ELSEVIER

Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID-19. The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information website.

70 years of machine learning in geoscience in review

Jesper Sören Dramsch*
Technical University of Denmark, Copenhagen, Denmark
*Corresponding author: e-mail address: jesper@dramsch.net

Contents

In recent years machine learning has become an increasingly important interdisciplinary tool that has advanced several fields of science, such as biology (Ching et al., 2018), chemistry (Schütt, Arbabzadah, Chmiela, Müller, & Tkatchenko, 2017), medicine (Shen, Wu, & Suk, 2017), and pharmacology (Kadurin, Nikolenko, Khrabrov, Aliper, & Zhavoronkov, 2017). Specifically, the method of deep neural networks has found wide application. While geoscience was slower in the adoption, bibliometrics show the adoption of deep learning in all aspects of geoscience. Most subdisciplines of geoscience have been treated to a review of machine learn-ing. Remote sensing has been an early adopter (Lary, Alavi, Gandomi, & Walker, 2016), with geomorphology (Valentine & Kalnins, 2016), solid Earth geoscience (Bergen, Johnson, Maarten, & Beroza, 2019), hydrogeo physics (Shen, 2018), seismology (Kong et al., 2019), seismic interpretation

(Wang, Di, Shafiq, Alaudah, & AlRegib, 2018), and geochemistry (Zuo, Xiong, Wang, & Carranza, 2019) following suite. Climate change, in particular, has received a thorough treatment of the potential impact of varying machine learning methods for modeling, engineering, and mitigation to address the problem (Rolnick et al., 2019). This review addresses the development of applied statistics and machine learning in the wider discipline of geoscience in the past 70 years and aims to provide context for the recent increase in interest and successes in machine learning and its challenges.[a]

Machine learning (ML) is deeply rooted in applied statistics, building computational models that use inference and pattern recognition instead of explicit sets of rules. Machine learning is generally regarded as a subfield of artificial intelligence (AI), with the notion of AI first being introduced by Turing (1950). Samuel (1959) coined the term machine learning itself, with Mitchell et al. (1997) providing a commonly quoted definition:

> *A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E.*
>
> **Mitchell et al. (1997)**

This means that a machine learning model is defined by a combination of requirements. A task such as, classification, regression, or clustering is improved by conditioning of the model on a training data set. The performance of the model is measured with regard to a loss, also called metric, which quantifies the performance of a machine learning model on the provided data. In regression, this would be measuring the misfit of the data from the expected values. Commonly, the model improves with exposure to additional samples of data. Eventually, a good model generalizes to unseen data, which was not part of the training set, on the same task the model was trained to perform.

Accordingly, many mathematical and statistical methods and concepts, including Bayes' rule (Bayes, 1763), least-squares (Legendre, 1805), and Markov models (Markov, 1906, 1971), are applied in machine learning. Gaussian processes stand out as they originate in time series applications (Kolmogorov, 1939) and geostatistics (Krige, 1951), which roots this machine learning application in geoscience (Rasmussen, 2003). "Kriging" originally applied two-dimensional Gaussian processes to the prediction

---

[a] The author of this chapter has a background in geophysics, exploration geoscience, and active source 4D seismic. While this skews the expertise, they attempt to give a full overview over developments in all of geoscience with the minimum amount of bias possible.

of gold mine valuation and has since found wide application in geostatistics. Generally, Matheron (1963) is credited with formalizing the mathematics of kriging and developing it further in the following decades.

Between 1950 and 2020 much has changed. Computational resources are now widely available both as hardware and software, with high-performance compute being affordable to anyone from cloud computing vendors. High-quality software for machine learning is widely available through the free- and open-source software movement, with major companies (Google, Facebook, Microsoft) competing for the usage of their open-source machine learning frameworks (Tensorflow, Pytorch, CNTK[b]) and independent developments reaching wide applications such as scikit-learn (Pedregosa et al., 2011) and xgboost (Chen & Guestrin, 2016).

Nevertheless, investigations of machine learning in geoscience are not a novel development. The research into machine learning follows interest in artificial intelligence closely. Since its inception, artificial intelligence has experienced two periods of a decline in interest and trust, which has impacted negatively upon its funding. Developments in geoscience follow this wide-spread cycle of enthusiasm and loss of interest with a time lag of a few years. This may be the result of a variety of factors, including research funding availability and a change in willingness to publish results.

1. Historic machine learning in geoscience

The 1950s and 1960s were decades of machine learning optimism, with machines learning to play simple games and perform tasks like route mapping. Intuitive methods such as k-means, Markov models, and decision trees have been used as early as the 1960s in geoscience. K-means was used to describe the cyclicity of sediment deposits (Preston & Henderson, 1964). Krumbein and Dacey (1969) give a thorough treatment of the mathematical foundations of Markov chains and embedded Markov chains in a geological context through application to sedimentological processes, which also provides a comprehensive bibliography of Markov processes in geology. Some selected examples of early applications of Markov chains are found in sedimentology (Schwarzacher, 1972), well-log analysis (Agterberg, 1966), hydrology (Matalas, 1967), and volcanology (Wickman, 1968). Decision tree-based methods found early applications in economic geology and prospectivity mapping (Newendorp, 1976; Reddy & Bonham-Carter, 1991).

---

[b] Deprecated 2019.

The 1970s were left with few developments in both the methods of machine learning, as well as, applications and adoption in geoscience (cf. Fig. 1), due to the "first AI winter" after initial expectations were not met. Nevertheless, as kriging was not considered an AI technology, it was unaffected by this cultural shift and found applications in mining (Huijbregts & Matheron, 1970), oceanography (Chiles & Chauvet, 1975), and hydrology (Delhomme, 1978). This was in part due to superior results over other interpolation techniques, but also the provision of uncertainty measures.

1.1 Expert systems to knowledge-driven AI

The 1980s marked uptake in interest in machine learning and artificial intelligence through so-called expert systems and corresponding specialized hardware. While neural networks were introduced in 1950, the tools of automatic differentiation and backpropagation for error-correcting machine learning were necessary to spark their adoption in geophysics in the late 1980s. Zhao and Mendel (1988) performed seismic deconvolution with a recurrent neural network (Hopfield network). Dowla, Taylor, and Anderson (1990) discriminated between natural earthquakes and underground nuclear explosions using feed-forward neural networks. An ensemble of networks was able
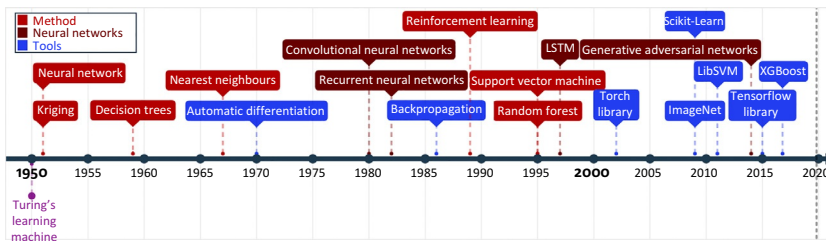

Fig. 1 Machine learning timeline from Dramsch (2019). Neural networks: Russell and Norvig (2010); Kriging: Krige (1951); decision trees: Belson (1959); nearest neighbors: Cover and Hart (1967); automatic differentiation: Linnainmaa (1970); convolutional neural networks: Fukushima (1980); LeCun, Bengio, and Hinton (2015); recurrent neural networks: Hopfield (1982); Backpropagation: Kelley (1960), Bryson (1961), Dreyfus (1962), and Rumelhart, Hinton, Williams, et al. (1988); reinforcement learning: Watkins (1989); support-vector machines: Cortes and Vapnik (1995); random forests: Ho (1995); LSTM: Hochreiter and Schmidhuber (1997); torch library: Collobert, Bengio, and Mariéthoz (2002); ImageNet: Deng et al. (2009); scikit-learn: Pedregosa et al. (2011); LibSVM: (Chang & Lin, 2011); generative adversarial networks: Goodfellow et al. (2014); Tensorflow: Abadi et al. (2015); XGBoost: Chen and Guestrin (2016).

to achieve 97% accuracy for nuclear monitoring. Moreover, the researchers inspected the network to gain the insight that the ratio of particular input spectra was beneficial to the discrimination of seismological events to the network. However, in practice the neural networks underperformed on uncurated data, which is often the case in comparison to published results. Huang, Chang, and Yen (1990) presented work on self-organizing maps (also Kohonen networks), a special type of unsupervised neural network applied to pick seismic horizons. The field of geostatistics saw a formalization of theory and an uptake in interest with Matheron et al. (1981) formalizing the relationship of spline-interpolation and kriging and Dubrule (1984) further develop the theory and apply it to well data. At this point, kriging is well-established in the mining industry as well as other disciplines that rely on spatial data, including the successful analysis and construction of the channel tunnel (Chilès & Desassis, 2018). The late 1980s then marked the second AI winter, where expensive machines tuned to run "expert systems" were outperformed by desktop hardware from nonspecialist vendors, causing the collapse of a half-billion-dollar hardware industry. Moreover, government agencies cut funding in AI specifically.

The 1990s are generally regarded as the shift from a knowledge-driven to a data-driven approach in machine learning. The term AI and especially expert systems were almost exclusively used in computer gaming and regarded with cynicism and as a failure in the scientific world. In the background, however, with research into applied statistics and machine learning, this decade marked the inception of support-vector machines (SVM) (Cortes & Vapnik, 1995), the tree-based method random forests (RF) (Ho, 1995), and a specific type of recurrent neural network (RNN) long short-term memories (LSTM) (Hochreiter & Schmidhuber, 1997). SVMs were utilized for land usage classification in remote sensing early on (Hermes, Frieauff, Puzicha, & Buhmann, 1999). Geophysics applied SVMs a few years later to approximate the Zoeppritz equations for AVO inversion, outperforming linearized inversion (Kuzma, 2003). Random forests, however, were delayed in broader adoption, due to the term "random forests" only being coined in 2001 (Breiman, 2001) and the statistical basis initially being less rigorous and implementation being more complicated. LSTMs necessitate large amounts of data for training and can be expensive to train, after further development in 2011 (Ciresan, Meier, Masci, Gambardella, & Schmidhuber, 2011) it gained popularity in commercial time series applications particularly speech and audio analysis.

1.2 Neural networks

McCormack (1991) marks the first review of the emerging tool of neural networks in geophysics. The paper goes into the mathematical details and explores pattern recognition. The author summarizes neural network applications over the 30 years prior to the review and presents worked examples in automated well–log analysis and seismic trace editing. The review comes to the conclusion that neural networks are, in fact, good function approximators, taking over tasks that were previously reserved for human work. He criticizes slow training, the cost of retraining networks upon new knowledge, imprecision of outputs, nonoptimal training results, and the black box property of neural networks. The main conclusion sees the implementation of neural networks in conventional computation and expert systems to leverage the pattern recognition of networks with the advantages of conventional computer systems.

Neural networks are the primary subject of the modern day machine learning interest, however, significant developments leading up to these successes were made prior to the 1990s. The first neural network machine was constructed by Minsky [described in Russell and Norvig (2010)] and soon followed by the "Perceptron," a binary decision boundary learner (Rosenblatt, 1958). This decision was calculated as follows:

$$
\begin{aligned}
o_j &= \sigma \left(\sum_j w_{ij} x_i + b \right) \\
&= \sigma \left(a_j \right) \\
&= \begin{cases} 1 & a_j > 0 \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
\tag{1}
$$

It describes a linear system with the output o, the linear activation a of the input data x, the index of the source i and target node j, the trainable weights w, the trainable bias b and a binary activation function σ (Fig. 2). The activation function σ in particular has received ample attention since its inception. During this period, a binary σ became uncommon and was replaced by nonlinear mathematical functions. Neural networks are commonly trained by gradient descent, therefore, differentiable functions like sigmoid or tanh, allowing for the activation o of each neuron in a neural network to be continuous (Fig. 4).

Deep learning (Dechter, 1986) expands on this concept. It is the combination of multiple layers of neurons in a neural network (Fig. 3).
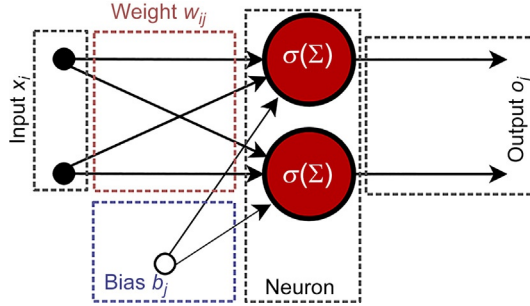
Fig. 2 Single layer neural network as described in Eq. (1). Two inputs x_i are multiplied by the weights w_{ij} and summed with the biases b_j. Subsequently an activation function σ is applied to obtain out outputs o_j.
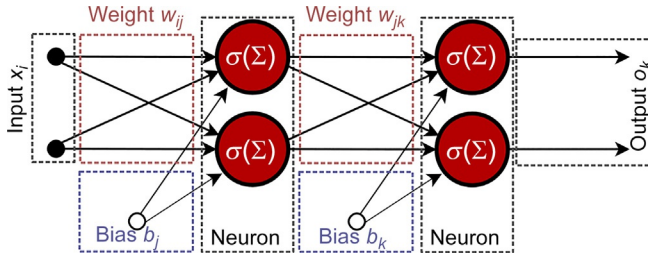

Fig. 3 Deep multilayer neural network as described in Eq. (2).

These deep networks learn representations with multiple levels of abstraction and can be expressed using Eq. (1) as input neurons to the next layer

$$o_k = \sigma \left(\sum_k w_{jk} \cdot o_j + b \right)$$
$$= \sigma \left(\sum_k w_{jk} \cdot \sigma \left(\sum_j w_{ij} x_i + b \right) + b \right) \quad (2)$$

Röth and Tarantola (1994) apply these building blocks of multilayered neural networks with sigmoid activation to perform seismic inversion. They successfully invert low-noise and noise-free data on small training data. The authors note that the approach is susceptible to errors at low signal–to–noise ratios and coherent noise sources. Further applications include electromagnetic subsurface localization (Poulton, Sternberg, & Glass, 1992), magnetotelluric inversion via Hopfield neural networks (Zhang & Paulson, 1997), and geomechanical microfractures modeling in triaxial compression tests (Feng & Seto, 1998) (Fig. 4).
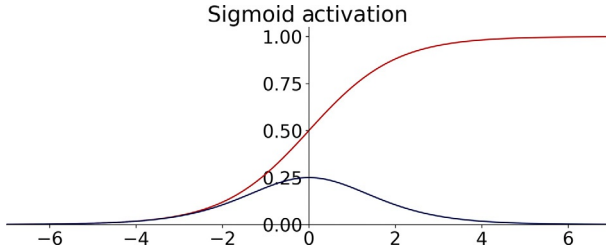
Fig. 4 Sigmoid activation function (*red*) and derivative (*blue*) to train multilayer neural network described in Eq. (2).

1.3 Kriging and Gaussian processes

Cressie (1990) reviews the history of kriging, prompted by the uptake of interest in geostatistics. The author defines kriging as best linear unbiased prediction and reviews the historical codevelopment of disciplines. Similar concepts were developed with mining, meteorology, physics, plant and animal breeding, and geodesy that relied on optimal spatial prediction. Later, Williams (1998) provide a thorough treatment of Gaussian processes, in the light of recent successes of neural networks.

> *An alternative method of putting a prior over functions is to use a Gaussian process (GP) prior over functions. This idea has been used for a long time in the spatial statistics community under the name of "kriging," although it seems to have been largely ignored as a general-purpose regression method.*
>
> **Williams (1998)**

Overall, Gaussian processes benefit from the fact that a Gaussian distribution will stay Gaussian under conditioning. That means that we can use Gaussian distributions in this machine learning process and they will produce a smooth Gaussian result after conditioning on the training data. To become a universal machine learning model, Gaussian processes have to be able to describe infinitely many dimensions. Instead of storing infinite values to describe this random process, Gaussian processes go the path of describing a distribution over functions that can produce each value when required.

$$p(x) \approx \mathcal{GP}(\mu(x), k(x, x')), \tag{3}$$

The multivariate distribution over functions $p(x)$ is described by the Gaussian process depends on mean a function $\mu(x)$ and a covariance function $k(x, x')$. It follows that choosing an appropriate mean and covariance function, also known as kernel, is essential. Very commonly, the mean function is chosen to be zero, as this simplifies some of the math. Therefore, data with a nonzero mean are commonly centered to comply with this assumption

(Görtler, Kehlbeck, & Deussen, 2019). Choosing an appropriate kernel for the machine learning task is one of the benefits of the Gaussian process. The kernel is where expert knowledge can be incorporated into data, e.g., seasonality meteorological data can be described by a periodic covariance function.

Fig. 5 presents a 2D slice of 3D data with two classes. This binary problem can be approached by applying a Gaussian process to it. In the second panel, a linear kernel is shown, which predicts the data relatively poorly with an accuracy of 71%. A radial basis function (RBF) kernel, shown in the third panel generalizes to unseen test data with an accuracy of 90%. This figure shows how a trained Gaussian process would predict any new data point presented to the model. The linear kernel would predict any data in the top part to be blue (Class 0) and any data in the bottom part to be red (Class 1). The RBF kernel, which we explore further in the section introducing support–vector machines, separates the prediction into four uneven quadrants. The choice of kernel is very important in Gaussian processes and research into extracting specific kernels is ongoing (Duvenaud, 2014).

