. The network generates a random set of input variables to influence a topology optimization stage of the network. Through training, the generator network attempts to produce new designs similar to the examples collected through data mining. The discriminator network is then trained using the sampled outputs from the generator network and the data mined examples to determine which are generated and are from data mining. Through many training iterations, the generator network produces designs indistinguishable from the database. Examples of the generated designs from this network are shown in Fig. 9c.

The examples provided only present current research incorporating machine learning in TO and is not an exhaustive review of all applications of TO. A general review of topology optimization advancements for additive manufacturing can be found in Liu et al. [135].

3.2. Machine learning assisted modeling of additive manufacturing

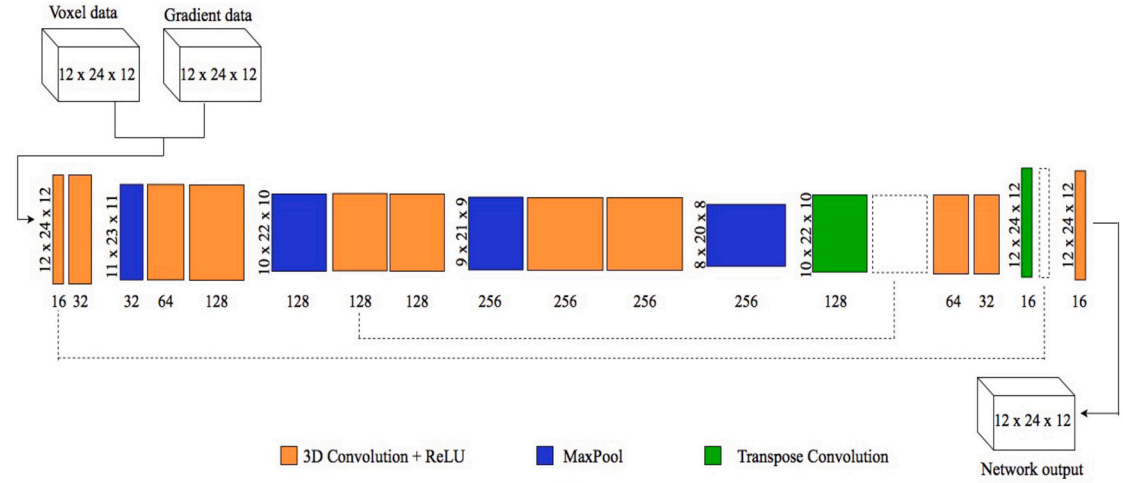
As previously discussed, the design space of AM experiments is often vast (e.g., Fig. 1). While the design of process parameters is often integral to the material design methods reviewed in the previous sections, there are some additional process-centric engineering objectives where machine learning methods may also be beneficial. This section reviews the use of machine learning algorithms to aid in computational design of additive manufacturing process developments. Martukanitz et al. published a full ICME investigation of AM [148]. There are two modeling scenarios that plague the advancement of AM: the case where a model exists but current numerical methods are too expensive to simulate the model; or the case where a model does not exist. Put slightly differently, in either case $\mathbf{y} = f(\mathbf{x})$ exists but cannot be computed in a reasonable amount of time; or $\mathbf{y} = f(\mathbf{x})$ does not exist.

Machine learning algorithms have addressed both these cases. In the first case, ML algorithms provide an alternative numerical method for calculating $\mathbf{y} = f(\mathbf{x})$ based on experimental measurements of $\mathbf{y}$ and $\mathbf{x}$, or based on the results of previously run simulations. Machine learning algorithms have also been developed to help visualize trends in high dimensional spaces, allowing researchers to study complex relationships and ask deeper questions. For the second case, ML algorithms provide a form of $\mathbf{y} = f(\mathbf{x})$ from observations (measurements) of the relationship between $\mathbf{y}$ and $\mathbf{x}$.

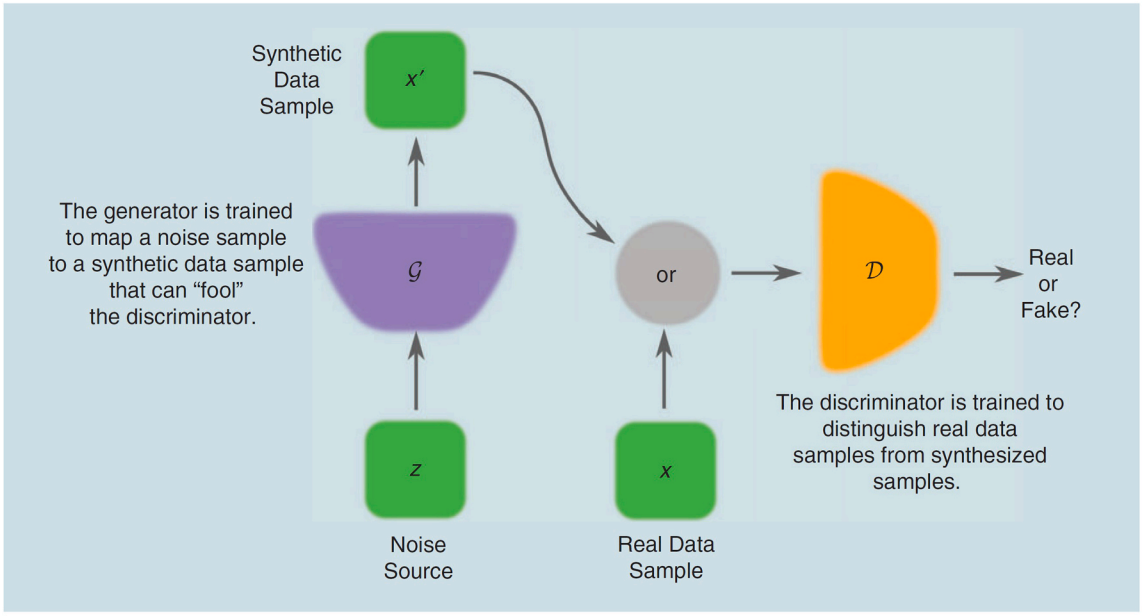
3.2.1. Machine learning as numerical methods for modeling

There is a suite of numerical methods that have been adopted by the materials science community for computing models of material phenomena. Finite element methods are some of the most common methods. Feedstock, heat source, and melt pool dynamics have been modeled by finite element methods [149–151] or finite volume methods [152]. Some AM-specific tweaks to the finite element method have been developed, such as the quiet/inactive method of Michaeleris et al. [153]. They have also been applied to microstructure development [111]. A review of finite element methods for AM can be found at [154]. Models of grain growth in AM have been solved using both phase field numerical methods [155–158], and cellular automata [95]. Francois et al. provide a review of ICME approaches across spatiotemporal scales [159].

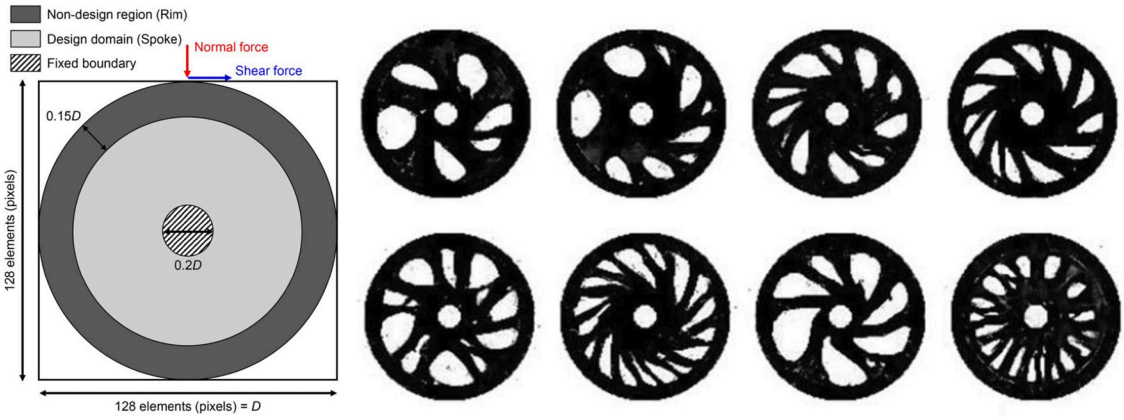
The success of these numerical methods have been in solving complex thermomechanical problems for engineering application. In AM, the number of models that need to be simultaneously considered/computed and the scale of the manufacturing process causes the



(a)



(b)



(c)

Fig. 9. The architectures for convolutional neural networks (CNN) and generative adversarial networks (GAN) are visually described. (a) The network architecture of the CNN used in Banga et al. with the voxel and gradient input data [139]. The dimensions are described as Height × Length × Width with channels below their respective layer. (b) The models and data used for training a generator and a discriminator in a GAN [140]. (c) Examples of Generated designs produced from GAN

network in Oh et al. for the design problem shown [141].
Image in (a) taken from [139]; image in (b) taken from [140]; image in (c) taken from [141].

computational complexity of these methods to grow quickly. Machine learning can make the modeling process more efficient through three primary applications: .

- Determine a priori which models *not* to run.
- Reduce dimensionality by discarding inputs or physics that are not relevant or that do not have an appreciable impact.
- Compute the same relationship using ML as the numerical method, instead of using explicit methods like FEA, cellular automata, etc.

In the case that models can be run, but are time-intensive, it behooves the researcher to run as few models as necessary to understand the material response. ML can identify which models will produce the most useful information, informing model choice to the researcher. In another case, ML can be used to identify physics that do not significantly impact the outcome of a model. Reduced-physics models can then be created with a reduced computational burden. In some cases ML can compute the same result as the explicit model with significantly less computational cost (under certain conditions and assumptions).

Dimensionality reduction algorithms identify which parameters are relevant to model in an ICME approach and which are not, enabling future ICME investigations to achieve the same result faster. Materials science has long had a need for dimensionally reduced, computationally accurate models. Some of the first applications of machine learning in materials science was for dimensionality reduction. Dimensionality reduction has been applied to find process-structure-property relations across multiple material length scales [78,160–162]. Homer applied dimensionality reduction to relate the impact of local atomic environments on mesoscale properties like atomic mobility at grain boundaries, demonstrating the benefit of the technique for advancing ICME [163].

Statistically driven approaches can focus on the parameters in $\mathbf{x}$ that strongly impact AM model outputs, leading to a dynamically guided design of experiments. In design of experiments, a random forest is trained on previously completed experiments. These are the rows of the matrix $\mathbf{B}$ in Eq. (17). Then, new points in the design space that have not been observed are given to the algorithm and predictions about the output are made. Dimensionality reduction using random forests proceeds differently.

The random forest is trained on subsets of the data in $\mathbf{B}$. The importance of a feature in the dataset is tested by randomly shuffling the values of one of the columns of $\mathbf{B}$. If randomly shuffling the values of a given feature does not significantly impact the prediction accuracy of the random forest then it is likely that the feature is not important. Towards Data Science provides a more in-depth tutorial on using random forests for feature importance determination [164]. This approach can be applied in computational models that consider many different physics. Several models are run under different initial conditions in the design space. The entries of the matrix $\mathbf{B}$ are the inputs and predicted outcomes of the model. If the exclusion of a model input does not significantly impact the random forest's prediction of the model output, then that input can likely be ignored in future simulations, saving computational time and reducing the number of models that need to be run.

Kamath used a random forest algorithm to screen out irrelevant modeling parameters for predicting maximum density of additively manufactured parts [165]. Kamath started with an experimental dataset of manufacturing parameters and multiple modeling methods. An Eagar–Tsai simulation of a Gaussian laser beam on a powder bed was used to model thermal conduction during manufacture, as well as the computationally more expensive Verhaeghe model. The Eagar–Tsai model originally began with four inputs (laser power, speed, beam size, and powder absorptivity) and a design space of 462 possible input combinations. Kamath used random forests to determine which input was most important for achieving fully dense parts. If simulations are time-intensive to run then 462 different simulations may be out of the question. The computational dataset was complemented with an experimental dataset of measured melt pool widths at various printing conditions. Identifying which parameters do not impact the final result reduces the size of input combinations, therefore reducing the number of computations or experiments to be performed.

Kamath identified that laser speed and power were the most important inputs out of the four to determine melt pool depth and shape. Now that the important physics have been identified, the researchers can proceed to the more expensive Verhaeghe model with knowledge of what parameters to vary. After determining the most important inputs, the same regression tree was applied to find optimized manufacturing conditions for fully dense parts. However, instead of identifying which features impacted the model standard deviation, the machine settings that maximized y were found.

A final technique for reducing the burden of computational models requires expressing model data in a matrix and performing matrix factorization. As before, model inputs can be formed into a matrix, $\mathbf{X}$, whose rows are coordinates in the design space. Matrix factorization techniques represent correlations in large datasets in a simplified way. The matrix $\mathbf{X}^T\mathbf{X}$ is a measure of covariance within $\mathbf{X}$. The matrix $\mathbf{X}^T\mathbf{X}$ can be very large due to the design space of additive manufacturing. One type of matrix factorization, called Principal Component Analysis (PCA) represents the data matrix $\mathbf{X}$ as

$$\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T \tag{21}$$

where the columns of $\mathbf{U}$ are the eigenvectors of $\mathbf{X}\mathbf{X}^T$ and are called the principal components of $\boldsymbol{X}$. Similarly, the columns of $\mathbf{V}$ are the eigenvectors of $\mathbf{X}^T\mathbf{X}$. The matrix $\boldsymbol{\Sigma}$ is a diagonal matrix whose entries are the singular values of $\mathbf{X}$. The first singular value, which corresponds to the first column vector in $\mathbf{V}$, has the highest variance (most information); the second singular value, the second most; and so on. Therefore, regression can be performed on one, a few, or many of the principal components to predict new model results using considerably less data than present in $\mathbf{X}$, but with a minimal loss of information and a minimal reduction in model accuracy. Materials science studies have used PCA previously to re-represent large datasets in simpler forms, such as predicting the formation energies of crystal structures from a lower dimensional space [166]. A review of applications of PCA in materials science can be found at [167].

In additive manufacturing, PCA can serve to reduce the number of features in the design space. The new vectors can then be used as regression model inputs for prediction of material properties based on trends observed in dataset $\mathbf{X}$.

3.2.2. *Machine learning for visualizing trends in the design space*

Visualizing relationships across high dimensional spaces helps researchers develop an intuitive understanding of data relationships that exist, an intuition that helps guide data preprocessing, feature engineering, model selection, and model training. However, visualizing an n-dimensional distribution is difficult. Process maps are commonly employed in AM to visualize 2D slices of the n-dimensional AM design space [42]. The Ashby plot is a well known generalization of process maps in materials science. Ashby plots show material properties as functions of two design coordinates, such as plotting mechanical strength of various alloys as a function of density and cost to produce. The process maps in welding and AM are more specific versions of Ashby plots. Process maps chart the possible values of machine inputs and identify regions of the design space with similar properties. A commonly employed process map in AM of Ti-6Al-4V describes grain morphology

as a function of solidification velocity R and temperature gradient G [168]. Extending process maps to n many process variables would require $\binom{n}{2}$ plots. Defining and examining metrics of similarity in an n dimensional space can reveal trends in a human interpretable way without relying on multiple 2D process maps.

t-distributed Stochastic Neighborhood Embedding (tSNE) is a visualization technique that measures distances in a high dimensional space and then projects data points onto a two dimensional plot. The similarity of all data points in the design space with each other is used to fit a distribution of similarities. The tSNE algorithm begins by fitting a probability distribution to all $\mathbf{x}$'s contained in a dataset. Relationships in n dimensional space are assessed through a kernel function $\kappa(\mathbf{x}, \mathbf{x}')$ that measures similarity between points in the design space. A commonly employed kernel is the Gaussian kernel

$$\kappa(\mathbf{x}, \mathbf{x}') = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{\mathbf{x} - \mathbf{x}}{'} 2\sigma^2\right] \tag{22}$$

where σ is a user-specified or fit standard deviation in the distribution of points in the design space. This kernel function assesses distance in the n dimensional space and assigns a similarity value between $\left[0, 1/\sqrt{2\pi\sigma^2}\right]$.

After the n dimensional dataset is fit, then a 2 dimensional coordinate $\mathbf{x}^*$ is assigned to each $\mathbf{x}$. The reason for choosing a 2 dimensional coordinate is so that the final result can be visualized on a 2D plot. The tSNE algorithm fits a probability distribution to the n dimensional data set first, then assigns values to each $\mathbf{x}^*$ such that they have the same probability as the associated high-dimensional $\mathbf{x}$. Once the probability distributions have been assigned, the $\mathbf{x}^*$ values can be visualized on a 2D plot to investigate trends.

The benefit of tSNE is that points that are close together in the n dimensional space appear close together on the 2 dimensional plot. This gives AM modelers an idea of how machine inputs and material behavior are distributed in the n dimensional space through a 2 dimensional visualization. Traditional process maps provide similar input/output relationships but are limited in the amount of processing parameters they can interpret at once. A comparison of process maps and tSNE is shown in Fig. 10.

3.2.3. Machine learning to create models of additive manufacturing processes

Another problem, equally important to solving models, is the creation of models for additive manufacturing problems. Scientists cannot engineer the additive manufacturing process without an understanding of how process parameters (inputs) impact material properties and performance (outputs). The generation of models in AM is a difficult task due to the large amount of physics that can be incorporated.

Many traditional material models from science and engineering have been applied to additive manufacturing, including thermal history models of heat transfer through the part [153], residual stress build up during manufacturing [170,171], and thermal signatures such as cooling rate and temperature gradient [115,172]. King et al. provide a review of the physics of AM modeling [173].

Phenomenon that are difficult to study experimentally, such as flow within the melt pool, are best studied through modeling approaches. Though expensive, full-physics modeling is often necessary to understand how physics at different scales interact to impact the AM process. If the computational expense of a simulation is too high then performing simulations at all relevant manufacturing conditions can be infeasible. While it is useful for optimization and visualization, reduced order models are unlikely to capture the full dynamics of solidification in AM.

Within the context of the literature reviewed herein, a surrogate model is a regression model that estimates the results of high-cost simulations. Surrogate models are regressed on the inputs and results of previously run simulations. Then, the surrogate model interpolates simulation results at new coordinates in the design space. Surrogate models preclude the need for running computationally expensive simulations for every possible manufacturing condition. Formulating surrogates can be as simple as performing linear regression between simulation inputs and results, but are often more complex. The accuracy of a surrogate model is dependent upon how many previous simulations have been run and at how many different points in the design space.

Tapia et al. built a surrogate model for laser powder bed fusion of 316 L stainless steel. They were concerned with predicting the melt pool depth of single-track prints solely from the laser power, velocity, and spot size [174]. The dataset used to build the surrogate was computationally derived, based on previous simulation methods used by the same research team [175]. In particular, they used the results from a computationally expensive but high-accuracy melt pool flow model of Khairallah et al. [150]. They ran powder bed simulations at various laser powers, velocities, and spot sizes, and the model told them the depth of the melt pool, amongst other information. The datasets provided enough information for a surrogate model to be trained to predict simulation results (Fig. 11).

To build their surrogate model, Tapia used a machine learning algorithm known as a Gaussian process model (GPM). A common model assumption in Gaussian process modeling is

$$z(\mathbf{x}) = y(\mathbf{x}) + \epsilon(\mathbf{x}) \tag{23}$$

where $y(\mathbf{x})$ is the approximation (surrogate) of the simulation process, $\epsilon(\mathbf{x})$ is a stochastic, randomly distributed noise in measurement, and $z(\mathbf{x})$ is the value given by a simulation. The primary goal in GPMs is to find model parameters for the mean process $y(\mathbf{x})$ and a covariance function $\kappa(\mathbf{x},\mathbf{x}')$, which is a function of similar form to Eq. (22). Fitting a Gaussian process model often begins with assuming a model function for

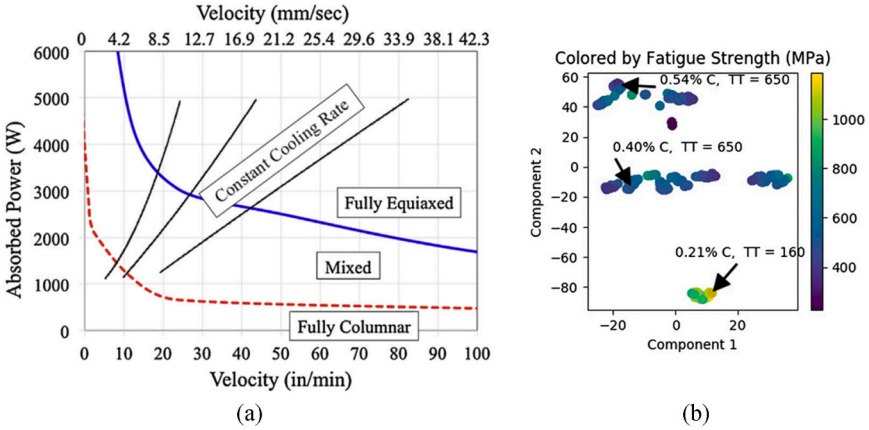


Fig. 10. Comparison of a traditional process map and tSNE plot. (a) A process map for predicting microstructure characteristics based on absorbed power and deposition velocity in electron beam wire feed additive manufacturing. (b) A tSNE plot from Ling's study showing clusters of samples with similar fatigue strengths [124]. While process maps can be useful for predicting manufacturing outcomes they are limited by only showing the behavior of two process parameters at a time. The tSNE algorithm can cluster data based on many manufacturing inputs simultaneously and then display that information in a 2D plot, allowing engineers to study how processing parameters lead to good or bad material properties.
Image reproduced from Gocket et al. [169].

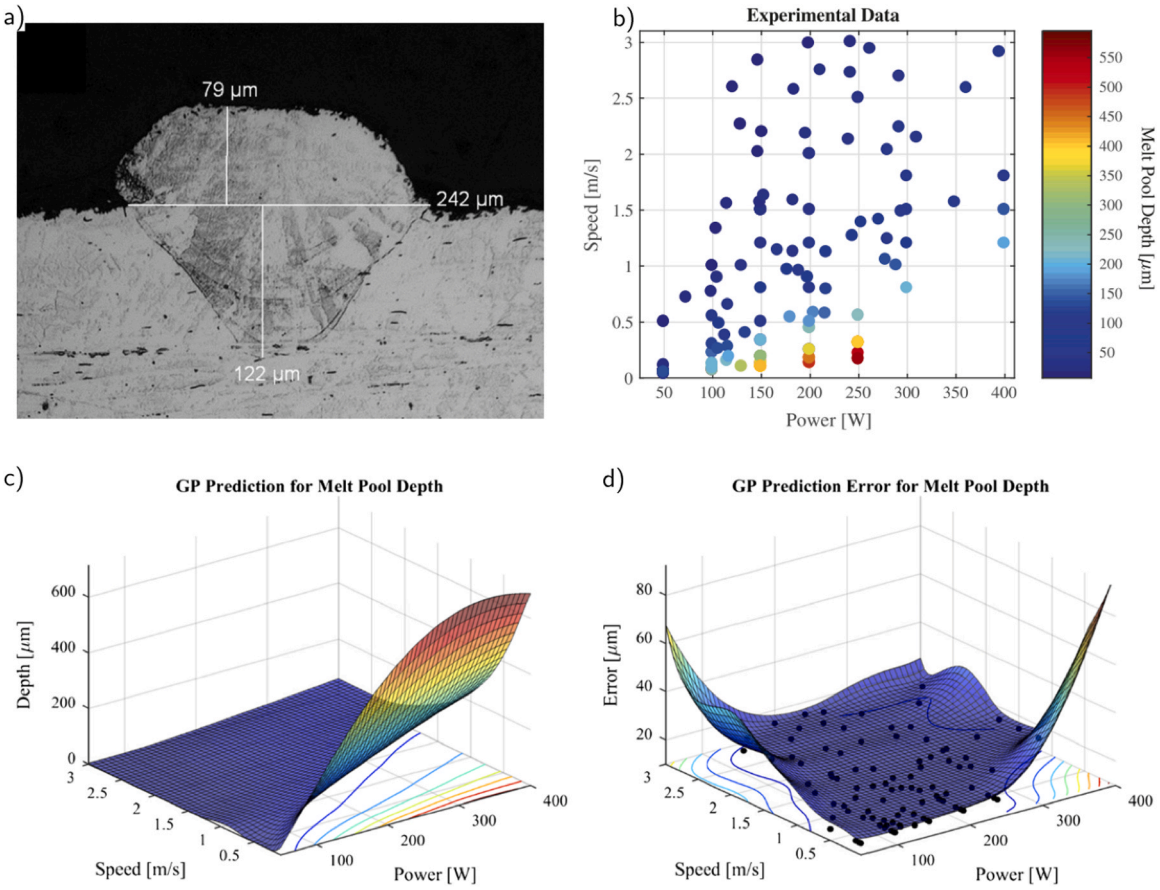


Fig. 11. Input data, response surface $y(\mathbf{x})$, and error $\epsilon(\mathbf{x})$ of Tapia's study in predicting melt pool depth for laser powder bed fusion [174]. (a–b) The input data used to train a Gaussian Process Regression model for predicting melt pool shape and depth. They started with a sparse sampling of the design space to build the model. (c–d) The Gaussian process model predictions for melt pool depth as a function of two input parameters $y(\mathbf{x})$ (left) and the associated prediction errors $\epsilon(\mathbf{x})$.

covariance, fitting the model parameters such as σ to the observed values $z(\mathbf{x})$, then using those model parameters to predict simulation results $y(\mathbf{x})$ at other locations in the design space.