In a more practical sense, Gaussian processes are computationally expensive, as an $n \times n$ matrix must be inverted, with n being the number of samples. This results in a space complexity of $\mathcal{O}(n^2)$ and a time complexity $\mathcal{O}(n^3)$ (Williams & Rasmussen, 2006). This makes Gaussian processes most feasible for smaller data problems, which is one explanation for their rapid uptake in geoscience. An approximate computation of the inverted matrix is possible using the conjugate gradient (CG) optimization method, which can be stopped early with a maximum time cost of $\mathcal{O}(n^3)$ (Williams & Rasmussen, 2006). For problems with larger data sets, neural networks
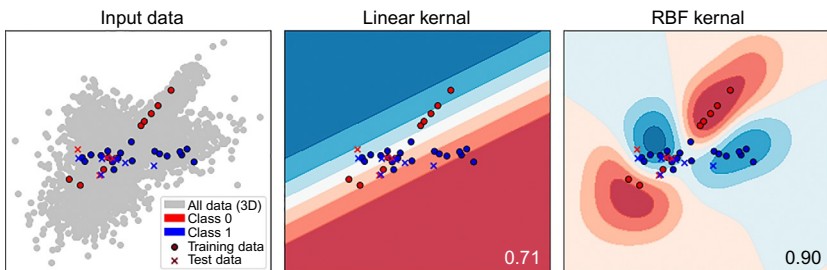
Fig. 5 Gaussian process separating two classes with different kernels. This image presents a 2D slice out of a 3D decision space. The decision boundary learnt from the data is visible, as well as the prediction in every location of the 2D slice. The two kernels presented are a linear kernel and a radial basis function (RBF) kernel, which show a significant discrepancy in performance. The bottom right number shows the accuracy on unseen test data. The linear kernel achieves 71% accuracy, while the RBF kernel achieves 90%.

become feasible due to being computationally cheaper than Gaussian processes, regularization on large data sets being viable, as well as, their flexibility to model a wide variety of functions and objectives. Regularization being essential as neural networks tend to not "overfit" and simply memorize the training data, instead of learning a generalizable relationship of the data. Interestingly, Hornik, Stinchcombe, and White (1989) showed that neural networks are a universal function approximator as the number of weights tend to infinity, and Neal (1996) was able to show that the infinitely wide stochastic neural network converges to a Gaussian process. Oftentimes Gaussian processes are trained on a subset of a large data set to avoid the computational cost. Gaussian processes have seen successful application on a wide variety of problems and domains that benefit from expert knowledge.

The 2000s were opened with a review by van der Baan and Jutten (2000) recapitulating the most recent geophysical applications in neural networks. They went into much detail on the neural networks theory and the difficulties in building and training these models. The authors identify the following subsurface geoscience applications through history: First-break picking, electromagnetics, magnetotellurics, seismic inversion, shear-wave splitting, well-log analysis, trace editing, seismic deconvolution, and event classification. They reveal a strong focus on exploration geophysics. The authors evaluated the application of neural networks as subpar to physics-based approaches and concluded that neural networks are too expensive and complex to be of real value in geoscience. This sentiment is consistent with the broader perception of artificial intelligence during this decade. Artificial intelligence and expert systems over-promised human-like performance, causing a shift in focus on research into specialized subfields, e.g., machine learning, fuzzy logic, and cognitive systems.

2. Contemporary machine learning in geoscience

Mjolsness and DeCoste (2001) review machine learning in a broader context outside of exploration geoscience. The authors discuss recent successes in applications of remote sensing and robotic geology using machine learning models. They review graphical models (hidden), Markov models, and SVMs and go on to disseminate the limitations of applications to vector data and poor performance when applied to rich data, such as graphs and text data. Moreover, the authors from NASA JPL go into detail on pattern recognition in automated rovers to identify geological prospects on Mars. They state:

> *The scientific need for geological feature catalogs has led to multiyear human surveys of Mars orbital imagery yielding tens of thousands of cataloged, characterized features including impact craters, faults, and ridges.*
>
> **Mjolsness and DeCoste (2001)**

The review points out the profound impact SVMs have on identifying geomorphological features without modeling the underlying processes.

2.1 Modern machine learning tools

This decade of the 2000s introduces a shift in tooling, which is a direct contributor to the recent increase in adoption and research of both shallow and deep machine learning research.

Machine learning software has been primarily comprised of proprietary software like MATLAB® with the Neural Networks Toolbox and Wolfram Mathematica or independent university projects like the Stuttgart Neural Network Simulator (SNNS). These tools were generally closed source and hard or impossible to extend and could be difficult to operate due to limited accompanying documentation. Early open–source projects include WEKA (Witten, Frank, & Hall, 2005), a graphical user interface to build machine learning and data mining projects. Shortly after that, LibSVM was released as free open–source software (FOSS) (Chang & Lin, 2011), which implements support–vector machines efficiently. It is still used in many other libraries to this day, including WEKA (Chang & Lin, 2011). Torch was then released in 2002, which is a machine learning library with a focus on neural networks. While it has been discontinued in its original implementation in the programming language Lua (Collobert et al., 2002), PyTorch, the reimplementation in the programming language Python, is one of the leading deep learning frameworks at the time of writing (Paszke et al., 2017). In 2007, the libraries Theano and scikit-learn were released openly licensed in Python (Pedregosa et al., 2011; Theano Development Team, 2016). Theano is a neural network library that was a tool developed at the Montreal Institute for Learning Algorithms (MILA) and ceased development in 2017 after strong industrial developers had released openly licensed deep learning frameworks. Scikit-learn implements many different machine learning algorithms, including SVMs, Random forests and single–layer neural networks, as well as utility functions including cross-validation, stratification, metrics and train–test splitting, necessary for robust machine learning model building and evaluation.

2.2 Support-vector machines

The impact of scikit–learn has shaped the current machine learning software package by implementing a unified application programming interface (API) (Buitinck et al., 2013). This API is explored by example in the following code snippets, the code can be obtained at Dramsch (2020b). First, we generate a classification dataset using a utility function. The `make_classification` function takes different arguments to adjust the desired arguments, we are generating 5000 samples (`n_samples`) for two classes, with five features (`n_features`), of which three features are actually relevant to the classification (`n_informative`). The data are stored in X, whereas the labels are contained in y.

```python
# Generate random classification dataset for example
from sklearn.datasets import make_classification
X, y = make_classification(n_samples=5000, n_features=5,
                           n_informative=3, n_redundant=0,
                           random_state=0, shuffle=False)
```

It is good practice to divide the available labeled data into a training data set and a validation or test data set. This split ensures that models can be evaluated on unseen data to test the generalization to unseen samples. The utility function `train_test_split` takes an arbitrary amount of input arrays and separates them according to specified arguments. In this case 25% of the data are kept for the hold–out validation set and not used in training. The `random_state` is fixed to make these examples reproducible.

```python
# Split data into train and validation set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=.25,
                                                    random_state=0)
```

Then we need to define a machine learning model, considering the previous discussion of high impact machine learning models, the first example is an SVM classifier. This example uses the default values for hyperparameters of the SVM classifier, for best results on real–world problems these have to be adjusted. The machine learning training is always done by calling `classifier.fit(X, y)` on the classifier object, which in this case is the SVM object. In more detail, the `.fit()` method implements an optimization loop that will condition the model to the training data by minimizing the defined loss function. In the case of the SVM classification the parameters are adjusted to optimize a hinge loss, outlined in Eq. (5). The trained model scikit–learn model contains information about all its hyperparameters in

addition to the trained model, shown below. The exact meaning of all these hyperparameters is laid out in the scikit-learn documentation (Buitinck et al., 2013).

```
# Define and train a Support Vector Machine Classifier
from sklearn.svm import SVC
svm = SVC(random_state=0)
svm.fit(X_train, y_train)

>>> SVC(C=1.0, break_ties=False, cache_size=200,
        class_weight=None, coef0=0.0, degree=3,
        decision_function_shape='ovr', gamma='scale',
        kernel='rbf', max_iter=-1, probability=False,
        random_state=0, shrinking=True, tol=0.001,
        verbose=False)
```

The trained SVM can be used to predict on new data, by calling `classifier.predict(data)` on the trained classifier object. The new data has to contain four features like the training data did. Generally, machine learning models always need to be trained on the same set of input features as the data available for prediction. The `.predict()` method outputs the most likely estimate on the new data to generate predictions. In the following code snippet, three predictions on three input vectors are performed on the previously trained model.

```
# Predict on new data with trained SVM
print(svm.predict([[0, 0, 0, 0, 0],
                   [-1, -1, -1, -1, -1],
                   [1, 1, 1, 1, 1]]))
>>> [1 0 1]
```

The blackbox model should be evaluated with the `classifier.score()` function. Evaluating the performance on the training data set gives an indication how well the model is performing, but this is generally not enough to gauge the performance of machine learning models. In addition, the trained model has to be evaluated on the hold-out set, a dataset the model has not been exposed to during training. This avoids that the model only performs well on the training data by "memorization" instead of extracting meaningful generalizable relationships, an effect called overfitting. In this example the hyperparameters are left to the default values, in real-life applications hyperparameters are usually adjusted to build better models. This can lead to an addition meta-level of overfitting on the hold-out set, which necessitates an additional third hold-out set to test the generalizability of the trained model with optimized hyperparameters. The default score uses the class accuracy, which suggests our model is approximately 90% correct. Similar train and test scores indicate that the model learned a generalizable

model, enabling prediction on unseen data without a performance loss. Large differences between the training score and test score indicate either overfitting, in the case of a better training score. A higher test score than training score can be an indication of a deeper problem with the data split, scoring, class imbalances, and needs to be investigated by means of external cross-validation, building standard "dummy" models, independence tests, and further manual investigations.

```python
# Score SVM on train and test data
print(svm.score(X_train, y_train))
print(svm.score(X_test, y_test))
>>> 0.9098666666666667
>>> 0.9032
```

Support-vector machines can be employed for each class of machine learning problem, i.e., classification, regression, and clustering. In a two-class problem, the algorithm considers the n-dimensional input and attempts to find a $(n-1)$-dimensional hyperplane that separates these input data points. The problem is trivial if the two classes are linearly separable, also called a hard margin. The plane can pass the two classes of data without ambiguity. For data with an overlap, which is usually the case, the problem becomes an optimization problem to fit the ideal hyperplane. The hinge loss provides the ideal loss function for this problem, yielding 0 if none of the data overlap, but a linear residual for overlapping points that can be minimized:

$$\max\left(0, \left(1 - \gamma_i(\vec{w} \cdot \vec{x}_i - b)\right)\right), \tag{4}$$

with γ_i being the current target label and $\vec{w} \cdot \vec{x}_i - b$ being the hyperplane under consideration. The hyperplane consists of w the normal vector and point x, with the offset b. This leads the algorithm to optimize

$$\left[\frac{1}{n}\sum_{i=1}^{n} \max\left(0, 1 - \gamma_i(w \cdot x_i - b)\right)\right] + \lambda \|w\|^2, \tag{5}$$

with λ being a scaling factor. For small λ the loss becomes the hard margin classifier for linearly separable problems. The nature of the algorithm dictates that only values for $\vec{x}$ close to the hyperplane define the hyperplane itself; these values are called the support vectors.

The SVM algorithm would not be as successful if it were simply a linear classifier. Some data can become linearly separable in higher dimensions. This, however, poses the question of how many dimensions should be

searched, because of the exponential cost in computation that follows due to the increase of dimensionality (also known as the curse of dimensionality). Instead, the "kernel trick" was proposed (Aizerman, 1964), which defines a set of values that are applied to the input data simply via the dot product. A common kernel is the radial basis function (RBF), which is also the kernel we applied in the example. The kernel is defined as:

$$k\left(\vec{x}_i, \vec{x}_j\right) \rightarrow \exp\left(-\gamma \parallel \vec{x}_i - \vec{x}_j \parallel^2\right) \tag{6}$$

This specifically defines the Gaussian radial basis function of every input data point with regard to a central point. This transformation can be performed with other functions (or kernels), such as, polynomials or the sigmoid function. The RBF will transform the data according to the distance between x_i and X_j, this can be seen in Fig. 6. This results in the decision surface in Fig. 7 consisting of various Gaussian areas. The RBF is generally regarded as a good default, in part, due to being translation invariant (i.e., stationary) and smoothly varying.

An important topic in machine learning is explainability, which inspects the influence of input variables on the prediction. We can employ the utility function `permutation_importance` to inspect any model and how they perform with regard to their input features (Breiman, 2001). The permutation importance evaluates how well the blackbox model performs, when a feature is not available. Practically, a feature is replaced with random noise. Subsequently, the score is calculated, which provides a representation how
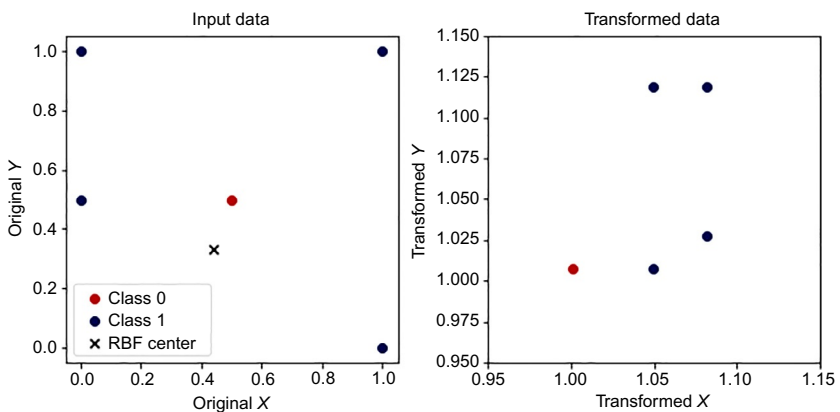

Fig. 6 Samples from two classes that are not linearly separable input data (*left*). Applying a Gaussian radial basis function centered around (0.4, 0.33) with $\lambda = 0.5$ results in the two classes being linearly separable.
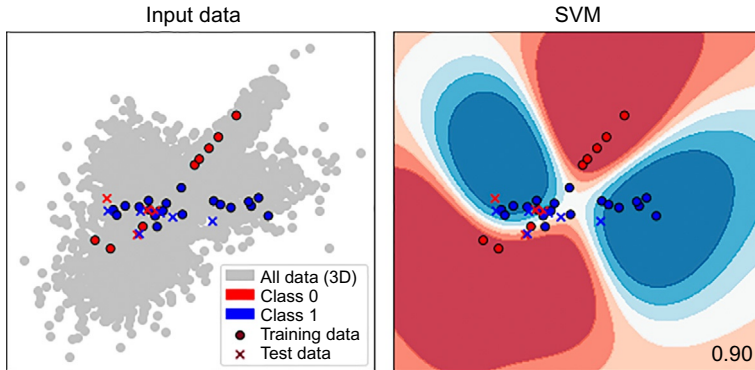
Fig. 7 Example of support-vector machine separating two classes, showing the decision boundary learnt from the data. The data contain three informative features, the decision boundary is therefore three dimensional, shown is a central slice of data points in 2D. (A video is available in Dramsch (2020a).)

informative a feature is compared to noise. The data we generated in the first example contain three informative features and two random data columns. The mean values of the calculated importances show that three features are estimated to be three magnitudes more important, with the second feature containing the maximum amount of information to predict the labels.

```
# Calculate permutation importance of SVM model
from sklearn.inspection import permutation_importance
importances = permutation_importance(svm, X_train, y_train,
                                     n_repeats=10, random_state=0)

# Show mean value of importances and the ranking
print(importances.importances_mean)
print(importances.importances_mean.argsort())
>>> [ 2.1787e-01  2.8712e-01  1.2293e-01 -1.8667e-04  7.7333e-04]
>>> [3 4 2 0 1]
```

Support–vector machines were applied to seismic data analysis (Li & Castagna, 2004) and the automatic seismic interpretation (Di, Shafiq, & AlRegib, 2017b; Liu et al., 2015; Mardan, Javaherian, et al., 2017). Compared to convolutional neural networks, these approaches usually do not perform as well, when the CNN can gain information from adjacent samples. Seismological volcanic tremor classification (Masotti et al., 2008; Masotti, Falsaperla, Langer, Spampinato, & Campanini, 2006) and analysis of ground–penetrating radar (Pasolli, Melgani, & Donelli, 2009; Xie, Qin, Yu, & Liu, 2013) were other notable applications of SVM in Geoscience. The 2016 Society of Exploration Geophysicists (SEG) machine learning challenge was held

using a SVM baseline (Hall, 2016). Several other authors investigated well–log analysis (Anifowose, Ayadiuno, & Rashedian, 2017; Caté, Schetselaar, Mercier-Langevin, & Ross, 2018; Gupta, Rai, Sondergeld, & Devegowda, 2018; Saporetti, da Fonseca, Pereira, & de Oliveira, 2018), as well as seismology for event classification (Malfante et al., 2018) and magnitude determination (Ochoa, Niño, & Vargas, 2018). These rely on SVMs being capable of regression on time series data. Generally, many applications in geoscience have been enabled by the strong mathematical foundation of SVMs, such as microseismic event classification (Zhao & Gross, 2017), seismic well ties (Chaki, Routray, & Mohanty, 2018), landslide susceptibility (Ballabio & Sterlacchini, 2012; Marjanović, Kovačević, Bajat, & Voženílek, 2011), digital rock models (Ma, Jiang, Tian, & Couples, 2012), and lithology mapping (Cracknell & Reading, 2013).