Tapia used Bayesian statistics to develop a probabilistic model that predicted melt pool depth from simulation inputs. They were able to successfully predict the outcomes of both high-fidelity simulations and experimental measurements solely by analyzing trends in previously obtained results. In particular, they were able to accurately predict the melt pool depth at a value that had never been observed before, either computationally or experimentally. For future investigations, predictions by the surrogate model can be relied upon instead of running a simulation or experiment. Regression models such as this provide engineers with faster routes toward optimized manufacturing states by predicting manufacturing at a wide range in the design space based on only a few initial experiments.

Gaussian Process Models provide robust uncertainty metrics on the predictions they make. Uncertainty estimation is important in materials informatics because it enables scientists to know how confident their models are in predictions in various regions of parameter space. Some machine learning models do not have straightforward ways of assessing model error [176].

Another benefit of GPM is that it aids in inverse design and design space visualization. GPMs can explicitly identify regions of the design space that will maximize or minimize a value. In the case of Tapia et al. response surfaces were created from the GPM that visualized the depth of melt pools as a function of laser power and speed. Doing so allows engineers to identify regions of the design space that provide specific material responses, an important tool in optimization for additive.

Machine learning is not only limited to ex situ experimental investigations or modeling approaches. Ideally, machine learning can be used to solve multiobjective optimization functions where multiple aspects of the AM process are optimized at once – energy density, melt pool shape, heat transfer, grain growth, and the list continues. Models can be created that solve this multiobjective optimization problem and present to the engineer what an optimal manufacturing process looks like. Actually creating that optimal process will require tight control of the manufacturing process. Machine learning models trained on correlations between build parameters, the dynamic response of the system, and the final part properties can be combined with real-time computer vision to simultaneously observe, characterize, and control many different aspects of the manufacturing process.

3.3. Process monitoring and control

The numerousness of signals to measure in situ for AM warrants use of quick, efficient, and robust signal processing methods for process monitoring, feedback, and control. These signal processing algorithms are closely related to machine learning. They serve as tools in their own right, and can also pre-process data for use in other machine learning applications, like clustering and regression. Computer vision is one class of image recognition algorithms that has been developed for automated feature identification in signals. Intelligent computer vision uses ML algorithms to identify objects and features in a wide variety of data types. We proceed to discuss the potential uses of data analytics and ML to advance our ability to directly study and control AM process technologies.

It is important here to reassert the importance of the relational hypothesis. If one tries to estimate a property from an in situ measurement

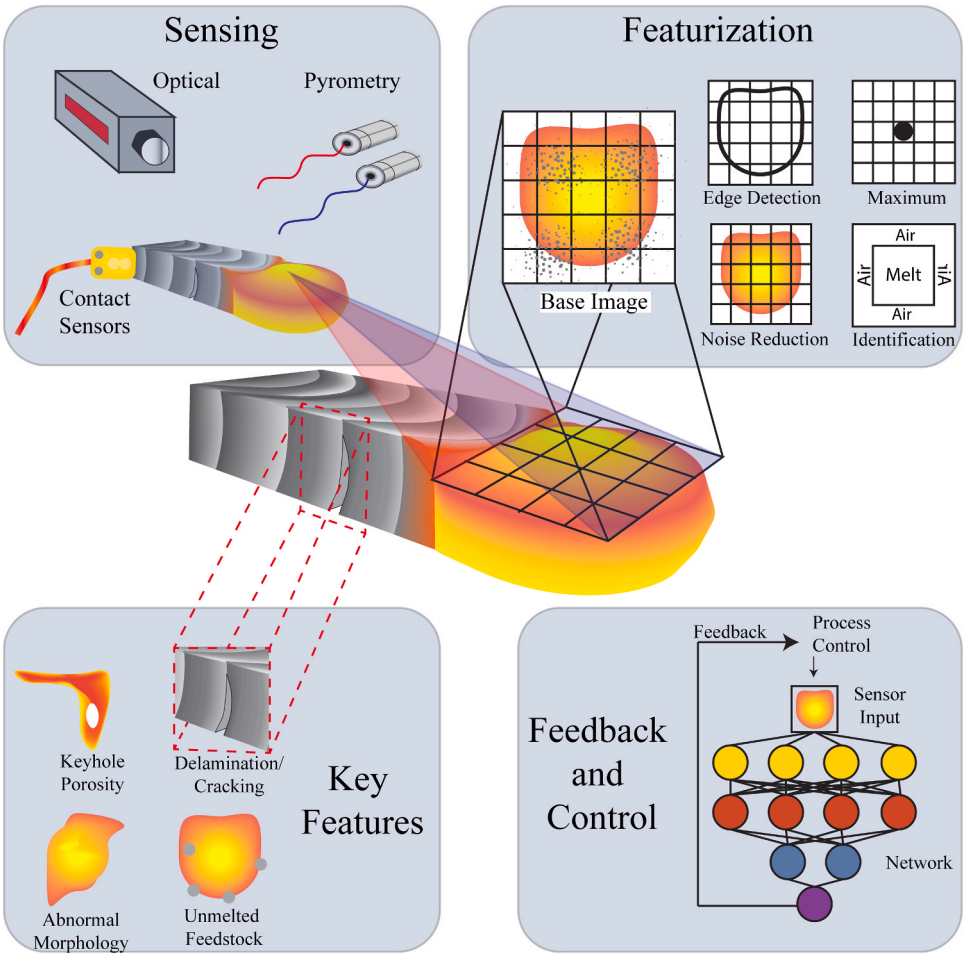


Fig. 12. A few examples of data types, data sensors, and features to detect in a laser powder bed fusion manufacturing process. The wide range of signals to monitor then control makes feedback and control in AM especially difficult. Computer vision techniques can be applied to automatically detect features of interest across multiple data types and data sensor simultaneously.

one must ensure that the signal being acquired has features that correlate to the property. Otherwise, a machine learning algorithm may end up identifying spurious correlations or uninterpretable relationships.

3.3.1. *In situ process monitoring and feedback*

In situ monitoring, feedback, and control has been consistently ranked as one of the most-needed technologies for advancing additive manufacturing [177–179]. The combination of rapid solidification and the small length scales of AM solidification can make traditional process monitoring approaches difficult. Furthermore, there are many processes/problems to monitor for during the manufacturing process, with equally as many sensor types for monitoring as shown in Fig. 12. Machine learning can fill in gaps that leverage correlations and relationships from previous measurements, observations, and responses.

Process monitoring involves acquisition of real-time signals that can reveal information about a wide variety of phenomenon during manufacturing. Many developments of in situ process monitoring technologies are focused on controlling (a) microstructure growth or development; or (b) the prevention of defect formation.

There are in-situ experiments being performed to inform models of the additive manufacturing process. In situ experiments advance our understanding of AM, as well as advance feedback and control for AM, through several outcomes. In some cases, in situ studies reveal what a ‘good’ or ‘bad’ AM process looks like. They also inform researchers of those conditions that must be met to achieve a desired outcome or prevent the formation of a defect. In situ experiments also push the development of sensor technology for AM. While sensor technology will not be covered in this review it is an important topic for the advancement of AM technology. Purtonen et al. wrote a review of common sensing methods for laser based manufacturing [180].

Early experiments using in situ monitoring for AM focused around either the ability to measure thermal signatures accurately or relating key features of the solidification process to important material properties. McKeown et al. has used dynamic transmission electron microscopy to measure solidification rates in powder bed AM [181]. Bertoli et al. have also characterized cooling rates using high speed imaging [182]. Raplee et al. have used thermography to monitor the solidification and cooling rates of electron beam powder bed fusion, relating the temperature profiles to defect and microstructural characteristics [183]. Distortion of parts due to thermal cycling was investigated by Denlinger et al. by means of thermocouples in contact with the build substrate [121]. Guo et al. used synchrotron X-ray imaging to characterize the dynamic behavior of spatter during laser-based AM [184]. Leung et al. likewise used synchrotron X-ray imaging to characterize defect formation and molten pool dynamics during laser powder bed fusion [185]. Based on the behavior they observed, Guo et al. were able to suggest control mechanisms for minimizing spatter during manufacture. Everton et al. provide a review of in situ monitoring for metal AM [186]. All of the data being recorded in these studies can be used as features for training machine-learning based feedback and control systems. The class of algorithms used in these cases is called computer vision.

The type of data being collected in situ is often in the form of time series or image data. In computer vision, as with traditional feedback and control, algorithms are used to identify deviations from a desired

signal. The power of computer vision approaches is their ability to simultaneously monitor and identify signal changes across multiple sensor types, as well as multiple different types of deviation from a single sensor. Examples include identifying a spike in temperature or a sharp change in intensity in an image indicating a deviation from a desired processing conditions. Image processing filters can be used to selectively modify or extract features in AM data. Image processing filters are mathematically analogous to those introduced for topology optimization (Section 3.1.3).

A filter is implemented as a mathematical operation, a kernel, applied to a window of time series data or an area of pixels in an image. For images, as previously discussed in Section 2.3, filters attempt to use local spatial information and a priori knowledge of the expected properties of the image to improve image quality and extract features, e.g., distinctive characteristics such as edges or regions of similar intensity (domains) that represent the boundaries or spatial extents of objects, phases, etc.

AM processes span several orders-of-magnitude in both length and time scales from ejected particles moving across the field of view in milliseconds to multi-hour builds and sub-millimeter melt pools to part-scale thermal distortions. Practically, then, in situ monitoring requires compromises in data collection rates and resolutions, and data processing filters are used to reduce noise and extract features, such as melt pool width, from the as-collected data. A comprehensive review of image filters is beyond the scope of this review, so the interested reader is directed to the many works on this topic, such as Vernon et al. [187]. However, three use cases are especially common and worth discussion here: reduction of high-frequency noise, also known as salt-and-pepper noise; additive noise reduction; and edge detection.

High frequency noise is characterized by sudden changes in intensity relative to the surrounding field. Although there are a number of possible causes, this may be caused by pixel-level variability or insufficiency in the detector, e.g. “dead pixels” or excessive gain. Median and conservative filters are commonly used when the fraction of noise pixels is large (1–10%) and small (<1%), respectively.

Additive noise, unlike high frequency noise, is a result of insufficient counting statistics, which may result from insufficient exposure time, or detector efficiency. A gaussian filter adjusts the intensity of each pixel according to the weighted intensities of neighboring pixels. Unlike median and conservative filters, a gaussian filter will soften edges, making adjacent domains less distinct.

Filters also have applications beyond noise reduction, primarily in object and feature detection. Detecting phenomena of interest during manufacturing is the first step to feedback and control mechanisms. Edge detection captures local changes in intensity to identify transitions between adjacent domains. Laplacian or Laplacian of Gaussian (LoG) filters themselves are sensitive to noisy images, identifying spurious edge artifacts, but are used as part of larger algorithms, such as Canny edge detection [188]. Canny edge detection include noise reduction to mitigate artifacts of LoG filters, and can be used to monitor melt pool shape and identify other features, such as unmelted powder particles attached to the build surface. Canny edge detection, along with other feature extraction algorithms, can be used to extract the features that characterize the build and can be used as part of a larger machine learning workflow to classify build quality. For example, these features can be used in learning algorithms to correlate characteristic features, such as melt pool width and hatch spacing, with particular behaviors, such as the formation of lack of fusion defects, in the manufacturing process. In this case, identification of a feature, or set of features, may be sufficient to indicate a particular process outcome.

Template matching is a computer vision method that can be used for automatic identification of common patterns. It involves the comparison of an unclassified input to a database of pre-identified patterns. For AM, template features include abnormal melt pool morphologies [189], inclusion of unmelted powder particles [190], and denudation near the melt zone [191]. The scale-invariant feature transform (SIFT) [192] and a variant, “Speeded-Up Robust Features” (SURF) [193] are both feature identification algorithms that can be used for template matching. Another template matching algorithm is the bag of visual words or dictionary method [49]. A collection (dictionary) of typical features from the AM process can be built based on features obtained from filters. The features measured in situ are compared with dictionary entries. If an

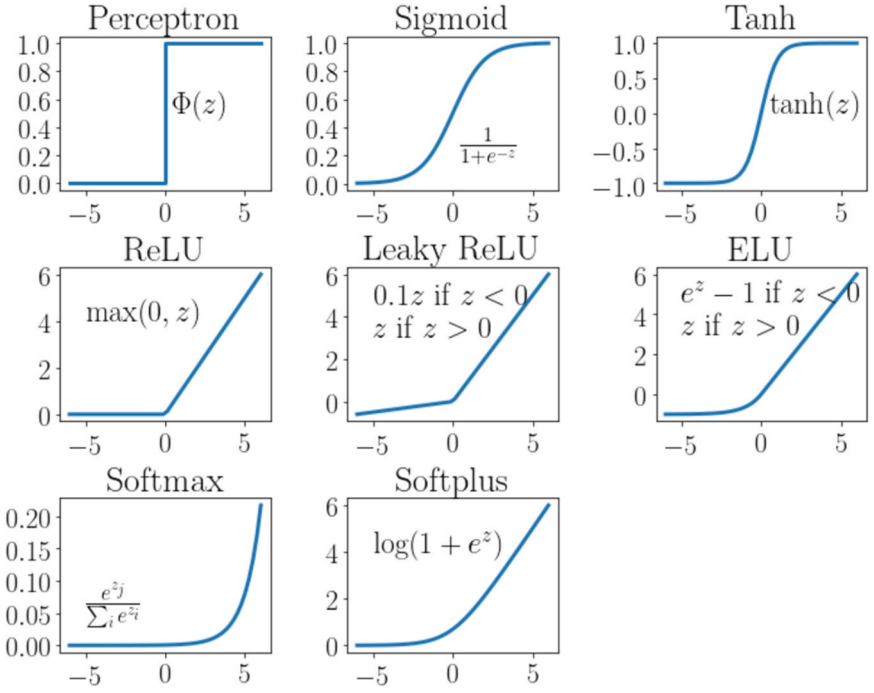


Fig. 13. Common activation functions in artificial neural networks (NNs) that introduce nonlinearity into the NN. The sigmoid is the archetype activation function because the closed form solution for the derivative of the sigmoid, which is used during model fitting, is an excellent pedagogical tool; however, the rectified linear unit (ReLU) is, at present, the most common activation function in the hidden layers of NN. Uses for the other activation functions are provided in the text.

in situ feature matches a defect-indicative feature from the dictionary, then it is likely a defect has formed during manufacturing.

Neural networks (NNs) are particularly well-suited to handle features extracted from images, or simply the images themselves. There are many references that describe neural networks in detail, such as the work of Hastie et al. [194], and an increasing number that address the specific challenges associated with neural networks in materials science [195]. There are several properties of NNs that are worth repeating here, however. Each layer in a NN is connected to the next layer through an affine (linear) transformation. This step stretches, scales, and skews the input vector.

$$\mathbf{z}^{(i+1)} = \boldsymbol{\theta}^T \boldsymbol{i}\mathbf{x}^{(i)} \tag{24}$$

where $\mathbf{z}^{(i+1)}$ is the input into the $(i+1)$ layer and $\mathbf{x}^{(i)}$ is the output from the previous, i^{th} layer. Then, an activation function, such as those summarized in Fig. 13, introduces a non-linearity that warps/distorts the vector input to that layer.

$$\mathbf{x}^{(i+1)} = f\left(\mathbf{z}^{(i+1)}\right) \tag{25}$$

The model parameters $\boldsymbol{\theta}_i^T$ are regression weights that associate outputs from each layer $\mathbf{x}^{(i)}$ to subsequent layers $\mathbf{z}^{(i+1)}$. By increasing the depth of the NN, that is, adding additional layers, and the width (number of nodes) of those layers, a NN can be used to approximate any function, making them powerful regression and classification tools [196]. However, the general sparsity of materials data coupled to the complexity of process–structure–process relationship requires an understanding of the tradeoffs and requirements of using NNs in materials science, and in AM more specifically. Beyond the basics of model architecture, overfitting and the bias–variance tradeoff that is part of any machine learning model, a basic understanding of the role of activation functions can help to develop an intuition for the use of NN in materials and manufacturing.

An early use of NNs was in classification. The perceptron, logistic sigmoid (or simply, sigmoid), and hyperbolic tangent are all activation functions that choose between two options (0 or 1, or in the case of tanh, −1 or 1). While a binary option may seem overly limiting, even multinomial classification can be broken down into a sequence of such binary classifications: *A* or not *A*; and if not *A*, then *B* or not *B*; and if not *B*, *C* or not *C*; etc. However, such a serial solution will require more layers and, with more layers, longer training on larger datasets to fit all model parameters.

Visual examples of these activation functions can be seen in Fig. 13. While each behaves differently, particularly across the negative domain ($x < 0$), the simplicity and robustness of the ReLU have made it the most commonly used activation function for hidden layers in regression neural networks.

In the case of a multinomial classification problem, a more simple network may be possible by using one-hot encoding. A one-hot encoding vector is defined for *N* exclusive options: one element in the *N*-element vector is 1, all other values are 0. Rather than using multiple layers to construct the binomial ladder required to simulate a multinomial decision, the softmax activation function selects one-from-many in a single layer. Since each value in the input vector appears in the softmax exponent, even small differences in the magnitude of z result in large differences in the output of this activation function; therefore one option, represented by one node or neuron in the layer, is approximately 1 and all others are nearly 0. Simplification of the network architecture by choosing activation functions that more closely resemble the nature of the problem emphasizes the importance of domain-specific knowledge in developing appropriate NN architectures.

Combining the concepts of neural networks and image processing filters, convolutional neural networks (CNNs) not only learn how to correlate features to results, they are designed to also identify the filters that extract those features. These networks require large numbers of parameters, in the tens to hundreds of millions, that introduces an

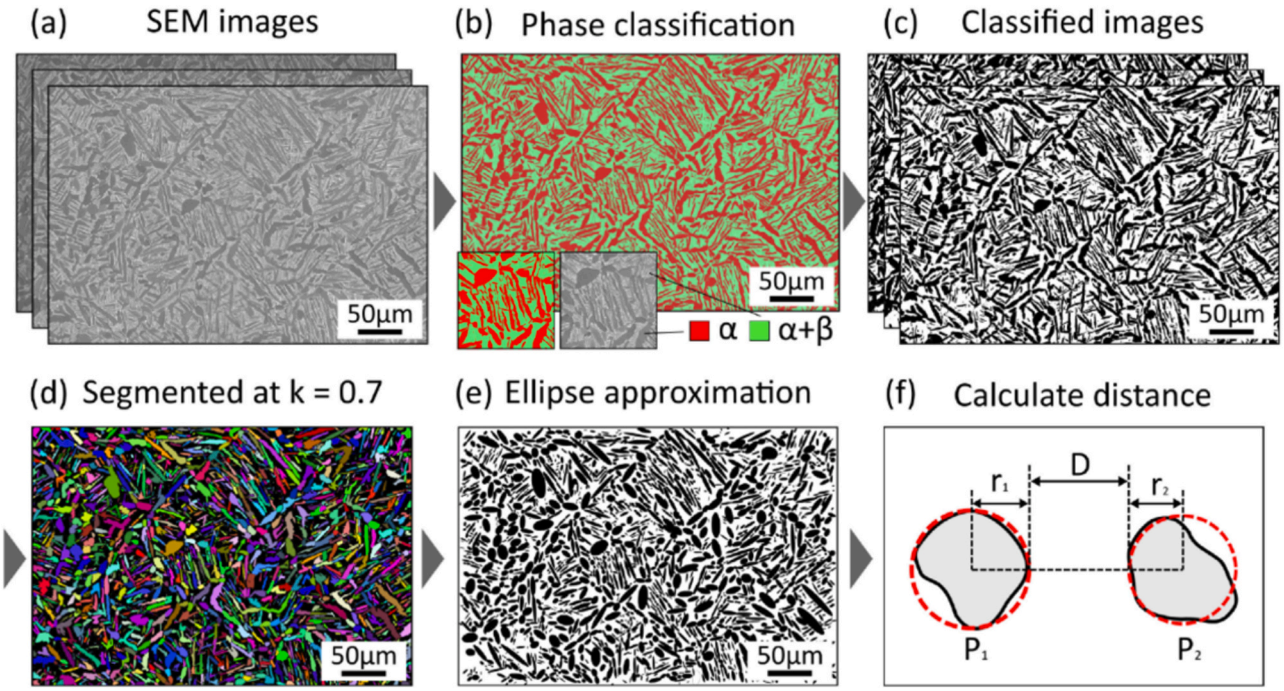


Fig. 14. The image segmentation approach implemented by Miyazaki et al. to automatically segment, classify, and characterize SEM images of Ti-6Al-4V microstructures. (a) First, the SEM images are obtained. (b) A random forest algorithm is used to classify regions of the image and (c) build a database of classified images. (d) Image segmentation proceeds to separate out the α and β phases. (e) An ellipse approximation is overlaid on the segmented image to characterize grain morphology and size. (f) The nearest neighbor distance can be calculated from the ellipse locations to provide a measure of grain distribution. Microstructures can be very complex for additively manufactured alloys and performing this characterization by hand becomes burdensome. Image recognition algorithms can automate the process and significantly speed up characterization, development, and qualification times.
Image reproduced from Miyazaki et al. [210].

insurmountable training burden due to the sparsity of materials data. However, CNNs trained on natural images have demonstrated a remarkable similarity in their initial layers [197]. These first few layers identify basic shapes, edges, and colors that are common to many image types; a phenomenon that many groups have exploited to overcome the limitation of data sparsity through transfer learning [124], including specific work in the field of additive manufacturing. Yuan et al. [198] were able to successfully monitor melt track width, standard deviation, and continuity of tracks in situ during laser powder bed manufacturing. Scime and Beuth trained a convolutional neural network to identify six different types of defect that are typical of laser powder bed fusion, with reasonable prediction accuracy [199]. Li et al. used a type of neural network method called *deep learning* to classify AM parts using microstructural images [200]. Kwon et al. classified melt pool morphologies using a neural network [201]. These studies represent only a few possible uses of CNN for in situ process monitoring of AM.

Scime and Beuth modified a well-known convolutional neural network architecture – known as AlexNet [202] – to perform classification of powder spreading errors that occur in laser powder bed fusion [203]. The study presented by Scime and Beuth go in-depth on the architecture of their CNN and directly explain how the training of filters applies in the context of AM images.

3.3.2. Featurization of qualitative image data

The same processing algorithms that are used for featurization and modeling of in situ signals can also be applied to automate part of the scientific process of studying additive manufacturing. Specifically, computer vision can be used to automate classification of microstructures during parametric analysis.

Parametric analysis in additive manufacturing requires the characterization and measurement of material properties that result from a specific coordinate in the design space. Often, material properties like mechanical strength, surface roughness, microstructure, or defect density have to be measured, analyzed, and quantified or classified as part of the parametric analysis process. This experimental process can be tedious. More often than not, images are relied upon heavily in classifying material properties, especially microstructures. Fortunately, machine learning algorithms can be applied to automate the analysis of images during the parametric analysis process.

It is worthwhile to mention up front that these algorithms have been *tested* on microstructure and, in some cases, additive-specific images. There are few algorithms that can process AM microstructure data "out-of-the-box." Rather, these algorithms will need to be tailored in order to quantify AM images specifically. However, the algorithms discussed here have been proven on non-AM microstructure datasets, thus they should be extensible to AM datasets. The computer vision approaches that work for microstructure data are often the same approaches that will be discussed again later for in situ monitoring and feedback.

One AM-related application of image characterization is measuring particle size distributions in AM powder feedstock. DeCost and Holm used SIFT with a dictionary classifier to measure the particle size distribution for a dataset of synthetic powder particles [204]. Particle size distribution plays in several steps across the additive process including energy absorption and part metrology [64,205,206]. DeCost created datasets with six different particle size distributions. Image features were identified and classified using *k*-means clustering on the features found by SIFT. Then, a classification algorithm known as a support vector machine (SVM) was trained to classify image features into particle sizes. DeCost was able to achieve 89% overall classification accuracy in measuring particle size distribution this way. DeCost later improved upon this powder classification method and were able to achieve higher classification accuracies for real powder images [207]. Machine learning algorithms have also been trained for the automatic classification/identification of EBSD texture maps [208,209] (Fig. 14).

Strides have been made in automatically identifying and quantifying information from metallographs [49,124,211,212]. A good portion of quality control in materials science as a whole, not just AM, involves classifying materials based on metallographs or micrographs of microstructure. Work is being done across materials science to apply machine learning based computer vision to classifying and quantifying information in these microstructural images. Doing so will speed up the process of materials characterization and qualification, while also providing methods of quantifying information that otherwise would have stayed in a qualitative form. Examples include classification of grain structures, measurements of grain size, pore size calculations, and more.

An additive-specific image segmentation algorithm was used by Miyazaki et al. [210]. Five image filters were convolved with microstructure images of selective laser melted Ti-6Al-4V. The features identified by these filters were used in a random forest algorithm to segment the image into regions of α phase grains and β phase grains. The algorithm was able to automatically calculate area fraction of primary and secondary α phases that form during cooling. It was also able to calculate the nearest-neighbor distance between grains. Nearest neighbor distance of grains is indicative of grain characteristics like size, morphology, and distribution.

Chowdhury et al. took a more expansive approach to performing feature identification in microstructures. In particular, they were looking to classify microstructures as either dendritic or non-dendritic. Chowdhury employed 8 different feature identification methods for a dataset of images. Classification was performed using an ensemble of ML techniques including support vector machines (SVM), Naïve Bayes, nearest neighbor, and a committee of the three previous classification methods [213]. Chowdhury's wide approach to image classification achieved classification accuracies above 90%.

Efforts are underway across materials science to implement computer vision for the automation of materials classification. Rather, the authors would like to refer the reader to the computer vision libraries listed in Table 4.