2.3 Random forests

The following example shows the application of random forests, to illustrate the similarity of the API for different machine learning algorithms in the scikit-learn library. The random forest classifier is instantiated with a maximum depth of seven, and the random state is fixed to zero again. Limiting the depth of the forest forces the random forest to conform to a simpler model. Random forests have the capability to become highly complex models that are very powerful predictive models (Fig. 8). This is not conducive to this small example dataset, but easy to modify for the inclined reader. The classifier is then trained using the same API of all classifiers in scikit-learn. The example shows a very high number of hyperparameters, however, random forests work well without further optimization of these.

```python
# Define and train a Random Forest Classifier
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(max_depth=7, random_state=0)
rf.fit(X_train, y_train)

>>> RandomForestClassifier(bootstrap=True, ccp_alpha=0.0,
                class_weight=None, criterion='gini', max_depth=7,
                max_features='auto', max_leaf_nodes=None,
                max_samples=None, min_impurity_decrease=0.0,
                min_impurity_split=None, min_samples_leaf=1,
                min_samples_split=2, min_weight_fraction_leaf=0.0,
                n_estimators=100, n_jobs=None, oob_score=False,
                random_state=0, verbose=0, warm_start=False)
```

The prediction of the random forest is performed in the same API call again, also consistent with all classifiers available. The values are slightly different from the prediction of the SVM.

```
# Predict on new data with trained Random Forest
print(rf.predict([[0, 0, 0, 0, 0],

                  [-1, -1, -1, -1, -1],
                  [1, 1, 1, 1, 1]]))
>>> [1 0 1]
```

The training score of the random forest model is 2.5% better than the SVM in this instance, this score however not informative. Comparing the test scores shows only a 0.88% difference, which is the relevant value to evaluate, as it shows the performance of a model on data it has not seen during the training stage. The random forest performed slightly better on the training set than the test data set. This slight discrepancy is usually not an indicator of an overfit model. Overfit models "memorize" the training data and do not generalize well, which results in poor performance on unseen data. Generally, overfitting is to be avoided in real application, but can be seen in competitions, on benchmarks, and show-cases of new algorithms and architectures to oversell the improvement over state-of-the-art methods (Recht, Roelofs, Schmidt, & Shankar, 2019).

```
# Score Random Forest on train and test data
print(rf.score(X_train, y_train))
print(rf.score(X_test, y_test))
>>> 0.9306
>>> 0.912
```

Random forests have specialized methods available for introspection, which can be used to calculate feature importance. These are based on the decision process the random forest used to build the machine learning model. The feature importance in random forests uses the same method as permutation importance, which is dropping out features to estimate their importance on the model performance. Random forests use a measure to determine the split between classes at each node of the trees called Gini impurity. While the permutation importance uses the accuracy score of the prediction, in random forests this Gini impurity can be used to measure how informative a feature is in a model. It is important to note that this impurity-based process can be susceptible to noise and overestimate high number of classes in features. Using the permutation importance instead is a valid choice. In this instance as opposed to the permutation importance, the random forest

estimates the two noninformative features to be one magnitude less useful than the informative features, instead of two magnitudes.

```
# Inspect random forest for feature importance
print(rf.feature_importances_)
print(rf.feature_importances_.argsort())
>>> [0.2324 0.4877 0.2527 0.0141 0.0129]
>>> [4 3 0 2 1]
```

Random forests and other tree-based methods, including gradient boosting, a specialized version of random forests, have generally found wider application with the implementation into scikit-learn and packages for the statistical languages R and SPSS. Similar to neural networks, this method is applied to ASI (Guillen, Larrazabal, González, Boumber, & Vilalta, 2015) with limited success, which is due to the independent treatment of samples, like SVMs. Random forests have the ability to approximate regression problems and time series, which made them suitable for seismological applications including localization (Dodge & Harris, 2016), event classification in volcanic tremors (Maggi et al., 2017), and slow slip analysis (Hulbert et al., 2018). They have also been applied to geomechanical applications in fracture modeling (Valera et al., 2017) and fault failure prediction (Rouet-Leduc, Hulbert, & Bolton, 2018; Rouet-Leduc, Hulbert, & Lubbers, 2017), as well as, detection of reservoir property changes from 4D seismic data (Cao & Roy, 2017). Gradient boosted trees were the winning models in the 2016 SEG machine learning challenge (Hall & Hall, 2017) for well-log analysis, propelling a variety of publications in facies prediction (Bestagini, Lipari, & Tubaro, 2017; Blouin, Caté, Perozzi, & Gloaguen, 2017; Caté et al., 2018; Saporetti et al., 2018).

Furthermore, various methods that have been introduced into scikit-learn have been applied to a multitude of geoscience problems. Hidden Markov models were used on seismological event classification (Beyreuther & Wassermann, 2008; Bicego, Acosta-Muñoz, & Orozco-Alzate, 2013; Ohrnberger, 2001), well-log classification (Jeong, Park, Han, & Kim, 2014; Wang, Wellmann, Li, Wang, & Liang, 2017), and landslide detection from seismic monitoring (Dammeier, Moore, Hammer, Haslinger, & Loew, 2016). These hidden Markov models are highly performant on time series and spatially coherent problems. The "hidden" part of Markov models enables the model to assume influences on the predictions that are not directly represented in the input data. The K-nearest neighbors method has been used for well-log analysis (Caté, Perozzi, Gloaguen, & Blouin, 2017; Saporetti et al., 2018), seismic well ties (Wang, Lomask, & Segovia, 2017) combined with dynamic
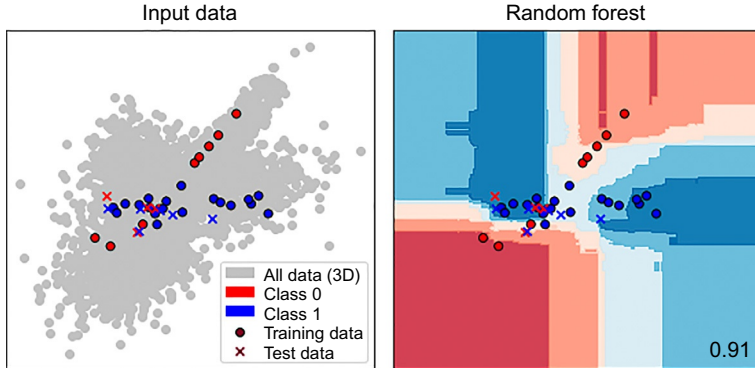
Fig. 8 Binary decision boundary for random forest in 2D. This is the same central slice of the 3D decision volume used in Fig. 7.

time warping and fault extraction in seismic interpretation (Hale, 2013), which is highly dependent on choosing the right hyperparameter k. The unsupervised k–NN equivalent, k–means has been applied to seismic interpretation (Di, Shafiq, & AlRegib, 2017 a), ground motion model validation (Khoshnevis & Taborda, 2018), and seismic velocity picking (Wei, Yonglin, Qingcai, Jiaqiang, et al., 2018). These are very simple machine learning models that are useful for baseline models. Graphical modeling in the form of Bayesian networks has been applied to seismology in modeling earthquake parameters (Kuehn & Riggelsen, 2011), basin modeling (Martinelli, Eidsvik, Sinding-Larsen, et al., 2013), seismic interpretation (Ferreira, Brazil, Silva, et al., 2018), and flow modeling in discrete fracture networks (Karra, O'Malley, Hyman, Viswanathan, & Srinivasan, 2018). These graphical models are effective in causal modeling and gained popularity in modern applications of machine learning explainability, interpretability, and generalization in combination with do–calculus (Pearl, 2012).

2.4 Modern deep learning

The 2010s marked a renaissance of deep learning and particularly convolutional neural networks. The convolutional neural network (CNN) architecture AlexNet (Krizhevsky, Sutskever, & Hinton, 2012) was the first CNN to enter the ImageNet challenge (Deng et al., 2009). The ImageNet challenge is considered a benchmark competition and database of natural images established in the field of computer vision. This improved the classification error rate from 25.8% to 16.4% (top–5 accuracy). This has

propelled research in CNNs, resulting in error rates on ImageNet of 2.25% on top–5 accuracy in 2017 (Russakovsky et al., 2015). The Tensorflow library (Abadi et al., 2015) was introduced for open–source deep learning models, with some different software design compared to the Theano and Torch libraries.

The following example shows an application of deep learning to the data presented in the previous examples. The classification data set we use has independent samples, which leads to the use of simple densely connected feed–forward networks. Image data or spatially correlated datasets would ideally be fed to a convolutional neural network (CNN), whereas time series are often best approached with recurrent neural networks (RNN). This example is written using the Tensorflow library. PyTorch would be an equally good library to use.

All modern deep learning libraries take a modular approach to building deep neural networks that abstract operations into layers. These layers can be combined into input and output configurations in highly versatile and cus-tomizable ways. The simplest architecture, which is the one we implement below, is a sequential model, which consists of one input and one output layer, with a "stack" of layers. It is possible to define more complex models with multiple inputs and outputs, as well as the branching of layers to build very sophisticated neural network pipelines. These models are called func-tional API and subclassing API, but would not be conducive to this example.

The example model consists of Dense layers and a Dropout layer, which are arranged in sequence. Densely connected layers contain a specified num-ber of neurons with an appropriate activation function, shown in the example below. Each neuron performs the calculation outlined in Eq. (1), with σ defining the activation. Modern neural networks rarely implement sigmoid and tanh activations anymore. Their activation characteristic leads them to lose information for large positive and negative values of the input, commonly called saturation (Hochreiter, Bengio, Frasconi, Schmidhuber, et al., 2001). This saturation of neurons prevented good deep neural network performance until new nonlinear activation functions took their place (Xu, Wang, Chen, & Li, 2015). The activation function Rectified linear unit (ReLU) is generally credited with facilitating the development of very deep neural networks, due to their nonsaturating properties (Hahnloser, Sarpeshkar, Mahowald, Douglas, & Seung, 2000). It sets all negative values to zero and provides a linear response for positive values, as seen in Eq. (7) and Fig. 9. Since it is inception, many more rectifiers with different properties have been introduced.
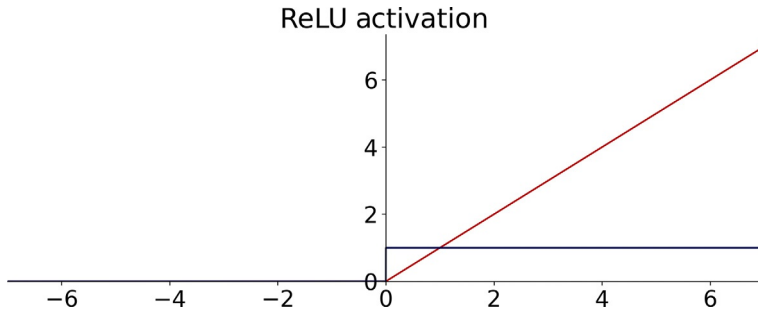
Fig. 9 ReLU activation (*red*) and derivative (*blue*) for efficient gradient computation.

$$\sigma(a) = max(0, a) \tag{7}$$

The other activation function used in the example is the "softmax" function on the output layer. This activation is commonly used for classification tasks, as it normalizes all activations at all outputs to one. It achieves this by applying the exponential function to each of the outputs in $\vec{a}$ for class C and dividing that value by the sum of all exponentials:

$$\sigma(\vec{a}) = \frac{e^{a_j}}{\sum\limits_{p}^{C} e^{a_p}} \tag{8}$$

The example additionally uses a Dropout layer, which is a common layer used for regularization of the network by randomly setting a specified percentage of nodes to zero for each iteration. Neural networks are particularly prone to overfitting, which is counteracted by various regularization techniques that also include input data augmentation, noise injection, $\mathcal{L}_1$ and $\mathcal{L}_2$ constraints, or early-stopping of the training loop (Goodfellow, Bengio, & Courville, 2016). Modern deep learning systems may even leverage noisy student–teacher networks for regularization (Xie, Hovy, Luong, & Le, 2019).

```python
import tensorflow as tf
model = tf.keras.models.Sequential([
tf.keras.layers.Dense(32, activation='relu'),
tf.keras.layers.Dropout(.3),
tf.keras.layers.Dense(16, activation='relu'),
tf.keras.layers.Dense(2, activation='softmax')])
```

These sequential models are also used for simple image classification models using CNNs. Instead of Dense layers, these are built up with convolutional layers, which are readily available in 1D, 2D, and 3D as Conv1D, Conv2D,

and Conv3D, respectively. A two-dimensional CNN learns a so-called filter f for the $n \times m$-dimensional image G, expressed as:

$$G^*(x, y) = \sum_{i=1}^{n}\sum_{j=1}^{m} f(i, j) \cdot G(x-i+c, \ y-j+c), \tag{9}$$

resulting in the central result G^* around the central coordinate c. In CNNs each layer learns several of these filters f, usually following by a down-sampling operation in n and m to compress the spatial information. This serves as a forcing function to learn increasingly abstract representations in subsequent convolutional layers (Fig. 10).
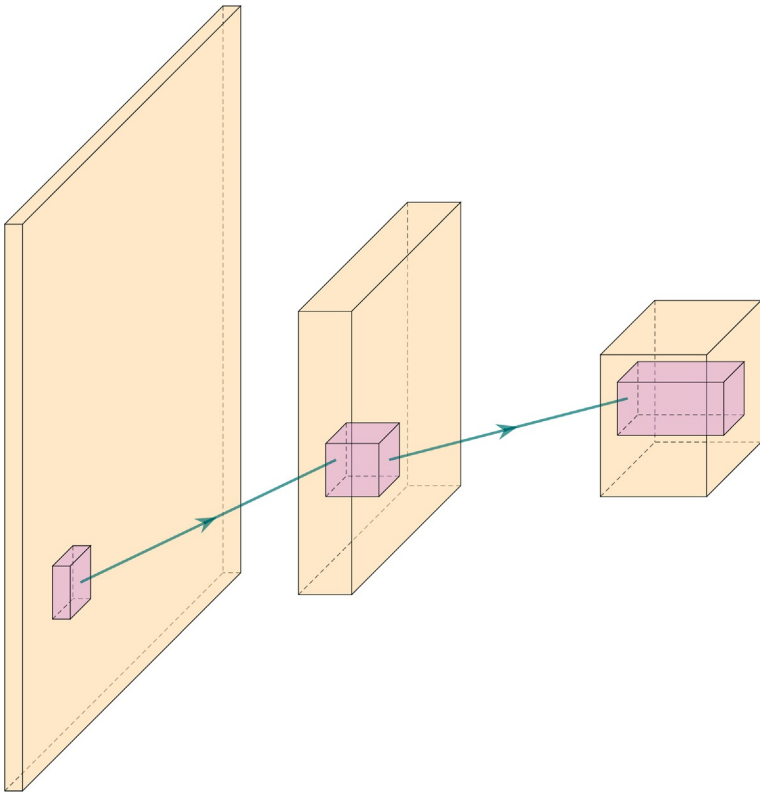
Fig. 10 Three layer convolutional network. The input image (*yellow*) is convolved with several filters or kernel matrices (*purple*). Commonly, the convolution is used to down-sample an image in the spatial dimension, while expanding the dimension of the filter response, hence expanding in "thickness" in the schematic. The filters are learned in the machine learning optimization loop. The shared weights within a filter improve efficiency of the network over classic dense networks.