4. Learning from the past: moving towards database-driven design of additive technologies

The scientific approaches to studying additive manufacturing discussed herein – parametric analysis, computational modeling, in situ monitoring, and the like – produce data. The application of machine learning to these scientific approaches likewise produces data. All of this data comprises a subset of the AM design space. The integration of this data into multi parameter, multi physics, multi printer datasets increases both the size of the design space that can be explored as well as the depth/accuracy at which certain regions of the design space can be modeled. Making AM process-structure and process-property data open and accessible to the scientific public accelerates the rate at which data-driven approaches can help to advance AM research and engineering. This potential is evident in examining some more mature examples of the use of data-driven approaches in materials science and engineering, which we proceed to briefly review in this section to motivate the development of data-driven approaches for AM.

Databases of process-structure and process-property relationships are not a new concept in materials science. Databases like the Linus Pauling Files or International Crystal Structure Database have been widely used for materials design. Domain-specific databases are also being generated from high throughput experimental and computational investigations that have occurred over the past thirty years. Experimental high throughput investigations have also been used in materials science for many decades [214]. Common deposition techniques (sputter, plasma, vapor, etc.) have enough degrees of freedom to allow for continuous compositional variation within a single sample, which allows for continuous mapping of composition-structure-property relationships [35–37]. These combinatorial synthesis methods present analogous design space challenges to AM: the number of possible input combinations obscures many of the important underlying process-structure phenomenon. It has long been established that synthesizing

and characterizing large combinatorial catalogues of samples can lead to the discovery of materials with optimized properties faster than a theory-driven approach by itself [215,216]. High throughput deposition studies with chemical vapor deposition, metallorganic chemical vapor deposition, physical vapor deposition, and atomic layer deposition, among other techniques are commonplace for the manufacturing of sensors, batteries, photovoltaics, electronics, shape memory alloys, and the like [87,217–222]. Furthermore, the parameters of interest in these studies can sometimes be quickly catalogued using high throughput characterization techniques like laboratory X-ray diffraction and electron probe microanalysis [223,224]. These combinatorial studies culminate in large libraries of material properties listed as a function of composition. As far back as the 1990s, data-driven algorithms were being applied to search and discover using these large libraries of composition-property data. Evolutionary and genetic algorithms were trained on composition to predict stable crystal structure and material properties [85,89,92,225,226]. Even neural networks, which did not have the widespread use then that they have now, were being applied for the prediction of crystal structures based on composition [227].

Modeling challenges in materials science have also been tackled using large databases with machine learning. Packages such as the Vienna Ab initio Simulation Package (VASP) have been employed for high throughput searches of stable material systems with a wide range of properties. The stability and maturity of these packages have enabled the reliable automated calculation of new stoichiometries and new phases [228] and enabled the automated and semi-automated search for new functional materials [229]. As these methods have improved, computational high throughput investigations continue to increasingly match and provide complementary information to experimental measurements [230]. High throughput density functional theory (DFT) studies generate quite a bit of data and are therefore well equipped for machine learning and database-driven design. The application of high throughput DFT is widespread for design of materials with all sorts of properties including high temperature superconductors [231], lithium ion batteries [83,232,233], molecule design [20,234], cathode materials [235], piezoelectrics [86], ferroelectrics [236], corrosion resistant films [237], and thermoelectrics [238,239]. Each of these studies, like parametric studies in additive, vary a set number of model input parameters and measure a material property as the dependent response.

Yet many of the same modeling obstacles exist in DFT as in AM, such as a lack of transferability between models and the computational expense of large material systems. The design space problem exists here as well – there are so many possible compositional combinations that knowing where to look is difficult. Machine learning was proposed as a solution for obstacles in high throughput DFT as early as 2005 [240]. Large unit cells whose properties cannot be directly calculated using DFT are often approximated using machine learning approaches like neural networks [241], genetic algorithms [93], and principal component analysis [162]. Studies applying machine learning to databases of computational information have gone beyond tackling computational problems. In some cases, the studies have revealed previously unobserved or uncharacterized relationships between crystal structure information and materials properties [242].

In other efforts to reduce the time to design and deploy new materials, programs like the Materials Project incorporate data taken from a wide range of experimental and computational methods into an open-source, accessible database. The Materials Project also features electronic, structural, and thermodynamic calculations of different materials as well as an automated workflow for doing DFT computations of material systems [75,243]. Other databases of materials information include AFLOWLib [74,244], the Harvard Clean Energy Project [245], Japan's National Institute of Material Science [246], and the Open Quantum Materials Database [247]. Some pipelines for high-throughput computation and analysis have included consideration of publication timelines in their processes [248]. These databases offer a multitude of benefits to materials researchers. First and foremost, publicly accessible databases offer an infrastructure for the free flow of experimental and computational results. Synergy between research groups becomes easier as data is shared more freely. Furthermore, many of these online databases also provide tools for performing material design. The Materials Project offers a design interface, whereby users can specify a set of material properties and are provided with a list of likely candidate materials. Other projects, like AFLOW, allow for fast high-throughput DFT calculations of a wide range of material systems.

The generation of databases that are accessible to the scientific public is a primary step on the roadmap of the Materials Genome Initiative [249]. Much of the development of materials databases have focused on computationally-derived materials information. Infrastructure and standards need to be developed that allow for sharing of experimental data that is understandable and usable by many researchers. Data journals are becoming more common for sharing datasets from scientific investigations and are making strides in standardizing data-sharing infrastructure [250], along with the publication of datasets themselves for public use [251,252]. By examining the development of image processing databases outside of materials science, it is evident that the collection and distribution of image databases have enabled rapid developments in the field of computer vision. Many of the more common objectives with computer vision – autonomous navigation, face recognition, object recognition, image segementation – have databases that are catalogued in online repositories like CVonline [253] and VisionScience [254]. Learning from these other fields, open sharing of AM microstructure image databases will aid in the development of segmentation and identification algorithms that are suited for materials, and more specifically AM-specific problems.

Having open, accessible databases improves the rate at which machine learning can be applied to design for additive manfuacturing. Machine learning as a tool driving materials design was proposed some time ago. Review articles have explored the many and varied uses of machine learning across materials science, with many of the applications finding great success [19,255,256]. A review article on best practices for machine learning in materials science can be found in the work of Wagner et al. [18]. Open sharing of databases also tackles a problem in ICME approaches to AM; that is, the integration of multiple data sources. AM incorporates relevant physics over many different time and length scales, to the extent that a single research group is unlikely to have access to all pertinent information. Open sharing of data sets, whether it is computationally derived, experimental, or images, allows research groups to incorporate multiple physics simultaneously. Furthermore, it will accelerate the rate at which AM materials research is performed as higher fidelity machine learning models can be built with more and diverse datasets.

Additive manufacturing should move toward the same types of infrastructure for open data sharing. The combinatorial problems in additive are widespread and cover many, many length scales. Large institutions may have the resources to link time- and length-scales in additive manufacturing. Smaller research groups are often limited to studying a single process phenomenon and do not necessarily have means to integrate their knowledge into other additive manufacturing studies. The generation of additive databases allows for a democratization of research and an acceleration of the pace at which additive manufacturing advances are made.

5. Conclusions

Materials informatics has demonstrated great success as a tool that can accelerate and reduce cost for discovery, design and optimization of many material systems. Metals additive manufacturing is primed to benefit from the same algorithms and statistical models. Many of the major obstacles that lie ahead in additive manufacturing – fully integrated ICME modeling approaches, data-driven design, feedback and control using in situ process monitoring sensors – can be attained by incorporating machine learning. However, machine learning itself is not

the end-all-be-all solution to developing AM technologies. There are many obstacles in the application of machine learning itself that will need to be addressed along the way. ML is a complementary tool to physics-based modeling and experiments. Just like transmission electron microscopy does not solve every problem by itself, neither will machine learning. Instead, it should be understood where and when ML is a desirable technique, and which class/type of ML algorithms is right for the problem. Since the goal of this review article is to introduce AM scientists and engineers to the concepts of ML and the selection and evaluation of ML algorithms for solving problems in AM, in conclusion, it is worthwhile to summarize the major AM challenges that can be solved using machine learning, as well as identify the major obstacles to implementation.

5.1. Key application areas for machine learning in additive manufacturing

- Coupled Physics-Statistics Models: The original goal of materials informatics, dating back to high throughput thin-film studies in the 1990 s, was to model material process-structure-property relationships that were highly complicated and lacked a single governing physical theory [214]. Additive manufacturing is the embodiment of a complicated physical system, where governing equations across optics, fluid mechanics, solid mechanics, thermodynamics, and kinetics have to be incorporated into one model. Machine learning can build computationally accessible surrogate models of more complicated physical systems that are useful for engineering and design.
- Materials Design: Materials design through machine learning has already been applied in a wide range of fields cited here, including thermoelectrics, photovoltaics, semiconductors, Heusler compounds, and many, many more. Design in these fields typically focuses around combinatorial studies of compositions, crystal structures, and a material response. Materials are manufactured through a wide variety of techniques but optimization is rarely applied to the manufacturing method itself, just the materials used in manufacturing. In additive, not only does the material system need to be tailored but the conditions of manufacturing also need optimization. Materials properties to consider range from composition and atomic properties to phase kinetics. Manufacturing optimization includes the energy density used, deposition rate, feedstock supply mechanism, and more. Machine learning can integrate optimization across these separate design considerations. Process optimization is likely to include in situ control.
- Automated Process Control: There are many variables to monitor and keep track of in the additive processes. There are equally many sensors and measurement techniques to monitor the process. Advancements in signal processing and computer vision must be taken advantage of to build incorporating process control models. Intelligent feedback and control for additive can simultaneously integrate and understand multiple signal types and optimize on multiple objective functions simultaneously. Taking full advantage of the promises of AM – topologically optimized geometries, functionally graded materials, minimized design-to-fly time – will require tight control over the manufacturing process.

5.2. Further developments are needed in both additive manufacturing and machine learning

- Data Sharing Infrastructure: Programs like the Materials Project, AFLOW, and OQMD have accelerated the rate at which materials design can occur, as well as the rate at which scientific data is shared. The democratization of data has allowed many different research teams to search through the materials design space in search of new materials, to great success. The same type of democratization is possible in additive if infrastructure exists for sharing of AM data. However, standardization of AM data types should be addressed before data can be shared in a useful, meaningful way.
- Curation of Data and AM Standards: Success in applying data-driven approaches is tied tightly to the quality of data being used. Even data that has been collected with the highest care and precision can be detrimental to a model if it is labeled incorrectly or inconsistently. Work is proceeding in standards development for additive manufacturing [257]. However, additive manufacturing technology development has sometimes proceeded faster than standardization. Care needs to be taken in developing AM standards that are consistent across manufacturing devices and can also account for developments in the broader technology.
- Experimental Measurement and Sensor Development: While in situ measurement devices are widespread, the time and length scales of additive manufacturing can push the limits of current high-end sensors. Imaging methods that can resolve the fast, dynamic, microscale melt pools of additive would allow for a huge leap in process monitoring and control. Equally important is developing methods of determining temperature history throughout the duration of builds. Both of these technologies are crucial for fine control over the additive process.
- Physics-Informed-Data-Driven Models: Additive manufacturing has developed amazingly over the past few decades thanks to traditional scientific and engineering approaches in many different fields. Modeling AM using classical thermal, mechanical and kinetic models has shown success in advancing and engineering the technology. This review is suggesting that machine learning be used as a complementary tool to these traditional approaches. It would be unwise to completely ignore physical theories that have shown applicability in AM. Rather, machine learning algorithms should be built around currently existing models. There are equally rich mathematical frameworks in both materials science and machine learning that are currently being utilized separately. The physics of AM at all length scales – solidification, phase kinetics, heat transfer, solid mechanics, etc. – should be used as first principles for building physics-informed statistical models. Many in the materials science community have considered how to use domain knowledge to build better informatics models [18,89,225,258]. The same should be applied to additive manufacturing.

Additive manufacturing stands to significantly expand humanity's ability to manufacture high performance, multifunctional, and highly customized engineering parts. At present, non trivial challenges in understanding the PSPP relationships stand in the way of achieving the full potential of AM. The development, integration, and application of statistical analysis, machine learning, and data-driven approaches into the additive manufacturing R&D ecosystem will tackle many of the problems currently facing the technology's advancement. Additive manufacturing is positioned to provide foundational case studies for the adoption of machine learning into physics-based integrated computational materials engineering, largely due to the simultaneous peak in funding for both additive manufacturing and data-driven materials research across the globe. The success of machine learning applications in metals additive manufacturing are poised to provide the foundation for a new paradigm in integrated computational materials engineering as a whole.

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: One of the authors of this article is an Associate Editor for the Additive Manufacturing Journal. To avoid potential conflicts of interest, the responsibility for the editorial and peer-review process of this article lies with a different Editor. Furthermore, the authors of this article were removed from the peer review process and have not had nor will they

have access to confidential information related to the editorial process of this article.

Acknowledgements

We acknowledge financial support from the U.S. Department of Defense, Office of Economic Adjustment, #CTGG1 2016-2166 and #ST1605-19-03. N.S. Johnson acknowledges the financial support of the ADAPT Additive Manufacturing Graduate Fellowship established by Los Alamos National Laboratory, USA.

References

[1] J.H. Panchal, S.R. Kalidindi, D.L. McDowell, Comput. Aided Des. 45 (2013) 4.
[2] G. Olson, Science 288 (2000) 993.
[3] A. Spierings, T. Starr, K. Wegener, Rapid Prototyp. J. 19 (2013) 88.
[4] E. Wycisk, A. Solbach, S. Siddique, D. Herzog, F. Walther, C. Emmelmann, Phys. Procedia 56 (2014) 371.
[5] P. Edwards, M. Ramulu, Mater. Sci. Eng. A 598 (2014) 327.
[6] M. Pröbstle, S. Neumeier, J. Hopfenmüller, L. Freun, T. Niendorf, D. Schwarze, M. Göken, Mater. Sci. Eng. A 674 (2016) 299.
[7] T.G. Gallmeyer, S. Moorthy, B.B. Kappes, M.J. Mills, B. Amin-Ahmadi, A. P. Stebner, Addit. Manuf. 31 (2019).
[8] J.H. Martin, B.D. Yahata, J.M. Hundley, J.A. Mayer, T.A. Schaedler, T.M. Pollock, Nat. Lett. 549 (2017) 365.
[9] Y.M. Wang, T. Voisin, J.T. McKeown, J. Ye, N.P. Calta, Z. Li, Z. Zeng, Y. Zhang, W. Chen, T.T. Roehling, R.T. Ott, M.K. Santala, P.J. Depond, M.J. Matthews, A. V. Hamza, T. Zhu, Nat. Mater. 50 (2017) 1.
[10] L. Liu, Q. Ding, Y. Zhong, J. Zou, J. Wu, Y. Chiu, J. Li, Z. Zhang, Q. Yu, Z. Shen, Mater. Today 21 (2017) 354.
[11] Z. Zhu, Q. Nguyen, F. Ng, X. An, X. Liao, P. Liaw, S. Nai, J. Wei, Scr. Mater. 20 (2018).
[12] T.M. Pollock, Nat. Mater. 15 (2016) 809.
[13] G. Olson, C. Keuhmann, Scr. Mater. 70 (2014) 25.
[14] United States National Science and Technology Council, Office of Science and Technology Policy, Materials genome initiative for global competitiveness, Technical Report, 2011.
[15] I. Bose, R.K. Mahapatra, Inf. Manag. 39 (2001) 211.
[16] R. Gómez-Bombarelli, J.N. Wei, D. Duvenaud, J.M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T.D. Hirzel, R. P. Adams, A. Aspuru-Guzik, ACS Cent. Sci. 4 (2018) 268.
[17] P. Brusilovsky, A. Kobsa, W. Nejdl, (Eds.), The Adaptive Web, Springer-Verlag, New York, 2007.
[18] N. Wagner, J. Rondinelli, Front. Mater. 3 (2016).
[19] R. Ramprasad, R. Batra, G. Pilania, A. Mannodi-Kanakkithodi, C. Kim, Nat. Comput. Mater. 3 (2017).
[20] K.T. Butler, D.W. Davies, H. Cartwright, O. Isayev, A. Walsh, Nat. Rev. 559 (2018).
[21] K. Rajan, Mater. Today 8 (2005) 38.
[22] A. Agrawal, A. Choudhary, APL Mater. 4 (2016).
[23] P. Ball, MRS Bull. 44 (2019).
[24] C. Druzgalski, A. Ashby, G. Guss, W. King, T. Roehling, M. Matthews, Addit. Manuf. (2020).
[25] J. Quinlan, Mach. Learn. 1 (1986) 81.
[26] R. Bro, A.K. Smilde, Anal. Methods Tutor. Rev. 6 (2014).
[27] J.B. Tenenbaum, V. de Silva, J.C. Langford, Science 290 (2000).
[28] S.T. Roweis, L.K. Saul, Science 290 (2000).
[29] J.J. Grefenstette, IEEE Trans. Syst. Man Cybern. 16 (1986).
[30] R.J. Murdock, S.K. Kauwe, A.Y.-T. Wang, T.D. Sparks, ChemRxiv (2020).
[31] H. Li, W. Li, X. Pan, J. Huang, T. Gao, L. Hu, H. Li, Y. Lu, J. Chemom. 32 (2017).
[32] B.D. Ripley, Spatial Statistics, first ed.,, John Wiley & Sons,, 1981.
[33] O. Scabenberger, C. a. Gotway, Statistical Methods for Spatial Data Analysis, Chapman and Hall, 2004.
[34] E.J. Candés, M.B. Wakin, IEEE Signal Process. Mag. 25 (2008) 21.
[35] C.J. Long, J. Hattrick-Simpers, M. Murakami, R.C. Srivastava, I. Takeuchi, V. L. Karen, X. Li, Rev. Sci. Instrum. 78 (2007), https://doi.org/10.1063/1.2755487.
[36] C.J. Long, D. Bunker, X. Li, V.L. Karen, I. Takeuchi, Rev. Sci. Instrum. 80 (2009), https://doi.org/10.1063/1.3216809.
[37] A.G. Kusne, T. Gao, A. Mehta, L. Ke, M.C. Nguyen, K.-M. Ho, V. Antropov, C.-Z. Wang, M.J. Kramer, C. Long, I. Takeuchi, Sci. Rep. 4 (2015) 6367.
[38] R. Szeliski, in: D. Gries, F.B. Schneider (Eds.), Computer Vision, first ed., Springer-Verlag London Limited, London, 2011.
[39] MathWorks®, Imfilter boundary padding options. https://www.mathworks.com/help/images/imfilter-boundary-padding-options.html.
[40] D.T.F.S.R. Niezgoda, S.R. Kalidindi, Acta Mater. 56 (2008) 942.
[41] W. Contributors, Kernel method-Wikipedia, the free encyclopedia, 2019.
[42] J. Beuth, N. Klingbeil, J. Mater.: Laser Process. (2001).
[43] W. Navidi, Statistics for Engineers, third ed., McGraw Hill,, New York, NY, 2006.
[44] C.P. DATAQUEST Data Science Tutorials, Tutorial: Understanding regression error metrics in python, 2019. https://www.dataquest.io/blog/understanding-regression-error-metrics/.
[45] S. Shan, G.G. Wang, Struct. Multidiscip. Optim. 41 (2010) 219.
[46] A. Botchkarev, arxiv PrePrint.
[47] A. MishraMetrics to evaluate your machine learning algorithm, 2018. https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234.
[48] S. Minaee,20 popular machine learning metrics. part 1: classification & regression evaluation metrics, 2019. https://towardsdatascience.com/20-popular-machine-learning-metrics-part-1-classification-regression-evaluation-metrics-1ca3e282a2ce.
[49] B.L. Decost, E.A. Holm, Comput. Mater. Sci. 110 (2015) 126.
[50] K. Kryzk, Linear regression (part 2): Cost function, 2018. https://towardsdatascience.com/coding-deep-learning-for-beginners-linear-regression-part-2-cost-function-49545303d29f.
[51] G. Varoquaux,Mathematical optimization: finding minima of functions. https://scipy-lectures.org/advanced/mathematical_optimization/index.html.
[52] C. McDonald, Machine learning fundamentals (i): cost functions and gradient descent, 2017. https://towardsdatascience.com/machine-learning-fundamentals-via-linear-regression-41a5d11f5220.
[53] E. Brochu, V.M. Cora, N. de Freitas, arxiv:1012.2599, 2010.
[54] B. Meredig, E. Antono, C. Church, M. Hutchinson, J. Ling, S. Paradiso, B. Blaiszik, I. Foster, B. Gibbons, J. Hattrick-Simpers, A. Mehta, L. Ward, Mol. Syst. Des. Eng. 3 (2018) 819.
[55] Classification: ROC curve and AUC, 2020. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc.
[56] S. Liu, B. Kappes, A. Stebner, X. Zhang, Addit. Manuf. Rev. 1 (2020).
[57] https://scikit-learn.org/stable/.
[58] https://www.tensorflow.org.
[59] https://keras.io.
[60] https://opencv.org.
[61] (a) https://www.mathworks.com/products/statistics.html.
[62] (b) https://www.mathworks.com/products/computer-vision.html.
[63] https://mlr3.mlr-org.com.
[64] C. Boley, S. Mitchell, A. Rubenchik, S. Qu, Appl. Opt. 55 (2016) 6496.
[65] J. Trapp, A.M. Rubenchik, G. Guss, M.J. Matthews, Appl. Mater. Today 9 (2017) 341.
[66] G. Bi, C.N. Sun, A. Gasser, J. Mater. Process. Technol. 213 (2013) 463.
[67] P. Collins, D. Brice, P. Samimi, I. Ghamarian, H. Fraser, Annu. Rev. Mater. Res. 46 (2016) 63.
[68] C.A. Brice, W.A. Tayon, J.A. Newman, M.V. Kral, C. Bishop, A. Sokolova, Mater. Charact. (2018).
[69] B. Wysocki, P. Maj, R. Sitek, J. Buhagiar, K.J. Kurzydłl, W. Święszkowski, Appl. Sci. 7 (2017).
[70] J. Slotwinski, E. Garboczi, P. Stutzman, C. Ferraris, S. Watson, M. Peltz, J. Res. Natl. Inst. Stand. Technol. 119 (2014).
[71] A. Strondl, O. Lyckefeldt, H. Brodin, U. Ackelid, JOM 67 (2015).
[72] A. Wang, S. Song, Q. Huang, F. Tsung, IEEE Trans. Autom. Sci. Eng. 14 (2017) 968.
[73] P. Villars, N. Onodera, S. Iwata, J. Alloy. Compd. 279 (1998) 1.
[74] S. Curtarolo, W. Setyawan, S. Wang, J. Xue, K. Yang, R.H. Taylor, L.J. Nelson, G.L. W. Hart, S. Sanvito, M. Buongiorno-Nardelli, N. Mingo, O. Levy, Comput. Mater. Sci. 58 (2012) 227.
[75] A. Jain, S.P. Ong, G. Hautier, W. Chen, W.D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder, K.A. Persson, APL Mater. 1 (2013).
[76] J.S. Nuechterlein, J.J. Iten, Patent no.: Us 10, 507, 638 bw, Electronically, 2016.
[77] A. Fransceschetti, A. Zunger, Lett. Nat. 402 (1999).
[78] C.C. Fischer, K.J. Tibbetts, D. Morgan, G. Ceder, Nat. Mater. 5 (2006) 641.
[79] A.R. Oganov, C.W. Glass, J. Chem. Phys. 124 (2006), https://doi.org/10.1063/1.2210932.
[80] Y. Ikeda, Mater. Trans. 38 (1997) 771.
[81] A.M. Gopakumar, P.V. Balachandran, D. Xue, J.E. Gubernatis, T. Lookman, Nat. Sci. Rep. 8 (2018).
[82] D. Wu, Y. Tian, L. Zhang, Z. Wang, J. Sheng, W. Wang, K. Zhou, L. Liu, Materials 11 (2018).
[83] S. Kirklin, B. Meredig, C. Wolverton, Adv. Energy Mater. 3 (2013) 252.
[84] W. Setyawan, R.M. Gaume, S. Lam, R.S. Feigelson, S. Curtarolo, ACS Comb. Sci. 382 (2011).
[85] D. Wolf, O. Buyevskaya, M. Baerns, Appl. Catal. A: Gen. 200 (2000) 63.
[86] A. Roy, J.W. Bennett, K.M. Rabe, D. Vanderbilt, Phys. Rev. Lett. 109 (2012) 1, arxiv:1107.5078.
[87] G. Gilmer, H. Huang, C. Roland, Comput. Mater. Sci. 12 (1998) 354.
[88] H. Koinuma, I. Takeuchi, Nat. Mater. 3 (2004) 429.
[89] J. Morris, D. Deaven, K. Ho, Phys. Rev. B 53 (1996) R1740.
[90] K.-M. Ho, A.A. Shvartsburg, B. Pan, Z.-Y. Lu, C.-Z. Wang, J.G. Wacker, J.L. Fye, M. F. Jarrold, Nat. (Lond.) 392 (1998) 582.
[91] G.H. Jóhannesson, T. Bligaard, A.V. Ruban, H.L. Skriver, K.W. Jacobsen, J. K. Norskov, Phys. Rev. Lett. 88 (2002), 2555061.
[92] D.P. Stucke, V.H. Crespi, Nano Lett. 3 (2003) 1183.
[93] G.L.W. Hart, V. Blum, M.J. Walorski, A. Zunger, Nat. Mater. 4 (2005) 391.
[94] W. Xu, P.R.-D. delCastillo, S. van der Zwaag, Philos. Mag. 88 (2008) 1825.
[95] W. Tan, N.S. Bailey, Y.C. Shin, Comput. Mater. Sci. 50 (2011) 2573.
[96] J. Andersson, T. Helander, L. Höglund, P. Shi, B. Sundman, Calphad 26 (2002) 273.
[97] S. Li, U.R. Kattner, C.E. Campbell, Integr. Mater. Manuf. Innov. 6 (2017) 229.
[98] S. Liu, B.B. Kappes, B. Amin-ahmadi, O. Benafan, X. Zhang, A.P. Stebner, arxiv PrePrint, 2020.
[99] A. Kulkarni, K. Krishnamurthy, S. Deshmukh, R. Mishra, Mater. Sci. Eng. A 372 (2004) 213.