This sequential example model of densely connected layers with a single input, 32, 16, and two neurons contains a total of 754 trainable weights. Initially, each of these weights is set to a pseudo-random value, which is often drawn from a distribution beneficial to fast training. Consequently, the data are passed through the network, and the result is numerically compared to the expected values. This form of training is defined as supervised training and error-correcting learning, which is a form of Hebbian learning. Other forms of learning exist and are employed in machine learning, e.g., competitive learning in self-organizing maps.

$$MAE = |y_j - o_j| \tag{10}$$

$$MSE = (y_j - o_j)^2 \tag{11}$$

In regression problems the error is often calculated using the mean absolute error (MAE) or mean squared error (MSE), the $\mathcal{L}_1$ shown in Eq. (10) and the $\mathcal{L}_2$ norm shown in Eq. (11), respectively. Classification problems form a special type of problem that can leverage a different kind of loss called cross-entropy (CE). The cross-entropy is dependent on the true label y and the prediction in the output layer.

$$CE = - \sum_j^C y_j \log\left(o_j\right) \tag{12}$$

Many machine learning data sets have one true label $y_{true} = 1$ for class $C_{j=true}$, leaving all other $y_j = 0$. This makes the sum over all labels obsolete. It is debatable how much binary labels reflects reality, but it simplifies Eq. (12) to minimizing the (negative) logarithm of the neural network output o_j, also known as negative log-likelihood:

$$CE = - \log\left(o_j\right) \tag{13}$$

Technically, the data we generated is a binary classification problem, and this means we could use the sigmoid activation function in the last layer and optimize a binary CE. This can speed up computation, but in this example, an approach is shown that works for many other problems and can therefore be applied to the readers data.

```
model.compile(optimizer='adam', # Often 'adam' or 'sgd' are good
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy']) # Monitor other metrics
```

Large neural networks can be extremely costly to train with significant developments in 2019/2020 reporting multibillion parameter language models (Google, OpenAI) trained on massive hardware infrastructure for weeks with a single epoch taking several hours. This calls for validation on unseen data after every epoch of the training run. Therefore, neural networks, like all machine learning models, are commonly trained with two hold-out sets, a validation and a final test set. The validation set can be provided or be defined as a percentage of the training data, as shown below. In the example, 10% of the training data are held out for validation after every epoch, reducing the training data set from 3750 to 3375 individual samples.

```
model.fit(X_train,
          y_train,
          validation_split=.1,
          epochs=100)
>>> [...]
    Epoch 100/100
    3375/3375 [==============================] - 0s 66us/sample
    loss: 0.1567 - accuracy: 0.9401 -
    val_loss: 0.1731 - val_accuracy: 0.9359
```

Neural networks are trained with variations of stochastic gradient descent (SGD), an incremental version of the classic steepest descent algorithm. We use the Adam optimizer, a variation of SGD that converges fast, but a full explanation would go beyond the scope of this chapter. The gist of the Adam optimizer is that it maintains a per-parameter learning rate of the first statistical moment (mean). This is beneficial for sparse problems and the second moment (uncentered variance), which is beneficial for noisy and nonstationary problems (Kingma & Ba, 2014). The main alternative to Adam is SGD with Nesterov momentum (Sutskever, Martens, Dahl, & Hinton, 2013), an optimization method that models conjugate gradient methods (CG) without the heavy computation that comes with the search in CG. SGD anecdotally finds a better optimal point for neural networks than Adam but converges much slower.

In addition to the loss value, we display the accuracy metric. While accuracy should not be the sole arbiter of model performance, it gives a reasonable initial estimate, how many samples are predicted correctly with a percentage between zero and one. As opposed to scikit-learn, deep learning models are compiled after their definition to make them fit for optimization on the available hardware. Then the neural network can be fit like the SVM and random forest models before, using the X_train and y_train data.

In addition, a number of epochs can be provided to run, as well as other parameters that are left on default for the example. The amount of epochs defines how many cycles of optimization on the full training data set are performed. Conventional wisdom for neural network training is that it should always learn for more epochs than machine learning researchers estimate initially.

It can be difficult to fix all sources of randomness and stochasticity in neural networks, to make both research and examples reproducible. This example does not fix these so-called random seeds as it would detract from the example. That implies that the results for loss and accuracy will differ from the printed examples. In research fixing the seed is very important to ensure reproducibility of claims. Moreover, to avoid bad practices or so-called lucky seeds, a statistical analysis of multiple fixed seeds is good practice to report results in any machine learning model.

```
model.evaluate(X_test, y_test)
>>> 1250/1250 [==============================] - 0s 93us/sample
    loss: 0.1998 - accuracy: 0.9360
    [0.19976349686831235, 0.936]
```

In the example before, the SVM and random forest classifier were scored on unseen data. This is equally important for neural networks. Neural networks are prone to overfit, which we try to circumvent by regularizing the weights and by evaluating the final network on an unseen test set. The prediction on the test set is very close to the last epoch in the training loop, which is a good indicator that this neural network generalizes to unseen data. Moreover, the loss curves in Fig. 11 do not converge too fast, while converging. However, it appears that the network would overfit if we let training continue. The exemplary decision boundary in Fig. 12 very closely models the local distribution of the data, which is true for the entire decision volume (Dramsch, 2020a).

These examples illustrate the open-source revolution in machine learning software. The consolidated API and utility functions make it seem trivial to apply various machine learning algorithms to scientific data. This can be seen in the recent explosion of publications of applied machine learning in geoscience. The need to be able to implement algorithms has been replaced by merely installing a package and calling `model.fit(X, y)`. These developments call for strong validation requirements of models to ensure valid, reproducible, and scientific results. Without this careful validation these modern day tools can be severely misused to oversell results and even come to incorrect conclusions.

In aggregate, modern day neural networks benefit from the development of nonsaturating nonlinear activation functions. The advancements of stochastic gradient descent with Nesterov momentum and the Adam
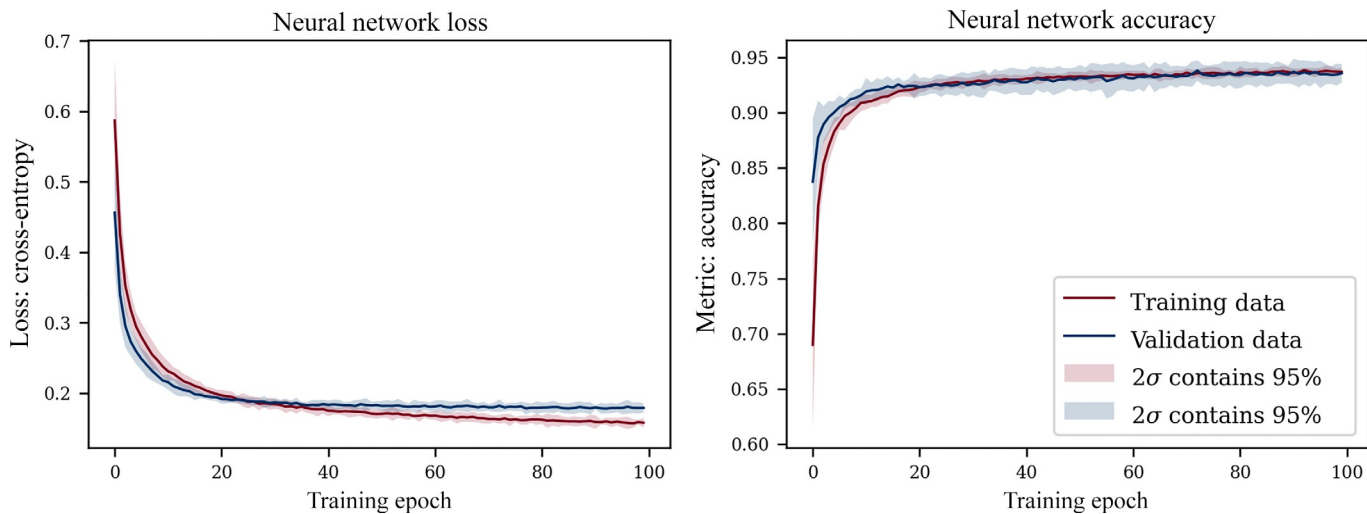
Fig. 11 Loss and accuracy of example neural network on ten random initializations. Training for 100 epochs with the shaded area showing the 95% confidence intervals of the loss and metric. Analyzing loss curves is important to evaluate overfitting. The training loss decreasing, while validation loss is close to plateauing is a sign of overfitting. Generally, it can be seen that the model converged and is only making marginal gains with the risk of overfitting.
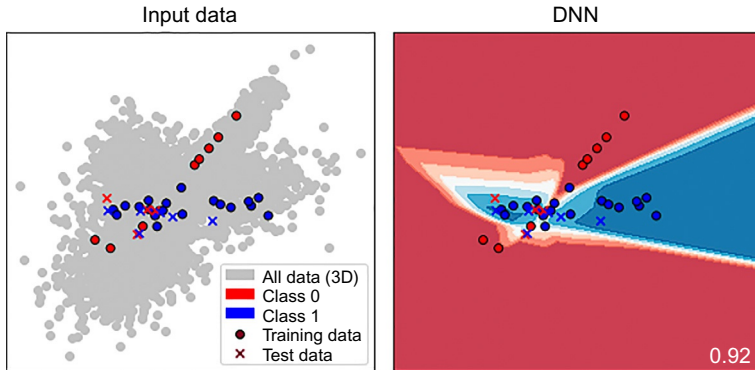
Fig. 12 Central 2D slice of decision boundary of deep neural network in trained on data with three informative features. The 3D volume is available in Dramsch (2020a).

optimizer (following AdaGrad and RMSProp) was essential faster training of deep neural networks. The leverage of graphics hardware available in most high-end desktop computers that is specialized for linear algebra computation, further reduced training times. Finally, open-source software that is well-maintained, tested, and documented with a consistent API made both shallow and deep machine learning accessible to nonexperts.

2.5 Neural network architectures

In deep learning, implementation of models is commonly more complicated than understanding the underlying algorithm. Modern deep learning makes use of various recent developments that can be beneficial to the data set it is applied to, without specific implementation details results are often not reproducible. However, the machine learning community has a firm grounding in openness and sharing, which is seen in both publications and code. New developments are commonly published alongside their open-source code, and frequently with the trained networks on standard benchmark data sets. This facilitates thorough inspection and transferring the new insights to applied tasks such as geoscience. In the following, some relevant neural network architectures and their application are explored.

2.6 Convolutional neural network architectures

The first model to discuss is the VGG-16 model, a 16-layer deep convolutional neural network (Simonyan & Zisserman, 2014) represented in Fig. 13[c]. This network was an attempt at building even deeper networks

---
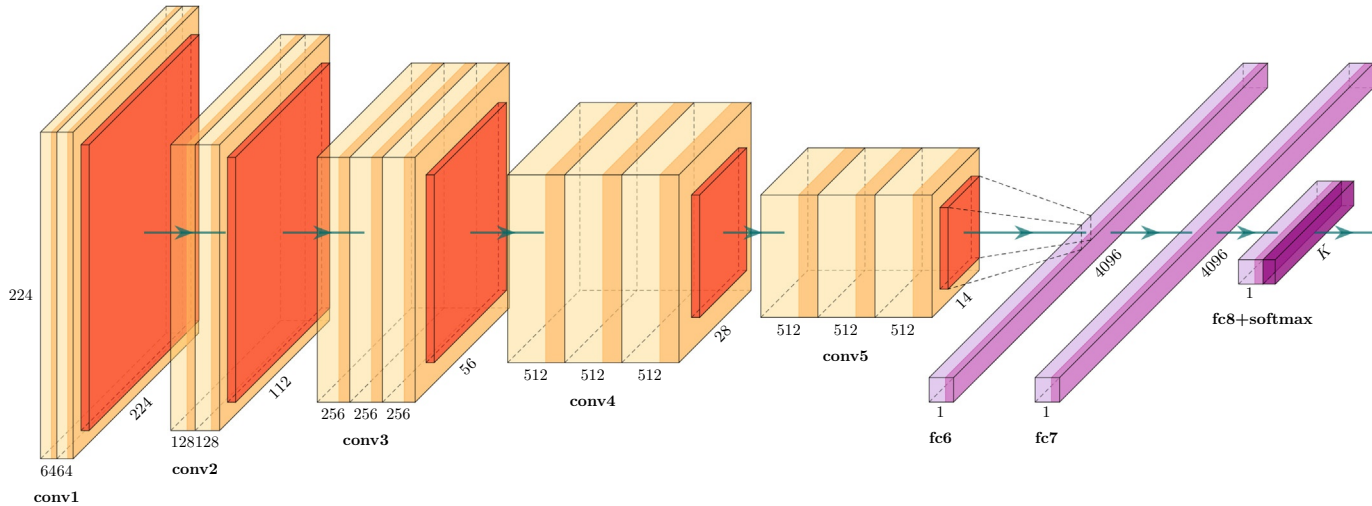[c] Visualization library (Iqbal, 2018).

Fig. 13 Schematic of a VGG16 network for ImageNet. The input data are convolved and down-sampled repeatedly. The final image classification is performed by flattening the image and feeding it to a classic feed-forward densely connected neural network. The 1000 output nodes for the 1000 ImageNet classes are normalized by a final softmax layer (cf. Eq. 8).

and uses small 3×3 convolutional filters in the network, called f in Eq. (9). This small filter-size was sufficient to build powerful models that abstract the information from layer to deeper layer, which is easy to visualize and generalize well. The trained model on natural images also transfers well to other domains like seismic interpretation (Dramsch & Lüthje, 2018). Later, the concept of Network-in-Network was introduced, which suggested defined subnetworks or blocks in the larger network structure (Lin, Chen, & Yan, 2013). The ResNet architecture uses this concept of blocks to define residual blocks. These use a shortcut around a convolutional block (He, Zhang, Ren, & Sun, 2016) to achieve neural networks with up to 152 layers that still generalize well. ResNets and residual blocks, in particular, are very popular in modern architectures including the shortcuts or skip connections they popularized, to address the following problem:

> *When deeper networks start converging, a degradation problem has been exposed: with the network depth increasing, accuracy gets saturated (which might be unsurprising) and then degrades rapidly. Unexpectedly, such degradation is not caused by overfitting, and adding more layers to a suitably deep model leads to higher training error.*
>
> **He et al. (2016)**

The developments and successes in image classification on benchmark competitions like ImageNet and Pascal-VOC inspired applications in automatic seismic interpretation. These networks are usually single image classifiers using convolutional neural networks (CNNs). The first application of a convolutional neural network to seismic data used a relatively small deep CNN for salt identification (Waldeland & Solberg, 2017). The open-source software "MaLenoV" implemented a single image classification network, which was the earliest freely available implementation of deep learning for seismic interpretation (Ildstad & Bormann, n.d.). Dramsch and Lüthje (2018) applied pretrained VGG-16 and ResNet50 single image seismic interpretation. Recent successful applications build upon pretrained prebuilt architectures to implement into more sophisticated deep learning systems, e.g., semantic segmentation. Semantic segmentation is important in seismic interpretation. This is already a narrow field of application of machine learning and it can be observed that many early applications focus on subsections of seismic interpretation utilizing these prebuilt architectures such as salt detection (Di, Wang, & AlRegib, 2018; Gramstad & Nickel, 2018; Waldeland, Jensen, Gelius, & Solberg, 2018), fault interpretation (Araya-Polo et al., 2017; Guitton, 2018; Purves, Alaei, & Larsen, 2018), facies classification (Chevitarese, Szwarcman, Silva, & Brazil, 2018; Dramsch & Lüthje, 2018),

and horizon picking (Wu & Zhang, 2018). In comparison, this is however, already a broader application than prior machine learning approaches for seismic interpretation that utilized very specific seismic attributes as input to self-organizing maps (SOM) for, e.g., sweet spot identification (Guo, Zhang, Lin, & Liu, 2017; Roden & Chen, 2017; Zhao, Li, & Marfurt, 2017).

In geoscience single image classification, as presented in the ImageNet challenge, is less relevant than other applications like image segmentation and time series classification. The developments and insights resulting from the ImageNet challenge were, however, transferred to network architectures that have relevance in machine learning for science. Fully convolutional networks are a way to better achieve image segmentation. A particularly useful implementation, the U-net, was first introduced in biomedical image segmentation, a discipline notorious for small datasets (Ronneberger, Fischer, & Brox, 2015). The U-net architecture shown in Fig. 14 utilizes several shortcuts in an encoder-decoder architecture to achieve stable segmentation results. Shortcuts (or skip connections) are a way in neural networks to combine the original information and the processed information, usually through concatenation or addition. In ResNet blocks this concept is extended to an extreme, where every block in the architecture contains a shortcut between the input and output, as seen in Fig. 15. These blocks are universally used in many architectures to implement deeper networks, i.e., ResNet-152 with 60 million parameters, with fewer parameters than previous architectures like VGG-16 with 138 million parameters. Essentially, enabling models that are ten times as deep with less than half the parameters, and significantly better accuracy on image benchmark problems.