[100] S. Dudiy, A. Zunger, Phys. Rev. Lett. 97 (2006).
[101] S.M. Anijdan, A. Bahrami, H.M. Hosseini, A. Shafyei, Mater. Des. 27 (2006) 605.
[102] G. Hautier, C.C. Fischer, A. Jain, T. Mueller, G. Ceder, Chem. Mater. 22 (2010) 3762.
[103] N. Chakraborti, Int. Mater. Rev. 49 (2004) 246.
[104] Z.-K. Liu, L.-Q. Chen, K. Rajan, J. Mater.: Integr. Comput. Mater. 42 (2006).
[105] G.A. Landrum, H. Genin, J. Solid State Chem. 176 (2003) 587.
[106] A. Oliynyk, E. Antono, T. Sparks, L. Ghadbeigi, M. Gaultois, B. Meredig, A. Mar, Chem. Mater. 28 (2016) 7324.
[107] J. Antony, Design of Experiments for Engineers and Scientists, second ed., Elsevier Ltd, 2014.
[108] A. Antonysamy, J. Meyer, P. Prangell, Mater. Charact. 84 (2013) 153.
[109] G. Strano, L. Hao, R.M. Everson, K.E. Evans, J. Mater. Process. Technol. 21 (2013) 589.
[110] S. Bontha, N.W. Klingbeil, P.A. Kobryn, H.L. Fraser, Mater. Sci. Eng. A 513–514 (2009) 311.
[111] P. Nie, O. Ojo, Z. Li, Acta Mater. 77 (2014) 85.
[112] Y. Chen, F. Lu, K. Zhang, P. Nie, S.R.E. Hosseini, K. Feng, Z. Li, J. Alloy. Compd. 670 (2016) 312.
[113] B. Baufeld, E. Brandl, O. Van Der Biest, J. Mater. Process. Technol. 211 (2011) 1146.
[114] S. Bontha, N.W. Klingbeil, P.A. Kobryn, H.L. Fraser, J. Mater. Process. Technol. 178 (2006) 135.
[115] Y. Li, D. Gu, Mater. Des. 63 (2014) 856.
[116] J. Cherry, H.M. Davies, S. Mehmood, N.P. Lavery, S.G.R. Brown, J. Sienz, Int. J. Adv. Manuf. Technol. 76 (2015) 869.
[117] Q. Jia, D. Gu, J. Alloy. Compd. 585 (2014) 713.
[118] J. Delgado, J. Ciurana, C.A. Rodriguez, Int. J. Adv. Manuf. Technol. 60 (2012) 601.
[119] A.M. Khorasani, I. Gibson, U.S. Awan, A. Ghaderi, Addit. Manuf. (2018).
[120] A.S. Wu, D.W. Brown, M. Kumar, G.F. Gallegos, W.E. King, Metall. Mater. Trans. A: Phys. Metall. Mater. Sci. 45 (2014) 6260.
[121] E.R. Denlinger, J.C. Heigel, P. Michaleris, T. Palmer, J. Mater. Process. Technol. 215 (2015) 123.
[122] C.E. Shannon, Bell Syst. Tech. J. 27 (1948) 379, arxiv:9411012 [chao-dyn].
[123] P. Wigley, P. Everitt, A. van den Hengel, J. Bastian, M. Sooriyabandara, G. McDonald, K. Hardman, C. Quinlivan, P. Manju, C. Kuhn, I. Petersen, A. Luiten, J. Hope, N. Robins, M. Hush, Sci. Rep. 6 (2016).
[124] J. Ling, M. Hutchinson, E. Antono, B. DeCost, E.A. Holm, B. Meredig, Mater. Discov. 10 (2017) 19.
[125] J. Ling, M. Hutchinson, E. Antono, S. Paradiso, B. Meredig, Integr. Mater. Manuf. Innov. 6 (2017) 207, arxiv:1704.07423.
[126] T. Ueno, T.D. Rhone, Z. Hou, T. Mizoguchi, K. Tsuda, Mater. Discov. 4 (2016) 18.
[127] M. Gaultois, A. Oliynyk, A. Mar, T. Sparks, G. Mulholland, B. Meredig, APL Mater. 4 (2016), 053213.
[128] J. Carrete, W. Li, N. Mingo, S. Wang, S. Curtarolo, Phys. Rev. X 4 (2014), 011019.
[129] C. Kim, G. Pilania, R. Ramprasad, Chem. Mater. 28 (2016) 1304.
[130] T. Zegard, G.H. Paulino, Struct. Multidiscip. Optim. 53 (2016) 175.
[131] O. Sigmund, K. Maute, Struct. Multidiscip. Optim. 48 (2013) 1031.
[132] A.T. Gaynor, J.K. Guest, Struct. Multidiscip. Optim. 54 (2016) 1157.
[133] M. Langelaar, Addit. Manuf. 12 (2016) 60.
[134] M. Langelaar, Struct. Multidiscip. Optim. 55 (2017) 871.
[135] J. Liu, A.T. Gaynor, S. Chen, Z. Kang, K. Suresh, A. Takezawa, L. Li, J. Kato, J. Tang, C.C. Wang, L. Cheng, X. Liang, A.C. To, Struct. Multidiscip. Optim. 57 (2018) 2457.
[136] X. Huang, C. Ye, S. Wu, K. Guo, J. Mo, Int. J. Adv. Manuf. Technol. 42 (2009).
[137] J. Vanek, G. JAG, B. Benes, Comput. Graph. Forum 33 (2014).
[138] J. Dumas, J. Herget, S. Lefebrvre, ACM Trans. Graph. 33 (2014).
[139] S. Banga, H. Gehani, S. Bhilare, S. Patel, L. Kara, CoRR, abs/1808.07440, arXiv: 1808.07440, 2018.
[140] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, A.A. Bharath, IEEE Signal Process. Mag. 35 (2018) 53.
[141] S. Oh, Y. Jung, S. Kim, I. Lee, N. Kang, J. Mech. Des. 141 (2019), https://doi.org/10.1115/1.4044229.
[142] Y. Lecun, Y. Bengio, G. Hinton, Nature 521 (2015) 436.
[143] R. Cang, H. Yao, Y. Ren, Comput. -Aided Des. 109 (2019) 12.
[144] Y. Yu, T. Hur, J. Jung, I.G. Jang, Struct. Multidiscip. Optim. 59 (2019) 787.
[145] E. Andreassen, A. Clausen, M. Schevenels, B.S. Lazarov, O. Sigmund, Struct. Multidiscip. Optim. 43 (2011) 1.
[146] D.J. Lohan, E.M. Dede, J.T. Allison, 2016. 10.1007/s00158–016-1563–6.
[147] L. Zimmermann, T. Chen, K. Shea, Artificial Intelligence for Engineering Design, Analysis and Manufacturing 32, Cambridge University Press, 2018, p. 189.
[148] R. Martukanitz, P. Michaleris, T. Palmer, T. DebRoy, Z.-K. Liu, R. Otis, T.W. Heo, L.-Q. Chen, Addtive Manuf. 1 (2014) 52.
[149] E. Toyserkani, A. Khajepour, S. Corbin, Opt. Lasers Eng. 41 (2004) 849.
[150] S.A. Khairallah, A.T. Anderson, A. Rubenchik, W.E. King, Acta Mater. 108 (2016) 36.
[151] V. Manvatkar, A. De, T. DebRoy, J. Appl. Phys. 116 (2014).
[152] D. Dai, D. Gu, Mater. Des. 55 (2014) 482.
[153] P. Michaleris, Finite Elem. Anal. Des. 86 (2014) 51.
[154] M. Gouge, P. Michaleris, E. Denlinger, J. Irwin, Thermo-Mechanical Modeling of Additive Manfuacturing, Chapter 2: The Finite Element Method for the Thermo-Mechanical Modeling of Additive Manufacturing Processes, Elsevier Inc., 2018.
[155] L.-Q. Chen, Annu. Rev. Mater. Res. 32 (2002) 113.
[156] X. Gong, K. Chou, J. Mater. 67 (2015) 1176.
[157] J. Kundin, L. Mushongera, H. Emmerich, Acta Mater. 95 (2015) 343.
[158] S. Sahoo, K. Chou, Addit. Manuf. 9 (2016) 14.
[159] M. Francois, A. Sun, W. King, N. Henson, D. Tourret, C. Bronkhorst, N. Carlson, C. Newman, T. Haut, J. Bakosi, J. Gibbs, V. Livescu, S. VanderWiel, A. Clarke, M. Schraad, T. Blacker, H. Lim, T. Rodgers, S. Owen, F. Abdeljawad, J. Madison, A. Anderson, J.-L. Fattebert, R. Ferencz, N. Hodge, S. Khairallah, O. Walton, Curr. Opin. Solid State Mater. Sci. 1 (2017).
[160] J.A. Flores-Livas, A. Sanna, S. Goedecker, Nov. Supercond. Mater. 3 (2017) 6.
[161] M. Rupp, A. Tkatchenko, K.-R. Müller, O.A. vonLilienfeld, Phys. Rev. Lett. 208 (2011) 1, arXiv:1109.2618.
[162] J.C. Snyder, M. Rupp, K. Hansen, K.R. Müller, K. Burke, Phys. Rev. Lett., 108, 1 (2012) arXiv:1112.5441.
[163] E.R. Homer, Comput. Mater. Sci. 161 (2019) 244.
[164] E. Lewinson, Explaining feature importance by example of a random forest, 2019. https://towardsdatascience.com/explaining-feature-importance-by-example-of-a-random-forest-d9166011959e.
[165] C. Kamath, Int. J. Adv. Manuf. Technol. 10 (2016).
[166] S. Curtarolo, D. Morgan, K. Persson, J. Rodgers, G. Ceder, Phys. Rev. Lett. 91 (2003), 135503 arXiv:0307262 [cond-mat].
[167] K. Rajan, C. Suh, P.F. Mendez, Stat. Anal. Data Min. 1 (2009) 361.
[168] R.R. Dehoff, M.M. Kirka, W.J. Sames, H. Bilheux, A.S. Tremsin, L.E. Lowe, S. S. Babu, Mater. Sci. Technol. 31 (2015).
[169] J. Gockel, J. Beuth, K. Taminger, Addit. Manuf. 1 (2014) 119.
[170] D. Pal, N. Patil, K. Zeng, B. Stucker, J. Manuf. Sci. Eng. 136 (2014) 1.
[171] J. Ding, P. Colegrove, J. Mehnen, S. Ganguly, P.S. Almeida, F. Wang, S. Williams, Comput. Mater. Sci. 50 (2011) 3315.
[172] N. Raghavan, R. Dehoff, S. Pannala, S. Simunovic, M. Kirka, J. Turner, N. Carlson, S.S. Babu, Acta Mater. 112 (2016) 303.
[173] W. King, A. Anderson, R. Ferencz, N. Hodge, C. Kamath, S. Khairallah, Mater. Sci. Technol. 31 (2015).
[174] G. Tapia, S.A. Khairallah, M. Matthews, W.E. King, Int. J. Adv. Manuf. Technol. 10 (2017).
[175] W.E. King, H.D. Barth, V.M. Castillo, G.F. Gallegos, J.W. Gibbs, D.E. Hahn, C. Kamath, A.M. Rubenchik, J. Mater. Process. Technol. 214 (2014) 2915.
[176] M.A. Bessa, R. Bostanabad, Z. Liu, A. Hu, D.W. Apley, C. Brinson, W. Chen, W. K. Liu, Comput. Methods Appl. Mech. Eng. 320 (2017) 633.
[177] S. Berumen, F. Bechmann, S. Lindner, J.-P. Kruth, T. Craeghs, Phys. Procedia 5 (2010) 617.
[178] G. Tapia, A. Elwany, J. Manuf. Sci. Eng. 136 (2014), 060801.
[179] M. Mani, B.M. Lane, M.A. Donmez, S.C. Feng, S.P. Moylan, Int. J. Prod. Res. 55 (2017).
[180] T. Purtonen, A. Kalliosaari, A. Salminen, Phys. Procedia 56 (2014) 1218.
[181] J.T. McKeown, K. Zweiacker, C. Liu, D.R. Coughlin, A.J. Clarke, J.K. Baldwin, J. W. Gibbs, J.D. Roehling, S.D. Imhoff, P.J. Gibbs, D. Tourret, J.M. Wiezorek, G. H. Campbell, JOM 68 (2016).
[182] U.S. Bertoli, G. Guss, S. Wu, M.J. Matthews, J.M. Schoenung, Mater. Des. 135 (2017) 385.
[183] J. Raplee, A. Plotkowski, M.M. Kirka, R. Dinwiddie, A. Okello, R.R. Dehoff, S. S. Babu, Sci. Rep. 7 (2017) 1.
[184] Q. Guo, C. Zhao, L.I. Escano, Z. Young, L. Xiong, K. Fezzaa, W. Everhart, B. Brown, T. Sun, L. Chen, Acta Mater. 151 (2018) 169.
[185] C.L.A. Leung, S. Marussi, R.C. Atwood, M. Towrie, P.J. Withers, P.D. Lee, Nat. Commun. 9 (2018).
[186] S.K. Everton, M. Hirsch, P. Stravroulakis, R.K. Leach, A.T. Clare, Mater. Des. 95 (2016) 431.
[187] D. Vernon, Machine Vision, Prentice-Hall,, 1991.
[188] J. Canny, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-8 (1986) 679.
[189] J.A. Kanko, A.P. Sibley, J.M. Fraser, J. Mater. Process. Technol. 231 (2016) 488.
[190] N. Yang, J. Lee, B. Zheng, K. Gaiser, T. Reynolds, L. Clemon, W. Lu, J. Schoenung, E. Lavernia, J. Therm. Spray. Technol. 26 (2017) 610.
[191] M.J. Matthews, G. Guss, S.A. Khairallah, A.M. Rubenchik, P.J. Depond, W.E. King, Acta Mater. 114 (2016) 33.
[192] D.G. Lowe, Int. J. Comput. Vis. 60 (2004) 91.
[193] H. Bay, A. Ess, T. Tuytelaars, L.V. Gool, Comput. Vis. Image Underst. 110 (2008) 346.
[194] T. Hastie, R. Tibshirani, J. Friedman, Elements 1 (2009) 337, arXiv:1010.3003.
[195] H.K.D.H. Bhadeshia, R.C. Dimitriu, S. Forsik, J.H. Pak, J.H. Ryu, Mater. Sci. Technol. 25 (2009) 504.
[196] K. Hornik, M. Stinchcombe, H. White, Neural Netw. 2 (1989) 359.
[197] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, Adv. Neural Inf. Process. Syst. 27 (Proc. NIPS) 27 (2014) 1, arxiv:1411.1792.
[198] B. Yuan, G.M. Guss, A.C. Wilson, S.P. Hau-Riege, P.J. DePond, S. McMains, M. J. Matthews, B. Giera, Adv. Mater. Technol. 1 (2018).
[199] L. Scime, J. Beuth, Addit. Manuf. 24 (2018) 273.
[200] X. Li, X. Jia, Q. Yang, J. Lee, J. Intell. Manuf. (2020).
[201] O. Kwon, H.G. Kim, M.J. Ham, W. Kim, G.-H. Kim, J.H. Cho, N.I. Kim, K. Kim, J. Intell. Manuf. (2018) 1–12.
[202] A. Krizhevsky, I. Sutskever, G.E. Hinton, Proceedings of the 25th International Conference on Neural Information Processing Systems, 1, 2012, p. 1097.
[203] L. Scime, J. Beuth, Addit. Manuf. 25 (2019) 151.
[204] B.L. DeCost, E.A. Holm, Comput. Mater. Sci. 126 (2017) 438.
[205] J. Zhou, Y. Zhang, J. Chen, J. Manuf. Sci. Eng. 131 (2009) 1.
[206] C. Boley, S. Khairallah, A. Rubenchik, Appl. Opt. 54 (2015) 2477.
[207] B.L. DeCost, H. Jain, A.D. Rollett, E.A. Holm, Jom 69 (2017) 456.
[208] Y.-F. Shen, R. Pokharel, T.J. Nizolek, A. Kumar, T. Lookman, Acta Mater. 170 (2019) 118.

[209] K. Kaufmann, C. Zhu, A.S. Rosengarten, D. Maryanovsky, T.J. Harrington, E. Marin, K.S. Vecchio, Science 367 (2020) 564.
[210] S. Miyazaki, M. Kusano, D.S. Bulgarevich, S. Kishimoto, A. Yumoto, M. Watanabe, Mater. Trans. 50 (2019) 561–568.
[211] B.L. DeCost, T. Francis, E.A. Holm, Acta Mater. 133 (2017) 30, arxiv:1702.01117.
[212] D.S. Bulgarevich, S. Tsukamoto, T. Kasuya, M. Demura, M. Watanabe, Sci. Rep. 8 (2018) 1.
[213] A. Chowdhury, E. Kautz, B. Yener, D. Lewis, Comput. Mater. Sci. 123 (2016) 176.
[214] X.D. Xiang, X. Sun, G. Briceno, Y. Lou, K.-A. Wang, H. Chang, W.G. Wallace-Freedman, S.-W. Chen, P.G. Schultz, Science 268 (1995) 1738.
[215] G. Ceder, Y.-M. Chiang, D.R. Sadoway, M.K. Aydinol, Y.-I. Jang, B. Huang, Nature 392 (1998) 694.
[216] G. Pilania, C. Wang, X. Jiang, S. Rajasekaran, R. Ramprasad, Sci. Rep. 3 (2013) 2810.
[217] M.J. Hampden-Smith, T.T. Kodas, Chem. Vap. Depos. 1 (1995).
[218] B. Mercey, P.A. Salvador, W. Prellier, T.-D. Doan, J. Wolfman, J.-F. Hamet, M. Hervieu, B. Raveau, J. Mater. Chem. 9 (1999) 233.
[219] D.B. Mitzi, Chem. Mater. 13 (2001) 3283.
[220] J. Cui, Y.S. Chu, O.O. Famodu, Y. Furuya, J. Hattrick-Simpers, R.D. James, A. Ludwig, S. Thienhaus, M. Wuttig, Z. Zhang, I. Takeuchi, Nat. Mater. 5 (2006) 286.
[221] A. Dwivedi, T.J. Wyrobek, O.L. Warren, J. Hattrick-Simpers, O.O. Famodu, I. Takeuchi, J. Appl. Phys. 104 (2008).
[222] K. Jin, R. Suchoski, S. Fackler, Y. Zhang, X. Pan, R.L. Greene, I. Takeuchi, APL Mater. 1 (2013).
[223] J. Gregoire, D.V. Campen, C. Miller, R. Jones, S. Suram, A. Mehta, J. Synchrotron Radiat. 21 (2014) 1262.
[224] F. Ren, R. Pandolfi, D.V. Campen, A. Hexemer, A. Mehta, ACS Comb. Sci. 19 (2017) 377.
[225] D.M. Deaven, K.M. Ho, Phys. Rev. Lett. 75 (1995) 288, arxiv:9506004 [mtrl-th].
[226] S.M. Woodley, P.D. Battle, J.D. Gale, C. Richard, A. Catlow, Phys. Chem. Chem. Phys. 1 (1999) 2535.
[227] B.G. Sumpter, D.W. Noid, Annu. Rev. Mater. Sci. 26 (1996) 223.
[228] C.W. Glass, A.R. Oganov, N. Hansen, Comput. Phys. Commun. 175 (2006) 713.
[229] J. Hafner, C. Wolverton, G. Ceder, MRS Bull. 31 (2006) 659.
[230] S. Curtarolo, D. Morgan, G. Ceder, Calphad: Comput. Coupling Phase Diagr. Thermochem. 29 (2005) 163, arxiv:0502465 [cond-mat].
[231] A.N. Kolmogorov, S. Curtarolo, Phys. Rev. B Condens. Matter Mater. Phys. 73 (2006) 1, arxiv:0603304 [cond-mat].
[232] K. Kang, Y.S. Meng, J. Breger, C.P. Grey, G. Ceder, Science 311 (2006) 977.
[233] H. Chen, G. Hautier, A. Jain, C. Moore, B. Kang, R. Doe, L. Wu, Y. Zhu, Y. Tang, G. Ceder, Chem. Mater. 24 (2012) 2009.
[234] A. Mannodi-Kanakkithodi, G. Pilania, T.D. Huan, T. Lookman, R. Ramprasad, Nat. Sci. Rep. 6 (2016) 1.
[235] G. Hautier, A. Jain, T. Mueller, C. Moore, S.P. Ong, G. Ceder, Chem. Mater. 25 (2013) 2064.
[236] J.W. Bennett, K.F. Garrity, K.M. Rabe, D. Vanderbilt, Phys. Rev. Lett. 109 (2012) 1, arxiv:1206.4732v1.
[237] C.V. Ciobanu, D.T. Tambe, V.B. Shenoy, Surf. Sci. 582 (2005) 145.
[238] S. Wang, Z. Wang, W. Setyawan, N. Mingo, S. Curtarolo, Phys. Rev. X 1 (2011) 1.
[239] J. Yan, P. Gorai, B. Ortiz, S. Miller, S.A. Barnett, T. Mason, V. Stevanovic, E. S. Toberer, Energy Environ. Sci. 8 (2015) 983.
[240] D. Morgan, G. Ceder, S. Curtarolo, Meas. Sci. Technol. 16 (2005) 296.
[241] J. Behler, Int. J. Quantum Chem. 115 (2015) 1032.
[242] L.M. Ghiringhelli, J. Vybiral, S.V. Levechenko, C. Draxl, M. Scheffler, Phys. Rev. Lett. 114 (2015).
[243] A. Jain, G. Hautier, C.J. Moore, S. PingOng, C.C. Fischer, T. Mueller, K.A. Persson, G. Ceder, Comput. Mater. Sci. 50 (2011) 2295.
[244] S. Curtarolo, W. Setyawan, G.L.W. Hart, M. Jahnatek, R.V. Chepulskii, R. H. Taylor, S. Wang, J. Xue, K. Yang, O. Levy, M.J. Mehl, H.T. Stokes, D. O. Demchenko, D. Morgan, Comput. Mater. Sci. 58 (2012) 218, arxiv:1308.5715.
[245] J. Hachmann, R. Olivares-Amaya, S. Atahan-Evrenk, C. Amador-Bedolla, R. S. Sánchez-Carrera, A. Gold-Parker, L. Vogt, A.M. Brockway, A. Aspuru-Guzik, J. Phys. Chem. Lett. 2 (2011).
[246] National Institute of Materials Science, 2017.
[247] J.E. Saal, S.K. nd Murathan Aykol, B. MEredig, C. Wolverton, JOM 65 (2013).
[248] I. Foster, R. Ananthakrisnan, B. Blaiszik, K. Chard, R. Osborn, S. Tuecke, M. Wilde, J. Wozniak, Big Data and HPC, 2015.
[249] J.J. De Pablo, B. Jones, C.L. Kovacs, V. Ozolins, A.P. Ramirez, Curr. Opin. Solid State Mater. Sci. 18 (2014) 99, arxiv:1011.1669v3.
[250] M.D. Wilkinson, M. Dumontier, I.J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomber, J.-W. Boiten, L.B. daSilvaSantos, P.E. Bourne, J. Bouwman, A. J. Brookes, T. Clark, M. Crosas, I. Dillo, O. Dumon, S. Edmunds, C.T. Evelo, R. Finkers, A. Gonzalez-Beltran, A.J. Gray, P. Groth, C. Goble, J.S. Grethe, J. Heringa, P.A. 't Hoen, R. Hooft, T. Kuhn, R. Kok, J. Kok, S.J. Lusher, M. E. Martone, A. Mons, A.L. Packer, B. Persson, P. Rocca-Serra, M. Roos, R. van Schaik, S.-A. Sansone, E. Schultes, T. Sengstag, T. Slater, G. Strawn, M.A. Swertz, M. Thompson, J. van der Lei, E. van Mulligen, J. Velterop, A. Waagmeester, P. Wittenburg, K. Wolstencroft, J. Zhao, B. Mons, Sci. Data 3 (2016).
[251] M. de Jong, W. Chen, H. Geerlings, M. Asta, K.A. Persson, Sci. Data 2 (2015), 150053.
[252] E. Kim, K. Huang, A. Tomala, S. Matthews, E. Strubell, A. Saunders, A. McCallum, E. Olivetti, Nat. Sci. Data 4 (2017).
[253] CVonline, Electronic, 2019. homepages.inf.ed.ac.uk/rbf/CVonline.
[254] VisionScience, Electronic. http://www.visionscience.com/vsImages.html.
[255] S.R. Kalidindi, D.B. Brough, S. Li, A. Cecen, A.L. Blekh, F.Y.P. Congo, C. Campbell, Mater. Res. Soc. Bull. 41 (2016).
[256] J. Gubernatis, T. Lookman, Phys. Rev. Mater. 2 (2018).
[257] M. Seifi, A. Salem, J. Beuth, O. Harrysson, J.J. Lewandowski, JOM 68 (2016) 747.
[258] P. Raccuglia, K.C. Elbert, P.D.F. Adler, C. Falk, M.B. Wenny, A. Mollo, M. Zeller, S. A. Friedler, J. Schrier, A.J. Norquist, Nat. Lett. 533 (2016) 73.