In 2018 the seismic contractor TGS made a seismic interpretation challenge available on the data science competition platform Kaggle. Successful participants in the competition combined ResNet architectures with the Unet architecture as their base architecture and modified these with state-of-the-art image segmentation applications (Babakhin, Sanakoyeu, & Kitamura, 2019). Moreover, Dramsch and Lüthje (2018) showed that transferring networks trained on large bodies of natural images to seismic data yields good results on small datasets, which was further confirmed in this competition. The learnings from the TGS Salt Identification challenge have been incorporated in production scale models that perform human-like salt interpretation (Sen, Kainkaryam, Ong, & Sharma, 2020). In broader geoscience, U-nets have been used to model global water storage using GRAVE satellite data (Sun et al., 2019), landslide prediction

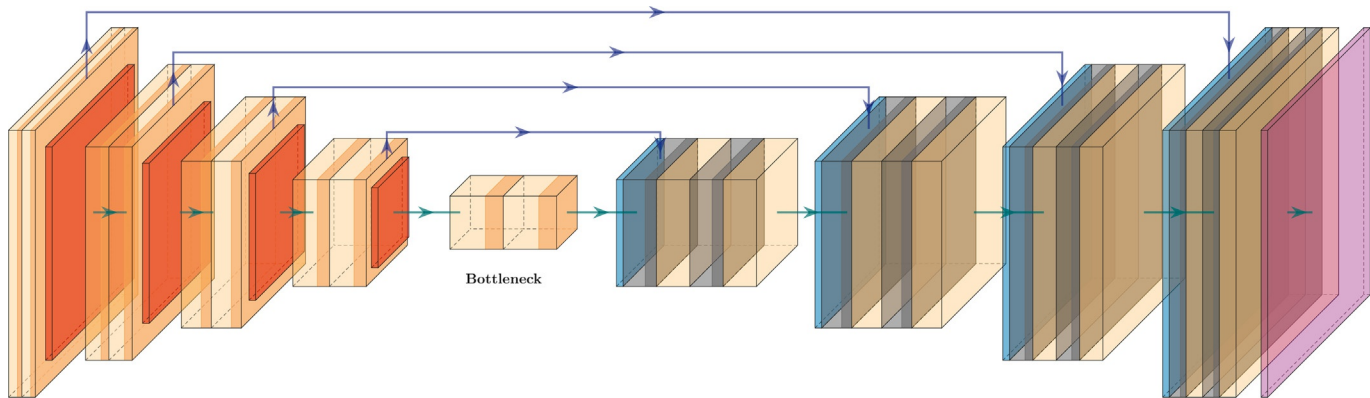**Fig. 14** Schematic of Unet architecture. Convolutional layers are followed by a down-sampling operation in the encoder. The central bottleneck contains a compressed representation of the input data. The decoder contains upsampling operations followed by convolutions. The last layer is commonly a softmax layer to provide classes. Equally sized layers are connected via shortcut connections.
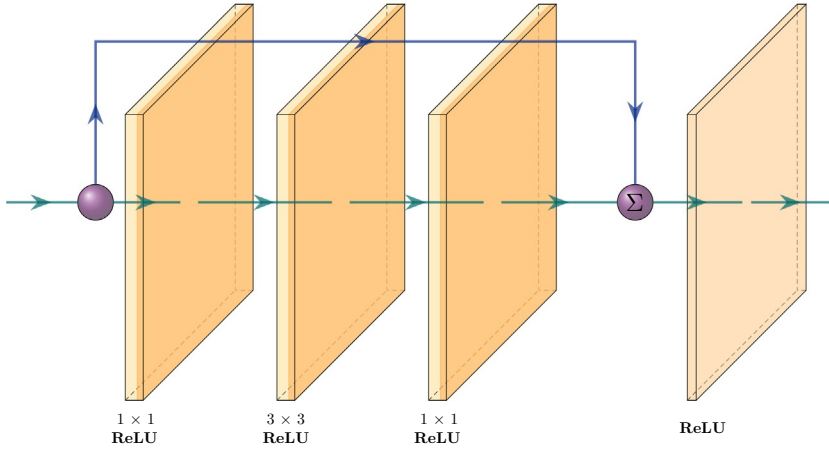
Bottleneck

1×1
ReLU
3×3
ReLU
1×1
ReLU
ReLU

Fig. 15 Schematic of a ResNet block. The block contains a 1×1, 3×3, and 1×1 convolution with ReLU activation. The output is concatenated with the input and passed through another ReLU activation function.

(Hajimoradlou, Roberti, & Poole, 2019), and earthquake arrival time picking (Zhu & Beroza, 2018). A more classical approach identifies subsea scale worms in hydrothermal vents (Shashidhara, Scott, & Marburg, 2020), whereas Dramsch, Christensen, MacBeth, & Lüthje (2019) includes a U-net in a larger system for unsupervised 3D timeshift extraction from 4D seismic.

This modularity of neural networks can be seen all throughout the research and application of deep learning. New insights can be incorporated into existing architectures to enhance their predictive power. This can be in the form of swapping out the activation function σ or including new layers for improvements, e.g., regularization with batch normalization (Ioffe & Szegedy, 2015). The U-net architecture originally is relatively shallow, but was modified to contain a modified ResNet for the Kaggle salt identification challenge instead (Babakhin et al., 2019). Overall, serving as examples for the flexibility of neural networks.

2.7 Generative adversarial networks

Generative adversarial networks (GAN) take composition of neural network to another level, where two networks are trained in aggregate to get a desired result. In GANs, a generator network G and a discriminator network D work against each other in the training loop (Goodfellow et al., 2014).

The generator G is set up to generate samples from an input, these were often natural images in early GANs, but has now progressed to anything from time series (Engel et al., 2019) to high-energy physics simulation (Paganini, de Oliveira, & Nachman, 2018). The discriminator network D attempts to distinguish whether the sample is generated from G, i.e., fake or a real image from the training data. Mathematically, this defines a min max game for the value function V of G and D

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))], \tag{14}$$

with x representing the data, z is the latent space G draws samples from, and p represents the respective probability distributions. Eventually reaching a Nash equilibrium (Nash, 1951), where neither the generator network G can produce better outputs, nor the discriminator network D can improve its capability to discern between fake and real samples.

Despite how versatile U-nets are, they still need an appropriate defined loss function and labels to build a discriminative model. GANs, however, build a generative model that approximates the training sample distribution in the Generator and a discriminative model of the discriminator modeled dynamically through adversarial training. The discriminator effectively providing an adversarial loss in a GAN. In addition to providing two models that serve different purposes, learning the training sample distribution with an adversarial loss makes GANs one of the most versatile models currently discovered. Mosser, Dubrule, and Blunt (2017) were applied GANs early on to geoscience, modeling 3D porous media at the pore scale with a deep convolutional GAN. The authors extended this approach to conditional simulations of oolithic digital rock (Mosser, Dubrule, & Blunt, 2018 a). Early applications of GANs also included approximating the problem of velocity inversion of seismic data (Mosser, Kimman, et al., 2018), geostatistical inversion (Laloy, Hérault, Jacques, & Linde, 2017), and generating seismograms (Krischer & Fichtner, 2017). Richardson (2018) integrate the generator of the GAN into full waveform inversion of the scalar wavefield. Alternatively, a Bayesian inversion using the generator as prior for velocity inversion was introduced in Mosser, Dubrule, and Blunt (2018 b). In geomodeling, generation of geological channel models was presented (Chan & Elsheikh, 2017), which was subsequently extended with the capability to be conditioned on physical measurements (Dupont, Zhang, Tilke, Liang, & Bailey, 2018).

Naturally, GANs were applied to the growing field of automatic seismic interpretation (Lu, Morris, Brazell, Comiskey, & Xiao, 2018).

2.8 Recurrent neural network architectures

The final type of architecture applied in geoscience is recurrent neural networks (RNN). In contrast to all previous architectures, recurrent neural networks feed back into themselves. There are many types of RNNs, Hopfield networks being one that were applied to seismic source wavelet prediction (Wang & Mendel, 1992) early on. However, LSTMs (Hochreiter & Schmidhuber, 1997) are the main application in geoscience and wider machine learning. This type of network achieves state-of-the-art performance on sequential data like language tasks and time series applications. LSTMs solve some common problems of RNNs by implementing specific gates that regulate information flow in an LSTM cell, namely, input gate, forget gate, and output gate, visualized in Fig. 16. The input gate feeds input values to the internal cell. The forget gate overwrites the previous state. Finally, the output gate regulates the direct contribution of the input value to the output
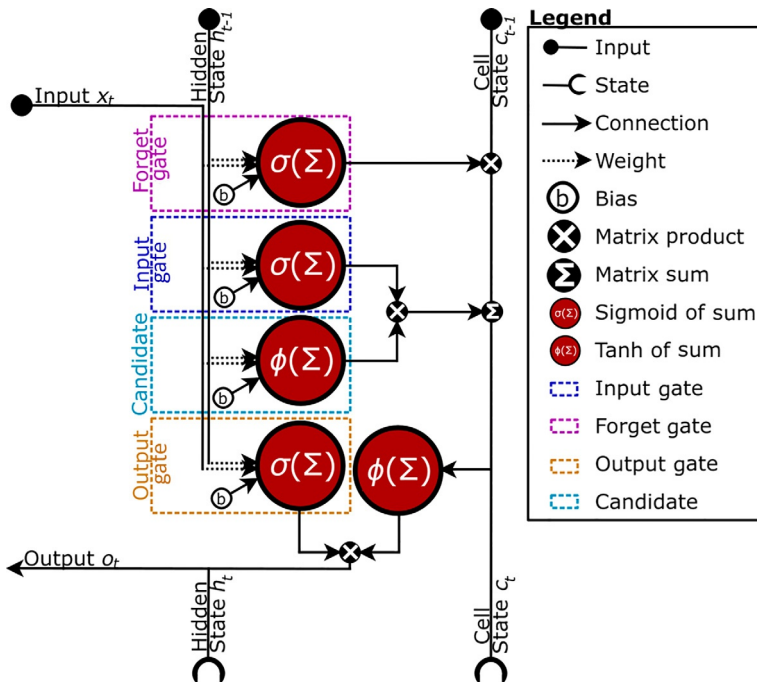

Fig. 16 Schematic of LSTM architecture. The input data are processed together with the hidden state and cell state. The LSTM avoid the exploding gradient problem by implemented a input, forget, and output gate.

value combined with the internal state of the cell. Additionally, a peephole functionality helps with the training that serves as a shortcut between inputs and gates.

A classic application of LSTMs is text analysis and natural language understanding, which has been applied to geological relation extraction from unstructured text documents (Blondelle, Juneja, Micaelli, & Neri, 2017; Luo, Zhou, Wang, Zhu, & Deng, 2017). Due to the nature of LSTMs being suited for time series data, it is has been applied to seismological event classification of volcanic activity Titos, Bueno, García, Benítez, and Ibañez (2018), multifactor landslide displacement prediction (Xie, Zhou, & Chai, 2019), and hydrological modeling (Kratzert et al., 2019). Talarico, Leão, and Grana (2019) applied LSTM to model sedimentological sequences and compared the model to baseline hidden Markov model (HMM), concluding that RNNs outperform HMMs based on first-order Markov chains, while higher order Markov chains were too complex to calibrate satisfactorily. Gated recurrent unit (GRU) (Cho et al., 2014) is another RNN developed based on the insights into LSTM, which was applied to predict petrophysical properties from seismic data (Alfarraj & AlRegib, 2018).

The scope of this review only allowed for a broad overview of types of networks that were successfully applied to geoscience. Many more specific architectures exist and are in development that provide different advantages. Siamese networks for one-shot image analysis (Koch, Zemel, & Salakhutdinov, 2015), transformer networks that largely replaced LSTM and GRU in language modeling (Vaswani et al., 2017), or attention as a general mechanism in deep neural networks (Zheng, Fu, Mei, & Luo, 2017).

Neural network architectures have been modified and applied to diverse problems in geoscience. Every architecture type is particularly suited to certain data types that are present in each field of geoscience. However, fields with data present in machine-readable format experienced accelerated adoption of machine learning tools and applications. For example, Ross, Meier, and Hauksson (2018) were able to successfully apply CNNs to seismological phase detection, relying on an extensive catalogue of hand-picked data (Ross, Meier, Hauksson, & Heaton, 2018a) and consequently generalize this work (Ross, Meier, Hauksson, & Heaton, 2018b). It has to be noted that synthetic or specifically sampled data can introduce an implicit bias into the network (Kim, Kim, Kim, Kim, & Kim, 2019; Wirgin, 2004). Nevertheless, particularly this blackbox property of machine learning model makes them versatile and powerful tools that were leveraged in every subdiscipline of the Earth sciences.

2.9 The state of ML on geoscience

Overall, geoscience and especially geophysics has followed developments in machine learning closely. Across disciplines, machine learning methods have been applied to various problems that can generally be categorized into three sections:

1. Build a surrogate ML model of a well-understood process. This model usually provides an advantage in computational cost.
2. Build an ML model of a task previously only possible with human interaction, interpretation, or knowledge and experience.
3. Build a novel ML model that performs a task that was previously not possible.

Granulometry on SEM images is an example of an application in category I, where previously sediments were hand-measured in images (Dramsch, Amour, & Lüthje, 2018). Applying large deformation diffeomorphic mapping of seismic data was computationally infeasible for matching 4D seismic data, however, made feasible by applying a U-net architecture to the problem of category II (Dramsch et al., 2019). The problem of earthquake magnitude prediction falls into category III due to the complexity of the system but was nevertheless approached with neural networks (Panakkat & Adeli, 2007).

The accessibility of tools, knowledge, and compute make this cycle of machine learning enthusiasm unique, with regard to previous decades. This unprecedented access to tools makes the application of machine learning algorithms to any problem possible, where data is available. The bibliometrics of machine learning in geoscience, shown in Fig. 17 serve as a proxy for increased access. These papers include varying degrees of depth in application and model validation. One of the primary influences for the current increase in publications are new fields such as automatic seismic interpretation, as well as, publications soliciting and encouraging machine learning publications. Computer vision models were relatively straight forward to transfer to seismic interpretation tasks, with papers in this subfield ranging from single 2D line salt identification models with limited validation to 3D multifacies interpretation with validation on a separate geographic area.

Geoscientific publishing can be challenging to navigate with respect to machine learning. While papers investigating the theoretical fundamentals of machine learning in geoscience exist, it is clear that the overwhelming majority of papers present applications of ML to geoscientific problems. It is complex to evaluate whether a paper is a case study or a methodological paper with an exemplary application to a specific data set. Despite the difficulty

Machine learning papers in geoscience
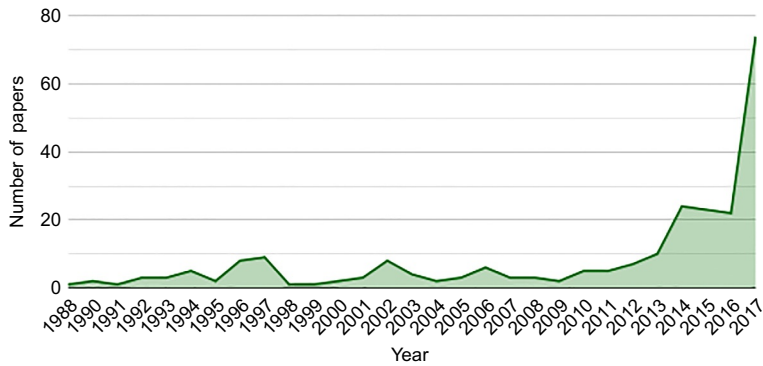A sample of 242 papers published before 2018

Fig. 17 Bibliometry of 242 papers in machine learning for geoscience per year. Search terms include variations of machine learning terms and geoscientific subdisciplines but exclude remote sensing and kriging.

of most thorough applications of ML, "idea papers" exist that simply present an established algorithm to a problem in geoscience without a specific implementation or addressing the possible caveats. On the flip-side, some papers apply machine learning algorithms as pure regression models without the aim to generalize the model to other data. Unfortunately, this makes meta-analysis articles difficult to impossible. This kind of meta-analysis article, is commonly done in medicine and considered a gold-standard study, and would greatly benefit the geoscientific community to determine the efficacy of algorithms on sets of similar problems.

Analogous to the medical field, obtaining accurate ground truth data, is often impossible and usually expensive. Geological ground truth data for seismic data is usually obtained through expert interpreters. Quantifying the uncertainty of these interpretations is an active field of research, which suggest a broader set of experiences and a diverse set of sources of information for interpretation facilitate correct geological interpretation between interpreters (Bond, Gibbs, Shipton, Jones, et al., 2007). Radiologists tasked to interpret X-ray images showed similar decreases in both inter- and intra-interpreter error rate with more diverse data sources Jewett et al. (1992). These uncertainties in the training labels are commonly known as "label noise" and can be a detriment to building accurate and generalizable machine learning models. A significant portion of data in geoscience, however, is not machine learning ready. Actual ground truth data from drilling reports is often locked away in running text reports, sometimes in

scanned PDFs. Data are often siloed and most likely proprietary. Sometimes the amount of samples to process is so large that many insights are yet to be made from samples in core stores or the storage rooms of museums. Benchmark models are either nonexistent or made by consortia that only provide access to their members. Academic data are usually only available within academic groups for competitive advantage, respect for the amount of work, and fear of being exposed to legal and public repercussions. These problems are currently addressed by a culture change. Nevertheless, liberating data will be a significant investment, regardless of who will work on it and a slow culture change can be observed already.

Generally, machine learning has seen the fastest successes in domains where decisions are cheap (e.g., click advertising), data are readily available (e.g., online shops), and the environment is simple (e.g., games) or unconstrained (e.g., image generation). Geoscience generally is at the opposite of this spectrum. Decisions are expensive, be it drilling new wells or assessing geohazards. Data are expensive, sparse, and noisy. The environment is heterogeneous and constrained by physical limitations. Therefore, solving problems like automatic seismic interpretation see a surge of activity having fewer constraints initially. Problems like inversion have solutions that are verifiably wrong due to physics. These constraints do not prohibit machine learning applications in geoscience. However, most successes are seen in close collaboration with subject matter experts. Moreover, model explainability becomes essential in the geoscience domain. While not being a strict equivalency, simpler models are usually easier to interpret, especially regarding failure modes.

A prominent example of "excessive" (Mignan & Broccardo, 2019a) model complexity was presented in DeVries, Viégas, Wattenberg, and Meade (2018) applying deep learning to aftershock prediction. Independent data scientists identified methodological errors, including data leakage from the train set to the test set used to present results (Shah & Innig, 2019). Moreover, Mignan and Broccardo (2019b) showed that using the central physical interpretation of the deep learning model, using the von Mises yield criterion, could be used to build a surrogate logistic regression. The resulting surrogate or baseline model outperforms the deep network and overfits less. Moreover, replacing the 13,000 parameter model with the two-parameter baseline model increases calculation speed, which is essential in aftershock forecasting and disaster response.[d] More generally, this is an example where data science

---

[d] All authors point out the potential in deep and machine learning research in geoscience regardless and do not wish to stifle such research (Mignan & Broccardo, 2019 b; Shah & Innig, 2019).

practices such as model validation, baseline models, and preventing data leakage and overfitting become increasingly important when the tools of applying machine learning become readily available.

Despite potential setbacks and the field of deep learning and data science being relatively young, they can rely on mathematical and statistical foundations and make significant contributions to science and society. Machine learning systems have contributed to modeling the protein structure of the current pandemic virus COVID-19 (Jumper, Tunyasuvunakool, Kohli, Hassabis, & Team, n.d.). A deep learning computer vision system was built to stabilize food safety by identifying Cassava plant disease on offline mobile devices (Ramcharan et al., 2017, 2019). Self-driving cars have become a possibility (Bojarski et al., 2016) and natural language understanding has progressed significantly (Devlin, Chang, Lee, & Toutanova, 2018).

Geoscience is slower in the adoption of machine learning, compared to other disciplines. To be able to adapt the progress in machine learning research, many valuable data sources have to be made machine-readable. There has already been a change in making computer code open-source, which has led to collaborations and accelerating scientific progress. While specific open benchmark data sets have been tantamount to the progress in machine learning, it is questionable whether these would be beneficial to machine learning in geoscience. The problems are often very complex with nonunique explanations and solutions, which historically has led to disagreements over geophysical benchmark data sets. Open data and open-source software, however, have and will play a significant role in advancing the field. Examples of this include basic utility function to load geoscientific data (Kvalsvik & Contributors, 2019) or more specifically cross-validation functions tailored to geoscience (Uieda, 2018).

Moreover, machine learning is fundamentally conservative, training on available data. This bias of data collection will influence the ability to generate new insights in all areas of geoscience. Machine learning in geoscience may be able to generate insights and establish relationships in existing data. Entirely new insights from previously unseen or analysis of particularly complex models will still be a task performed by trained geoscientists. Transfer learning is an active field of machine learning research that geoscience can significantly benefit from. However, no significant headway has been made to transfer trained machine learning models to out-of-distribution data, i.e., data that are conceptually similar but explicitly different from the training data set. The fields of self-supervised learning, including reinforcement learning

that can learn by exploration, may be able to approach some of these problems. They are, however, notoriously hard to set up and train, necessitating significant expertise in machine learning.

Large portions of publications are concerned with weakly or unconstrained predictions such as seismic interpretation and other applications that perform image recognition on SEM or core photography. These methods will continue to improve by implementing algorithmic improvements from machine learning research, specialized data augmentation strategies, and more diverse training data being available. New techniques such as multitask learning (Kendall, Gal, & Cipolla, 2018) which improved computer vision and computer linguistic models, deep Bayesian networks (Mosser, Oliveira, & Steventon, 2019) to obtain uncertainties, noisy teacher–student networks (Xie, Hovy, et al., 2019) to improve training, and transformer networks (Graves, 2012) for time series processing, will significantly improve applications in geoscience. For example, automated seismic interpretation may advance to provide reliable outputs for relatively difficult geological regimes beyond existing solutions. Success will be reliant on interdisciplinary teams that can discern why geologically specific faults are important to interpret, while others would be ignored in manual interpretations, to encode geological understanding in automatic interpretation systems.

Currently, the most successful applications of machine learning and deep learning, tie into existing workflows to automate subtasks in a grander system. These models are highly specific, and their predictive capability does not resemble an artificial intelligence or attempt to do so. Mathematical constraints and existing theory in other applied fields, especially neuroscience, were able to generate insights into deep learning and geoscience has the opportunity to develop significant contributions to the area of machine learning, considering their unique problem set of heterogeneity, varying scales and nonunique solutions. This has already taken place with the wider adoption of "kriging" or more generally Gaussian processes into machine learning. Moreover, known applications of signal theory and information theory employed in geophysics are equally applicable in machine learning, with examples utilizing complex-valued neural networks (Trabelsi et al., 2017), deep Kalman filters (Krishnan, Shalit, & Sontag, 2015), and Fourier analysis (Tancik et al., 2020). Therefore, possibly enabling additional insights, particularly when integrated with deep learning, due to its modularity and versatility.

Previous reservations about neural networks included the difficulty of implementation and susceptibility to noise in addition to computational

costs. Research into updating trained models and saving the optimizer state with the model has in part alleviated the cost of retraining existing models. Moreover, fine-tuning pretrained large complex models to specific problems has proven successful in several domains. Regularization techniques and noise modeling, as well as data cleaning pipelines, can be implemented to lessen the impact of noise on machine learning models. Specific types of noise can be attenuated or even used as an additional source of information. The aforementioned concerns have mainly transitioned into a critique about overly complex models that overfit the training data and are not interpretable. Modern software makes very sophisticated machine learning models, and data pipelines available to researchers, which has, in turn, increased the importance to control for data leakage and perform thorough model validation.

Currently, machine learning for science primarily relies on the emerging field of explainability (Lundberg et al., 2018). These provide primarily post hoc explanations for predictions from models. This field is particularly important to evaluate which inputs from the data have the strongest influence on the prediction result. The major point of critique regarding post hoc explanations is that these methods attempt to explain how the algorithm reached a wrong prediction with equal confidence. Bayesian neural networks intend to address this issue by providing confidence intervals for the prediction based on prior beliefs. These neural networks intend to incorporate prior expert knowledge into neural networks, which can be beneficial in geoscientific applications, where strong priors can be necessary. Machine learning interpretability attempts to impose constraints on the machine learning models to make the model itself explainable. Closely related to these topics is the statistics field of causal inference. Causal inference attempts to model the cause of variable, instead of correlative prediction. Some methods exist that can perform causal machine learning, i.e., causal trees (Athey & Imbens, 2016). These three fields will be necessary to glean verifiable scientific insights from machine learning in geoscience. They are active fields of research and more involved to correctly apply, which often makes cooperation with a statistician necessary.

In conclusion, machine learning has had a long history in geoscience. Kriging has progressed into more general machine learning methods, and geoscience has made significant progress applying deep learning. Applying deep convolutional networks to automatic seismic interpretation has progressed these methods beyond what was possible, albeit still being an active field of research. Using modern tools, composing custom neural networks, and conventional machine learning pipelines has become

increasingly trivial, enabling wide-spread applications in every subfield of geoscience. Nevertheless, it is important to acknowledge the limitations of machine learning in geoscience. Machine learning methods are often cutting edge technology, yet properly validated models take time to develop, which is often perceived as inconvenient when working in a hot scientific field. Despite being cutting edge, it is important to acknowledge that none of these applications are fully automated, as would be suggested by the lure of artificial intelligence. Nevertheless, within applied geoscience, significant new insights have been presented. Applications in geoscience are using machine learning as a utility for data preprocessing, implementing previous insights beyond the theory and synthetic cases, or the model itself enabling unprecedented applications in geoscience. Overall, applied machine learning has matured into an established tool in computational geoscience and has the potential to provide further insights into the theory of geoscience itself.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., … Zheng, X. (2015). In *TensorFlow: Large-scale machine learning on heterogeneous systems*. http://tensorflow.org/ Software available from tensorflow.org.

Agterberg, F. P. (1966). Markov schemes for multivariate well data. In *Proceedings, symposium on applications of computers and operations research in the mineral industries, Pennsylvania State University, State College, Pennsylvania: Vol. 2.* (pp. X1–X18).

Aizerman, M. A. (1964). Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, *25*, 821–837.

Alfarraj, M., & AlRegib, G. (2018). Petrophysical property estimation from seismic data using recurrent neural networks. *SEG Technical Program Expanded Abstracts 2018*, 2141–2146.

Anifowose, F., Ayadiuno, C., & Rashedian, F. (2017). Carbonate reservoir cementation factor modeling using wireline logs and artificial intelligence methodology. In *79th EAGE conference and exhibition 2017-workshops*.

Araya-Polo, M., Dahlke, T., Frogner, C., Zhang, C., Poggio, T., & Hohl, D. (2017). Automated fault detection without seismic processing. *The Leading Edge*, *36*(3), 208–214.

Athey, S., & Imbens, G. (2016). Recursive partitioning for heterogeneous causal effects. *Proceedings of the National Academy of Sciences*, *113*(27), 7353–7360.

Babakhin, Y., Sanakoyeu, A., & Kitamura, H. (2019). Semi-supervised segmentation of salt bodies in seismic images using an ensemble of convolutional neural networks. In *German conference on pattern recognition (GCPR)*.

Ballabio, C., & Sterlacchini, S. (2012). Support vector machines for landslide susceptibility mapping: The Staffora River Basin Case Study, Italy. *Mathematical Geosciences*, *44*(1), 47–70. https://doi.org/10.1007/s11004-011-9379-9.

Bayes, T. (1763). LII. An essay towards solving a problem in the doctrine of chances. By the late Rev. Mr. Bayes, FRS communicated by Mr. Price, in a letter to John Canton, AMFR S. *Philosophical Transactions of the Royal Society of London*, *53*, 370–418.

Belson, W. A. (1959). Matching and prediction on the principle of biological classification. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, *8*(2), 65–75.

Bergen, K. J., Johnson, P. A., Maarten, V., & Beroza, G. C. (2019). Machine learning for data-driven discovery in solid earth geoscience. *Science*, *363*(6433), eaau0323.

Bestagini, P., Lipari, V., & Tubaro, S. (2017). A machine learning approach to facies classification using well logs. In *SEG technical program expanded abstracts 2017*. Society of Exploration Geophysicists, pp. 2137–2142. https://doi.org/10.1190/segam2017-17729805.1.

Beyreuther, M., & Wassermann, J. (2008). Continuous earthquake detection and classification using discrete Hidden Markov models. *Geophysical Journal International*, *175*(3), 1055–1066. https://doi.org/10.1111/j.1365-246X.2008.03921.x.

Bicego, M., Acosta-Muñoz, C., & Orozco-Alzate, M. (2013). Classification of seismic volcanic signals using Hidden–Markov-model-based generative embeddings. *IEEE Transactions on Geoscience and Remote Sensing*, *51*(6), 3400–3409. https://doi.org/10.1109/TGRS.2012.2220370.

Blondelle, H., Juneja, A., Micaelli, J., & Neri, P. (2017). *Machine learning can extract the information needed for modelling and data analysing from unstructured documents*. In *79th EAGE conference and exhibition 2017-workshops*. earthdoc.org. http://www.earthdoc.org/publication/publicationdetails/?publication=89273.

Blouin, M., Caté, A., Perozzi, L., & Gloaguen, E. (2017). *Automated facies prediction in drillholes using machine learning*. In *79th EAGE conference and exhibition 2017-workshops*. earthdoc.org. http://www.earthdoc.org/publication/publicationdetails/?publication=89276.

Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal , P., et al. (2016). End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*.

Bond, C. E., Gibbs, A. D., Shipton, Z. K., & Jones, S. (2007). What do you think this is? "Conceptual uncertainty" in geoscience interpretation. *GSA Today*, *17*(11), 4.

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5–32.

Bryson, A. E. (1961). A gradient method for optimizing multi-stage allocation processes. In *Proc Harvard Univ symposium on digital computers and their applications*, Vol. 72.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., … Varoquaux, G. (2013). API design for machine learning software: Experiences from the scikit-learn project. In *ECML PKDD workshop: Languages for data mining and machine learning*, pp. 108–122.

Cao, J., & Roy, B. (2017). Time-lapse reservoir property change estimation from seismic using machine learning. *The Leading Edge*, *36*(3), 234–238. https://doi.org/10.1190/tle36030234.1.

Caté, A., Perozzi, L., Gloaguen, E., & Blouin, M. (2017). Machine learning as a tool for geologists. *The Leading Edge*, *36*(3), 215–219. https://doi.org/10.1190/tle36030215.1.

Caté, A., Schetselaar, E., Mercier-Langevin, P., & Ross, P.-S. (2018). Classification of lithostratigraphic and alteration units from drillhole lithogeochemical data using machine learning: A case study from the Lalor volcanogenic massive sulphide deposit, Snow Lake, Manitoba, Canada. *Journal of Geochemical Exploration*, *188*, 216–228. https://www.sciencedirect.com/science/article/pii/S0375674217305083.

Chaki, S., Routray, A., & Mohanty, W. K. (2018). Well-Log and seismic data integration for reservoir characterization: A signal processing and machine-learning perspective. *IEEE Signal Processing Magazine*, *35*(2), 72–81. https://doi.org/10.1109/MSP.2017.2776602.

Chan, S., & Elsheikh, A. H. (2017). Parametrization and generation of geological models with generative adversarial networks. *arXiv preprint arXiv:1708.01810*.

Chang, C.-C., & Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, *2*(3), 27:1–27:27. https://doi.org/10.1145/1961189.1961199.

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. New York, NY, USA: ACM, pp. 785–794. https://doi.org/10.1145/2939672.2939785.

Chevitarese, D., Szwarcman, D., Silva, R. M. D., & Brazil, E. V. (2018). Seismic facies segmentation using deep learning. In *ACE 2018 annual convention & exhibition*. searchanddiscovery.com.

Chiles, J. P., & Chauvet, P. (1975). Kriging: A method for cartography of the sea floor. *The International Hydrographic Review*.

Chilès, J.-P., & Desassis, N. (2018). Fifty years of kriging. In *Handbook of mathematical geosciences*. Springer, pp. 589–612.

Ching, T., Himmelstein, D. S., Beaulieu-Jones, B. K., Kalinin, A. A., Do, B. T., Way, G. P., … Greene, C. S. (2018). Opportunities and obstacles for deep learning in biology and medicine. *Journal of the Royal Society Interface*, *15*(141). https://doi.org/10.1098/rsif.2017.0387.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Ciresan, D. C., Meier, U., Masci, J., Gambardella, L. M., & Schmidhuber, J. (2011). Flexible, high performance convolutional neural networks for image classification. In *Twenty-second international joint conference on artificial intelligence*.

Collobert, R., Bengio, S., & Mariéthoz, J. (2002). *Torch: A modular machine learning software library*.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273–297.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*(1), 21–27.

Cracknell, M. J., & Reading, A. M. (2013). The upside of uncertainty: Identification of lithology contact zones from airborne geophysics and satellite data using random forests and support vector machines. *Geophysics*, *78*(3), WB113–WB126.

Cressie, N. (1990). The origins of kriging. *Mathematical Geology*, *22*(3), 239–252.

Dammeier, F., Moore, J. R., Hammer, C., Haslinger, F., & Loew, S. (2016). Automatic detection of alpine rockslides in continuous seismic data using hidden Markov models. *Journal of Geophysical Research: Earth Surface*, *121*(2), 351–371. https://doi.org/10.1002/2015JF003647.

Dechter, R. (1986). *Learning while searching in constraint-satisfaction problems*. University of California, Computer Science Department.

Delhomme, J. P. (1978). Kriging in the hydrosciences. *Advances in Water Resources*, *1*, 251–266.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pretraining of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

DeVries, P. M. R., Viégas, F., Wattenberg, M., & Meade, B. J. (2018). Deep learning of aftershock patterns following large earthquakes. *Nature*, *560*(7720), 632.

Di, H., Shafiq, M., & AlRegib, G. (2017a). *Multi-attribute k-means cluster analysis for salt boundary detection*. In *79th EAGE conference and exhibition*. earthdoc.org. http://www.earthdoc.org/publication/publicationdetails/?publication=88632.

Di, H., Shafiq, M. A., & AlRegib, G. (2017b). Seismic-fault detection based on multiattribute support vector machine analysis. In *SEG technical program expanded abstracts 2017*. Society of Exploration Geophysicists, pp. 2039–2044.

Di, H., Wang, Z., & AlRegib, G. (2018). Deep convolutional neural networks for seismic salt-body delineation. In *AAPG annual convention & exhibition*. searchanddiscovery.com.

Dodge, D. A., & Harris, D. B. (2016). Large-scale test of dynamic correlation processors: Implications for correlation-based seismic pipelines. *Bulletin of the Seismological Society of America*. https://pubs.geoscienceworld.org/ssa/bssa/article-abstract/106/2/435/332173.

Dowla, F. U., Taylor, S. R., & Anderson, R. W. (1990). Seismic discrimination with arti-ficial neural networks: Preliminary results with regional spectral data. *Bulletin of the Seismological Society of America, 80*(5), 1346–1373. https://pubs.geoscienceworld.org/ssa/bssa/article-abstract/80/5/1346/119382.

Zuo, R., Xiong, Y., Wang, J., & Carranza, E. J. M. (2019). Deep learning and its application in geochemical mapping. *Earth-Science Reviews, 192*, 1–14.

Dramsch, J. (2019). Machine learning in 4D seismic data analysis: Deep neural networks in geophysics (PhD thesis).

Dramsch, J. S. (2020a). In *3D decision volume of SVM, random forest, and deep neural network*. figshare. https://doi.org/10.6084/m9.figshare.12640226.v1.

Dramsch, J. S. (2020b). *Code for 70 years of machine learning in geoscience in review*. figshare. https://doi.org/10.6084/m9.figshare.12666140.v1.

Dramsch, J. S., Amour, F., & Lüthje, M. (2018). Gaussian mixture models for robust unsupervised scanning-electron microscopy image segmentation of North Sea Chalk. In *First EAGE/PESGB workshop machine learning*. EAGE Publications BV. https://doi.org/10.3997/2214-4609.201803014.

Dramsch, J. S., Christensen, A. N., MacBeth, C., & Lüthje, M. (2019). Deep unsupervised 4D seismic 3D time-shift estimation with convolutional neural networks. *EarthArxiv*.

Dramsch, J. S., & Lüthje, M. (2018). Deep-learning seismic facies on state-of-the-art CNN architectures. In *Seg technical program expanded abstracts 2018*. Society of Exploration Geophysicists, pp. 2036–2040. https://doi.org/10.1190/segam2018-2996783.1.

Dreyfus, S. (1962). The numerical solution of variational problems. *Journal of Mathematical Analysis and Applications, 5*(1), 30–45.

Dubrule, O. (1984). Comparing splines and kriging. *Computers & Geosciences, 10*(2–3), 327–338.

Duvenaud, D. (2014). Automatic model construction with Gaussian processes (PhD thesis). University of Cambridge.

Dupont, E., Zhang, T., Tilke, P., Liang, L., & Bailey, W. (2018). Generating realistic geol-ogy conditioned on physical measurements with generative adversarial networks. *arXiv preprint arXiv:180203065*.

Engel, J., Agrawal, K. K., Chen, S., Gulrajani, I., Donahue, C., & Roberts, A. (2019). Gansynth: Adversarial neural audio synthesis. *arXiv preprint arXiv:190208710*.

Feng, X.-T., & Seto, M. (1998). Neural network dynamic modelling of rock microfracturing sequences under triaxial compressive stress conditions. *Tectonophysics, 292*(3), 293–309. https://doi.org/10.1016/S0040-1951(98)00072-9.

Ferreira, R., Brazil, E. V., & Silva, R. (2018). Texture-based similarity graph to aid seismic interpretation. In *ACE 2018 annual*. searchanddiscovery.com.

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mech-anism of pattern recognition unaffected by shift in position. *Biological Cybernetics, 36*(4), 193–202.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., … Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680.

Görtler, J., Kehlbeck, R., & Deussen, O. (2019). A visual https://distill.pub/2019/visual-exploration-gaussian-processes exploration of Gaussian processes. https://distill.pub/2019/visual-exploration-gaussian-processes *Distill*. https://doi.org/10.23915/distill.00017. https://distill.pub/2019/visual-exploration-gaussian-processes.

Gramstad, O., & Nickel, M. (2018). *Automated top salt interpretation using a deep convolutional net. In 80th EAGE conference and exhibition 2018*. earthdoc.org. http://www.earthdoc.org/publication/publicationdetails/?publication=92117.

Graves, A. (2012). Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711.*

Guillen, P., Larrazabal, G., González, G., Boumber, D., & Vilalta, R. (2015). Supervised learning to detect salt body. In *SEG technical program expanded abstracts 2015.* Society of Exploration Geophysicists, pp. 1826–1829. https://doi.org/10.1190/segam2015-5931401.1.

Guitton, A. (2018). 3D convolutional neural networks for fault interpretation. In *80th EAGE conference and exhibition 2018.* earthdoc.org. http://www.earthdoc.org/publication/publicationdetails/?publication=92118.

Guo, R., Zhang, Y. S., Lin, H., & Liu, W. (2017). Sweet spot interpretation from multiple attributes: Machine learning and neural networks technologies. In *First EAGE/AMGP/AMGE Latin.* earthdoc.org. http://www.earthdoc.org/publication/publicationdetails/?publication=90731.

Gupta, I., Rai, C., Sondergeld, C., & Devegowda, D. (2018). Rock typing in the upper Devonian-lower Mississippian woodford shale formation, Oklahoma, USA. *Interpretation, 6*(1), SC55–SC66. https://doi.org/10.1190/INT-2017-0015.1.

Hahnloser, R. H. R., Sarpeshkar, R., Mahowald, M. A., Douglas, R. J., & Seung, H. S. (2000). Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature, 405*(6789), 947–951.

Hajimoradlou, A., Roberti, G., & Poole, D. (2019). Predicting landslides using contour aligning convolutional neural networks. *arXiv: Computer Vision and Pattern Recognition.*

Hale, D. (2013). Methods to compute fault images, extract fault surfaces, and estimate fault throws from 3D seismic images. *Geophysics, 78*(2), O33–O43.

Hall, B. (2016). Facies classification using machine learning. *The Leading Edge, 35*(10), 906–909. https://doi.org/10.1190/tle35100906.1.

Hall, M., & Hall, B. (2017). Distributed collaborative prediction: Results of the machine learning contest. *The Leading Edge, 36*(3), 267–269. https://doi.org/10.1190/tle36030267.1.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* pp. 770–778.

Hermes, L., Frieauff, D., Puzicha, J., & Buhmann, J. M. (1999). Support vector machines for land usage classification in Landsat TM imagery. In *IEEE 1999 international geoscience and remote sensing symposium. IGARSS'99 (Cat. No. 99CH36293)* (Vol 1, pp. 348–350).

Ho, T. K. (1995). Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition* (Vol. 1, pp. 278–282).

Hochreiter, S., Bengio, Y., Frasconi, P., & Schmidhuber, J. (2001). *Gradient flow in recurrent nets: The difficulty of learning long-term dependencies.* A field guide to dynamical recurrent neural networks. IEEE Press.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences, 79*(8), 2554–2558.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks, 2*(5), 359–366. https://doi.org/10.1016/0893-6080(89)90020-8.

Huang, K. Y., Chang, W. R. I., & Yen, H. T. (1990). *Self-organizing neural network for picking seismic horizons. In Seg technical program expanded.* library.seg.org. https://library.seg.org/doi/pdf/10.1190/1.1890183.

Huijbregts, C., & Matheron, G. (1970). Universal kriging (an optimal method for estimating and contouring in trend surface analysis): 9th Intern. In *Sym on decision making in the mineral industries (proceedings to be published by Canadian Inst Mining), Montreal.*

Hulbert, C., Rouet-Leduc, B., Ren, C. X., Riviere, J., Bolton, D. C., Marone, C., & Johnson, P. A. (2018). Estimating the physical state of a laboratory slow slipping fault from seismic signals. http://arxiv.org/abs/1801.07806.

Ildstad, C. R., & Bormann, P. (n.d.). MalenoV_nD (MAchine LEarNing of Voxels). Retrieved from https://github.com/bolgebrygg/MalenoV.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Iqbal, H. (2018). In *HarisIqbal88/PlotNeuralNet v100*. https://doi.org/10.5281/zenodo.2526396.

Jeong, J., Park, E., Han, W. S., & Kim, K.-Y. (2014). A novel data assimilation methodology for predicting lithology based on sequence labeling algorithms. *Journal of Geophysical Research: Solid Earth*, *119*(10), 7503–7520. https://doi.org/10.1002/2014JB011279.

Jewett, M. A. S., Bombardier, C., Caron, D., Ryan, M. R., Gray, R. R., Louis, E. L. S., … Psihramis, K. E. (1992). Potential for inter-observer and intra-observer variability in x-ray review to establish stone-free rates after lithotripsy. *The Journal of Urology*, *147*(3), 559–562.

Jumper, J., Tunyasuvunakool, K., Kohli, P., Hassabis, D., & Team, A. (n.d.). Computational predictions of protein structures associated with COVID-19 (Tech. Rep.). Retrieved from https://deepmind.com/research/open-source/computational-predictions-of-protein-structures-associated-with-COVID-19.

Kadurin, A., Nikolenko, S., Khrabrov, K., Aliper, A., & Zhavoronkov, A. (2017). druGAN: An advanced generative adversarial autoencoder model for de novo generation of new molecules with desired molecular properties in silico. *Molecular Pharmaceutics*, *14*(9), 3098–3104. https://doi.org/10.1021/acs.molpharmaceut.7b00346.

Karra, S., O'Malley, D., Hyman, J. D., Viswanathan, H. S., & Srinivasan, G. (2018). Modeling flow and transport in fracture networks using graphs. *Physical Review E*, *97*(3-1), 033304. https://doi.org/10.1103/PhysRevE.97.033304.

Kelley, H. J. (1960). Gradient theory of optimal flight paths. *ARS Journal*, *30*(10), 947–954.

Kendall, A., Gal, Y., & Cipolla, R. (2018). Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7482–7491.

Khoshnevis, N., & Taborda, R. (2018). Prioritizing groundmotion validation metrics using semisupervised and supervised learning. *Bulletin of the Seismological Society of America*. https://pubs.geoscienceworld.org/ssa/bssa/article-abstract/108/4/2248/536309.

Kim, B., Kim, H., Kim, K., Kim, S., & Kim, J. (2019). Learning not to learn: Training deep neural networks with biased data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 9012–9020.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv:14126980*.

Koch, G., Zemel, R., & Salakhutdinov, R. (2015). Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, Vol. 2.

Kolmogorov, A. N. (1939). Sur l'interpolation et extrapolation des suites stationnaires. *Comptes Rendus de l'Académie des Sciences*, *208*, 2043–2045.

Kong, Q., Trugman, D. T., Ross, Z. E., Bianco, M. J., Meade, B. J., & Gerstoft, P. (2019). Machine learning in seismology: Turning data into insights. *Seismological Research Letters*, *90*(1), 3–14.

Kratzert, F., Klotz, D., Shalev, G., Klambauer, G., Hochreiter, S., & Nearing, G. (2019). Benchmarking a catchment-aware long short-term memory network (LSTM) for large-scale hydrological modeling. *arXiv preprint arXiv:1907.08456*.

Krige, D. G. (1951). A statistical approach to some mine valuation and allied problems on the Witwatersrand (PhD thesis). Johannesburg.

Krischer, L., & Fichtner, A. (2017). Generating seismograms with deep neural networks. *AGU Fall Meeting Abstracts*, *2017*, S41D-03.

Krishnan, R. G., Shalit, U., & Sontag, D. (2015). Deep Kalman filters. *arXiv preprint arXiv:1511.05121*.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105.

Krumbein, W. C., & Dacey, M. F. (1969). Markov chains and embedded Markov chains in geology. *Journal of the International Association for Mathematical Geology*, *1*(1), 79–96.

Kuehn, N. M., & Riggelsen, C. (2011). Modeling the joint probability of earthquake, site, and ground–motion parameters using Bayesian networks. *Bulletin of the Seismological Society of America*, *101*(1), 235–249. https://pubs.geoscienceworld.org/ssa/bssa/article-abstract/101/1/235/349494.

Kuzma, H. A. (2003). A support vector machine for avo interpretation. In *SEG technical program expanded abstracts 2003*. Society of Exploration Geophysicists, pp. 181–184.

Kvalsvik, J., et al. (2019). In *SegyIO*. https://github.com/equinor/segyio/.

Laloy, E., Hérault, R., Jacques, D., & Linde, N. (2017). *Efficient training-image based geostatistical simulation and inversion using a spatial generative adversarial neural network.* arXiv preprint arXiv:1708.04975.

Lary, D. J., Alavi, A. H., Gandomi, A. H., & Walker, A. L. (2016). Machine learning in geosciences and remote sensing. *Geoscience Frontiers*, 7(1), 3–10. https://doi.org/10.1016/j.gsf.2015.07.003.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.

Legendre, A. M. (1805). *Nouvelles méthodes pour la détermination des orbites des comètes*. F. Didot.

Li, J., & Castagna, J. (2004). Support vector machine (SVM) pattern recognition to AVO classification. *Geophysical Research Letters*, *31*(2), 948. https://doi.org/10.1029/2003GL018299.

Linnainmaa, S. (1970). The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors (Master's thesis, in Finnish). Univ. Helsinki.

Lin, M., Chen, Q., & Yan, S. (2013). Network in network. *arXiv preprint arXiv:1312.4400*.

Liu, Y., Chen, Z., Wang, L., Zhang, Y., Liu, Z., & Shuai, Y. (2015). Quantitative seismic interpretations to detect biogenic gas accumulations: A case study from Qaidam Basin, China. *Bulletin of Canadian Petroleum Geology*, *63*(1), 108–121. https://doi.org/10.2113/gscpgbull.63.1.108.

Lu, P., Morris, M., Brazell, S., Comiskey, C., & Xiao, Y. (2018). Using generative adversarial networks to improve deep-learning fault interpretation networks. *The Leading Edge*, *37*(8), 578–583.

Lundberg, S. M., Nair, B., Vavilala, M. S., Horibe, M., Eisses, M. J., Adams, T., et al. (2018). Explainable machine-learning predictions for the prevention of hypoxaemia during surgery. *Nature Biomedical Engineering*, *2*(10), 749–760.

Luo, X., Zhou, W., Wang, W., Zhu, Y., & Deng, J. (2017). Attention-based relation extraction with bidirectional gated recurrent unit and highway network in the analysis of geological data. *IEEE Access*, *6*, 5705–5715.

Ma, J., Jiang, Z., Tian, Q., & Couples, G. D. (2012). Classification of digital rocks by machine learning. In *ECMOR XIII-13th European*. earthdoc.org. http://www.earthdoc.org/publication/publicationdetails/?publication=62262.

Maggi, A., Ferrazzini, V., Hibert, C., Beauducel, F., Boissier, P., & Amemoutou, A. (2017). Implementation of a multistation approach for automated event classification at Piton de la Fournaise Volcano. *Seismological Research Letters*, *88*(3), 878–891. https://doi.org/10.1785/0220160189. https://pubs.geoscienceworld.org/ssa/srl/article–abstract/88/3/878/284054.

Malfante, M., Mura, M. D., Metaxian, J., Mars, J. I., Macedo, O., & Inza, A. (2018). Machine learning for volcano-seismic Signals: Challenges and perspectives. *IEEE Signal Processing Magazine*, *35*(2), 20–30. https://doi.org/10.1109/MSP.2017.2779166.

Mardan, A., & Javaherian, A. (2017). *Channel characterization using support vector machine. In 79th EAGE conference.* earthdoc.org. http://www.earthdoc.org/publication/publicationdetails/?publication=89283.

Marjanović, M., Kovačević, M., Bajat, B., & Voženílek, V. (2011). Landslide susceptibility assessment using SVM machine learning algorithm. *Engineering Geology*, *123*(3), 225–234. https://doi.org/10.1016/j.enggeo.2011.09.006.

Markov, A. A. (1906). Rasprostranenie zakona bol'shih chisel na velichiny, zavisyaschie drug ot druga. *Izvestiya Fiziko-matematicheskogo obschestva pri Kazanskom universitete*, *15*(135–156), 18.

Markov, A. A. (1971). Extension of the limit theorems of probability theory to a sum of variables connected in a Chain. *Dynamic Probabilistic Systems*, *1*, 552–577. Reprint in English of Markov, 1906.

Martinelli, G., Eidsvik, J., & Sinding-Larsen, R. (2013). Building Bayesian networks from basin-modelling scenarios for improved geological decision making. *Petroleum*, *19*, 289–304. http://pg.lyellcollection.org/content/early/2013/06/24/petgeo2012-057.abstract.

Masotti, M., Campanini, R., Mazzacurati, L., Falsaperla, S., Langer, H., & Spampinato, S. (2008). TREMOrEC: A software utility for automatic classification of volcanic tremor. *Geochemistry, Geophysics, Geosysystem*, *9*(4), 4007. https://doi.org/10.1029/2007GC001860.

Masotti, M., Falsaperla, S., Langer, H., Spampinato, S., & Campanini, R. (2006). Application of support vector machine to the classification of volcanic tremor at Etna, Italy. *Geophysical Research Letters*, *33*(20), 113. https://doi.org/10.1029/2006GL027441.

Matalas, N. C. (1967). Mathematical assessment of synthetic hydrology. *Water Resources Research*, *3*(4), 937–945.

Matheron, G. (1963). Principles of geostatistics. *Economic Geology*, *58*(8), 1246–1266.

Matheron, G. (1981). *Splines and kriging; their formal equivalence*. Syracuse, New York: Syracuse University Geology Contribution D. F. Merriam ed.

McCormack, M. (1991). Neural computing in geophysics. *The Leading Edge*, *10*(1), 11–15. https://doi.org/10.1190/1.1436771.

Mignan, A., & Broccardo, M. (2019a). A deeper look into 'deep learning of aftershock patterns following large earthquakes': Illustrating first principles in neural network physical interpretability. In *International work-conference on artificial neural networks*. pp. 3–14.

Mignan, A., & Broccardo, M. (2019b). One neuron versus deep learning in aftershock prediction. *Nature*, *574*(7776), E1–E3.

Mitchell, T. M. (1997). *Machine learning*. New York: McGraw-Hill.

Mjolsness, E., & DeCoste, D. (2001). Machine learning for science: State of the art and future prospects. *Science*, *293*(5537), 2051–2055. https://doi.org/10.1126/science.293.5537.2051.

Mosser, L., Dubrule, O., & Blunt, M. J. (2017). Reconstruction of three-dimensional porous media using generative adversarial neural networks. *Physical Review E*, *96*(4-1), 043309. https://doi.org/10.1103/PhysRevE.96.043309.

Mosser, L., Dubrule, O., & Blunt, M. J. (2018a). Conditioning of three-dimensional generative adversarial networks for pore and reservoirscale models. http://arxiv.org/abs/1802.05622.

Mosser, L., Dubrule, O., & Blunt, M. J. (2018b). Stochastic seismic waveform inversion using generative adversarial networks as a geological prior. http://arxiv.org/abs/1806.03720.

Mosser, L., Kimman, W., Dramsch, J. S., Purves, S., De la Fuente Briceño, A., & Ganssle, G. (2018). Rapid seismic domain transfer: Seismic velocity inversion and modeling using deep generative neural networks. In *80th EAGE conference and exhibition 2018, June*. EAGE Publications BV. https://doi.org/10.3997/2214-4609.201800734.

Mosser, L., Oliveira, R., & Steventon, M. (2019). Probabilistic seismic interpretation using Bayesian neural networks. In *81st EAGE conference and exhibition 2019. Vol. 2019*. (pp. 1–5).

Nash, J. (1951). Non-cooperative games. *Annals of Mathematics*, *54*(2), 286–295.

Neal, R. M. (1996). *Bayesian learning for neural networks*. Springer Science & Business Media.

Newendorp, P. D. (1976). *Decision analysis for petroleum exploration*. Tulsa, OK: Penn Well Books.

Ochoa, L. H., Niño, L. F., & Vargas, C. A. (2018). Fast magnitude determination using a single seismological station record implementing machine learning techniques. *Geodesy and Geodynamics*, *9*(1), 34–41. https://doi.org/10.1016/j.geog.2017.03.010.

Ohrnberger, M. (2001). researchgate.net. Accessed: 2018-12-17.

Paganini, M., de Oliveira, L., & Nachman, B. (2018). CaloGAN: Simulating 3D high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks. *Physical Review D*, *97*(1), 014021.

Panakkat, A., & Adeli, H. (2007). Neural network models for earthquake magnitude prediction using multiple seismicity indicators. *International Journal of Neural Systems*, *17*(01), 13–33.

Pasolli, E., Melgani, F., & Donelli, M. (2009). Automatic analysis of GPR images: A pattern-recognition approach. *IEEE Transactions on Geoscience and Remote Sensing*, *47*(7), 2206–2217.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., … Lerer, A. (2017). Automatic differentiation in PyTorch. In *Nips autodiff workshop*.

Pearl , J. (2012). The do-calculus revisited. *arXiv preprint arXiv:1210.4852*.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Poulton, M., Sternberg, B., & Glass, C. (1992). Location of subsurface targets in geophysical data using neural networks. *Geophysics*, *57*(12), 1534–1544. https://doi.org/10.1190/1.1443221.

Preston, F. W., & Henderson, J. (1964). *Fourier series characterization of cyclic sediments for stratigraphic correlation*. Kansas Geological Survey.

Purves, S., Alaei, B., & Larsen, E. (2018). Bootstrapping Machine-Learning based seismic fault interpretation. In *ACE 2018 annual convention & exhibition*. searchanddiscovery.com. http://www.searchanddiscovery.com/abstracts/html/2018/ace2018/abstracts/2856016.html

Ramcharan, A., Baranowski, K., McCloskey, P., Ahmed, B., Legg, J., & Hughes, D. P. (2017). Deep learning for image-based cassava disease detection. *Frontiers in Plant Science*, *8*, 1852.

Ramcharan, A., McCloskey, P., Baranowski, K., Mbilinyi, N., Mrisho, L., Ndalahwa, M., … Hughes, D. P. (2019). A mobile-based deep learning model for cassava disease diagnosis. *Frontiers in Plant Science*, *10*, 272.

Rasmussen, C. E. (2003). Gaussian processes in machine learning. In *Summer school on machine learning*, pp. 63–71.

Recht, B., Roelofs, R., Schmidt, L., & Shankar, V. (2019). Do imagenet classifiers generalize to imagenet? *arXiv preprint arXiv:1902.10811*.

Reddy, R. K. T., & Bonham-Carter, G. F. (1991). A decision-tree approach to mineral potential mapping in snow lake area, Manitoba. *Canadian Journal of Remote Sensing*, *17*(2), 191–200. https://doi.org/10.1080/07038992.1991.10855292.

Richardson, A. (2018). Seismic Full-Waveform inversion using deep learning tools and techniques. http://arxiv.org/abs/1801.07232.

Roden, R., & Chen, C. W. (2017). Interpretation of DHI characteristics with machine learning. *First Break*. http://www.earthdoc.org/publication/publicationdetails/?publication=88069.

Rolnick, D., Donti, P. L., Kaack, L. H., Kochanski, K., Lacoste, A., Sankaran , K., et al. (2019). Tackling climate change with machine learning. *arXiv preprint arXiv:1906.05433*.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International conference on medical image computing and computer-assisted intervention*, pp. 234–241.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386.

Ross, Z. E., Meier, M.-A., Hauksson, E., & Heaton, T. H. (2018a). Generalized seismic phase detection with deep learning. http://arxiv.org/abs/1805.01075.

Ross, Z. E., Meier, M. A., & Hauksson, E. (2018). P-wave arrival picking and first-motion polarity determination with deep learning. *Journal of Geophysical Research: Solid Earth*, *123*, 5120–5129. https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2017JB015251.

Ross, Z. E., Meier, M.-A., Hauksson, E., & Heaton, T. H. (2018b). Generalized seismic phase detection with deep learning. *Bulletin of the Seismological Society of America*, *108*(5A), 2894–2901.

Röth, G., & Tarantola, A. (1994). Neural networks and inversion of seismic data. *Journal of Geophysical Research: Solid Earth*, *99*(B4), 6753. https://doi.org/10.1029/93JB01563.

Rouet-Leduc, B., Hulbert, C., & Bolton, D. C. (2018). Estimating fault friction from seismic signals in the laboratory. *Geophysical Research Letters*, *45*, 1321–1329. https://onlinelibrary.wiley.com/doi/abs/10.1002/2017GL076708.

Rouet-Leduc, B., Hulbert, C., & Lubbers, N. (2017). Machine learning predicts laboratory earthquakes. *Geophysical Research Letters*, *44*, 9276–92826. https://onlinelibrary.wiley.com/doi/abs/10.1002/2017GL074677.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1988). Learning representations by back-propagating errors. *Cognitive Modeling*, *5*(3), 1.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., … Fei-Fei, L. (2015). ImageNet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, *115*(3), 211–252. https://doi.org/10.1007/s11263-015-0816-y.

Russell, S. J., & Norvig, P. (2010). *Artificial intelligence—A modern approach third international edition*. Pearson Education. http://vig.pearsoned.com/store/product/1,1207,store-12521_isbn-0136042597,00.html.

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, *3*(3), 210–229.

Saporetti, C. M., da Fonseca, L. G., Pereira, E., & de Oliveira, L. C. (2018). Machine learning approaches for petrographic classification of carbonate-siliciclastic rocks using well logs and textural information. *Journal of Applied Geophysics*, *155*, 217–225. https://doi.org/10.1016/j.jappgeo.2018.06.012.

Schütt, K. T., Arbabzadah, F., Chmiela, S., Müller, K. R., & Tkatchenko, A. (2017). Quantum-chemical insights from deep tensor neural networks. *Nature Communication*, *8*, 13890. https://doi.org/10.1038/ncomms13890.

Schwarzacher, W. (1972). The semi-Markov process as a general sedimentation model. In *Mathematical models of sedimentary processes*. Springer, pp. 247–268.

Sen, S., Kainkaryam, S., Ong, C., & Sharma, A. (2020). SaltNet: A production-scale deep learning pipeline for automated salt model building. *The Leading Edge*, *39*(3), 195–203.

Shah, R., & Innig, L. (2019). In *Aftershock issues*. https://github.com/rajshah4/aftershocks_issues.

Shashidhara, B. M., Scott, M., & Marburg, A. (2020). Instance segmentation of benthic scale worms at a hydrothermal site. In *The ieee winter conference on applications of computer vision*, pp. 1314–1323.

Shen, C. (2018). A transdisciplinary review of deep learning research and its relevance for water resources scientists. *Water Resources Research*, *54*(11), 8558–8593.

Shen, D., Wu, G., & Suk, H.-I. (2017). Deep learning in medical image analysis. *Annual Review of Biomedical Engineering*, *19*, 221–248. https://doi.org/10.1146/annurev-bioeng-071516-044442.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sun, A. Y., Scanlon, B. R., Zhang, Z., Walling, D., Bhanja, S. N., Mukherjee, A., & Zhong, Z. (2019). Combining physically based modeling and deep learning for fusing GRACE satellite data: Can we learn from mismatch? *Water Resources Research*, *55*(2), 1179–1195.

Sutskever, I., Martens, J., Dahl, G., & Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In S. Dasgupta & D. McAllester (Eds.), *Proceedings of the 30th international conference on machine learning, 17–19 June*. (Vol. 28, pp. 1139–1147). Atlanta, Georgia, USA: PMLR. http://proceedings.mlr.press/v28/sutskever13.html.

Talarico, E., Leäo, W., & Grana, D. (2019). Comparison of recursive neural network and Markov chain models in facies inversion. In *Petroleum geostatistics 2019*. (Vol. 2019, pp. 1–5).

Tancik, M., Srinivasan, P. P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., … Ng, R. (2020). Fourier features let networks learn high frequency functions in low dimensional domains. *arXiv preprint arXiv:2006.10739*.

Theano Development Team. (2016). Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, *1605.02688*. http://arxiv.org/abs/1605.02688.

Titos, M., Bueno, A., García, L., Benítez, M. C., & Ibañez, J. (2018). Detection and classification of continuous volcano-seismic signals with recurrent neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, *57*(4), 1936–1948.

Trabelsi, C., Bilaniuk, O., Zhang, Y., Serdyuk, D., Subramanian, S., Santos, J. F., … Pal, C. J. (2017). Deep complex networks. *arXiv preprint arXiv:1705.09792*.

Turing, A. M. (1950). I.—Computing machinery and intelligence. *Mind*, *LIX*(236), 433–460. https://doi.org/10.1093/mind/LIX.236.433.

Uieda, L. (2018). Verde: Processing and gridding spatial data using Green's functions. *Journal of Open Source Software*, *3*(29), 957. https://doi.org/10.21105/joss.00957.

Valentine, A. P., & Kalnins, L. M. (2016). An introduction to learning algorithms and potential applications in geomorphometry and earth surface dynamics. *Earth Surface Dynamics*, *4*, 445–460.

Valera, M., Guo, Z., Kelly, P., Matz, S., Cantu, V. A., Percus, A. G., … Viswanathan, H. S. (2017). Machine learning for graphbased representations of three-dimensional discrete fracture networks. http://arxiv.org/abs/1705.09866.

van der Baan, M., & Jutten, C. (2000). Neural networks in geophysical applications. *Geophysics*, *65*(4), 1032–1047. https://doi.org/10.1190/1.1444797.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008.

Waldeland, A., Jensen, A., Gelius, L., & Solberg, A. (2018). Convolutional neural networks for automated seismic interpretation. *The Leading Edge*, *37*(7), 529–537. https://doi.org/10.1190/tle37070529.1.

Waldeland, A. U., & Solberg, A. (2017). Salt classification using deep learning. *79th EAGE conference and exhibition*. http://www.earthdoc.org/publication/publicationdetails/?publication=88635.

Wang, H., Wellmann, J. F., Li, Z., Wang, X., & Liang, R. Y. (2017). A segmentation approach for stochastic geological modeling using Hidden Markov random fields. *Mathematical Geosciences*, *49*(2), 145–177. https://doi.org/10.1007/s11004-016-9663-9.

Wang, K., Lomask, J., & Segovia, F. (2017). Automatic, geologic layer-constrained well-seismic tie through blocked dynamic warping. *Interpretation*, *5*(3), SJ81–SJ90. https://doi.org/10.1190/INT-2016-0160.1.

Wang, L. X., & Mendel, J. M. (1992). Adaptive minimum prediction-error deconvolution and source wavelet estimation using Hopfield neural networks. *Geophysics*, *57*(5), 670–679. https://library.seg.org/doi/abs/10.1190/1.1443281.

Wang, Z., Di, H., Shafiq, M. A., Alaudah, Y., & AlRegib, G. (2018). Successful leveraging of image processing and machine learning in seismic structural interpretation: A review. *The Leading Edge*, *37*(6), 451–461.

Watkins, C. J. C. H. (1989). *Learning from delayed rewards*. Cambridge: King's College.

Wei, S., Yonglin, O., Qingcai, Z., & Jiaqiang, H. (2018). *Unsupervised machine learning: K-means clustering velocity semblance Auto-Picking. In 80th EAGE conference*. earthdoc. org. http://www.earthdoc.org/publication/publicationdetails/?publication=92299

Wickman, F. E. (1968). Repose period patterns of volcanoes. V. General discussion and a tentative stochastic model. *Arkiv for Mineralogi Och Geologi*, *4*(5), 351.

Williams, C. K. I. (1998). Prediction with Gaussian processes: From linear regression to linear prediction and beyond. In M. I. Jordan (Ed.), *Learning in graphical models* (pp. 599–621). Springer.

Williams, C. K. I., & Rasmussen, C. E. (2006). *Gaussian processes for machine learning (Vol. 2) (No. 3)*. Cambridge, MA: MIT Press.

Wirgin, A. (2004). The inverse crime. *arXiv preprint math-ph/0401050*.

Witten, I. H., Frank, E., & Hall, M. A. (2005). *Practical machine learning tools and techniques*. Morgan Kaufmann. p. 578.

Wu, H., & Zhang, B. (2018). A deep convolutional encoderdecoder neural network in assisting seismic horizon tracking. http://arxiv.org/abs/1804.06814.

Xie, P., Zhou, A., & Chai, B. (2019). The application of long short-term memory (lstm) method on displacement prediction of multifactor-induced landslides. *IEEE Access*, *7*, 54305–54311.

Xie, Q., Hovy, E., Luong, M.-T., & Le, Q. V. (2019). Self-training with Noisy Student improves ImageNet classification. *arXiv preprint arXiv:191104252*.

Xie, X., Qin, H., Yu, C., & Liu, L. (2013). An automatic recognition algorithm for GPR images of RC structure voids. *Journal of Applied Geophysics*, *99*, 125–134. https://doi.org/10.1016/j.jappgeo.2013.02.016.

Xu, B., Wang, N., Chen, T., & Li, M. (2015). Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*.

Zhang, Y., & Paulson, K. V. (1997). Magnetotelluric inversion using regularized Hopfield neural networks. *Geophysical Prospecting*, *45*(5), 725–743. http://www.earthdoc.org/publication/publicationdetails/?publication=33881.

Zhao, T., Li, F., & Marfurt, K. (2017). Constraining self-organizing map facies analysis with stratigraphy: An approach to increase the credibility in automatic seismic facies classification. *Interpretation*, *5*(2), T163–T171. https://doi.org/10.1190/INT-2016-0132.1.

Zhao, X., & Mendel, J. M. (1988). *Minimum-variance deconvolution using artificial neural networks. In library.seg.org*. https://library.seg.org/doi/pdf/10.1190/1.1892433.

Zhao, Z., & Gross, L. (2017). Using supervised machine learning to distinguish microseismic from noise events. In *SEG technical program expanded abstracts 2017*. Society of Exploration Geophysicists, pp. 2918–2923. https://doi.org/10.1190/segam2017-17727697.1.

Zheng, H., Fu, J., Mei, T., & Luo, J. (2017). Learning multi-attention convolutional neural network for fine-grained image recognition. In *Proceedings of the IEEE international conference on computer vision*, pp. 5209–5217.

Zhu, W., & Beroza, G. C. (2018). PhaseNet: A deep-neural-network–based seismic arrival time picking method. http://arxiv.org/abs/1803.03211.