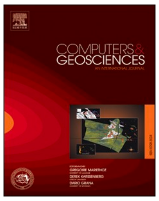
Contents lists available at ScienceDirect

Computers and Geosciences

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



Semantic segmentation of explosive volcanic plumes through deep learning

T.C. Wilkes [*], T.D. Pering, A.J.S. McGonigle

Department of Geography, University of Sheffield, Winter Street, Sheffield, S10 2TN, United Kingdom

A R T I C L E I N F O

Keywords:
Deep learning
Semantic segmentation
Convolutional neural network
Volcanic explosion
Volcanic ash

A B S T R A C T

Tracking explosive volcanic phenomena can provide important information for hazard monitoring and volcano research. Perhaps the simplest forms of monitoring instruments are visible-wavelength cameras, which are routinely deployed on volcanoes around the globe. Here, we present the development of deep learning models, based on convolutional neural networks (CNNs), to perform semantic segmentation of explosive volcanic plumes on visible imagery, therefore classifying each pixel of an image as either explosive plume or not explosive plume. We have developed 3 models, each with average validation accuracies of >97% under 10-fold cross-validation; although we do highlight that, due to the limited training and validation dataset, this value is likely an over-estimate of real-world performance. We then present model deployment for automated retrieval of plume height, rise speed and propagation direction, all parameters which can have great utility particularly in ash dispersion modelling and associated aviation hazard identification. The 3 trained models are freely available for download at https://doi.org/10.15131/shef.data.17061509.

1. Introduction

A large number of active volcanoes exhibit some form of explosive behaviour, with repose periods between explosions that can vary drastically, from minutes to years or much greater. Explosive phenomena include low-explosivity Strombolian eruptions, larger ash-rich Vulcanian eruptions, and more explosive/rare Sub-Plinian and Plinian events. Ash-rich explosive columns can give rise to local hazards such as ash fall, volcanic bomb ejection, and pyroclastic density currents, whilst also having the potential to cause significant disruption to aviation (e.g., Alexander, 2013; Brown et al., 2017). Furthermore, explosion frequency and magnitude can provide important insights into the activity state of a volcano (e.g., Fee et al., 2017; Taddeucci et al., 2012) and unlock information about explosion dynamics (e.g., Campion et al., 2018; Cassidy et al., 2015). Identifying and tracking explosive phenomena is therefore of critical importance for scientific and monitoring purposes.

A wide range of instrumentation is available for studying volcanic phenomena, for example, seismometers, gas sensors, infrared cameras, satellite data, infrasound microphones; however, some of the simplest and cheapest tools are visible imaging systems. Indeed, a large number of active volcanoes have some form of permanent camera, such as a web camera, monitoring them, which can be used to identify changing surficial behaviour and to identify explosive volcanic plumes (e.g., Hort et al., 2018; Witsil and Johnson, 2020). Visible imagery has also commonly been combined with other data streams (e.g. infrasound, seismics, gas measurements) in multi-parametric studies, since it is able to provide a lot of important tangible context and information on surficial behaviour (Gerst et al., 2013; Matoza et al., 2022; Yamada et al., 2019). Since manual inspection and analysis of image datasets can be time consuming, some focus has been placed on automating image analysis procedures for both visible and other forms of imagery (Dye and Morra, 2020; Valade et al., 2014; Witsil and Johnson, 2020; Witt et al., 2018). As with some of those works, machine learning can support such efforts.

Due to significant advances in computing power, computer literacy and software availability, the use of machine learning, and in particular the deep learning subset, has proliferated in recent years (Bergen et al., 2019). Furthermore, these state-of-the-art computational methods are being used to solve problems in volcanology (e.g., Dempsey et al., 2020; Dye and Morra, 2020; Fenner et al., 2021; Manley et al., 2021; Ren et al., 2020; Witsil and Johnson, 2020); however, Witsil and Johnson (2020) highlight that, prior to their work, the application of machine learning to visible imagery of volcanoes was lacking in the literature. Their work used an artificial neural network (ANN) to classify images from Villarrica volcano into 5 pertinent classes (inactivity, nocturnal glow, clouds, dark emissions, light emissions), whilst also presenting a blob detection algorithm to extract and analyse plume pixels by identifying the dominant colour contrasting region of an image. However, the blob

* Corresponding author.
E-mail address: tcwilkes1@sheffield.ac.uk (T.C. Wilkes).

https://doi.org/10.1016/j.cageo.2022.105216
Received 8 December 2021; Received in revised form 7 June 2022; Accepted 10 August 2022
Available online 6 September 2022

detection is at times quite coarse and is not specifically designed for solely segmenting volcanic plumes; for example, it is possible that other features of an image, such as clouds, could be segmented if they form the most significant contrasting region of the image. Furthermore, meteorological conditions make image classification complex when attempting to define an image as either cloud-covered or containing volcanic gas or ash. In reality, it is possible for both volcanic and meteorological components to be present and, thus, a higher-resolution classification could be advantageous for some applications. To date, the use of semantic segmentation (pixel-level classification) with convolutional neural networks (CNNs), which would provide a further layer of detail to classification, is absent from volcanological research; however, a number of works have already achieved pixel-level plume classification in volcano infrared imagery with great success and utility (Bombrun et al., 2018; Tournigand et al., 2019; Valade et al., 2014; Wood et al., 2019).

Much like the proliferation of machine learning, in recent years open-source software has become widespread in the geosciences (e.g., David et al., 2016). Such developments provide researchers with highly specialised tools to augment and, importantly, standardise their research, whilst also avoiding the requirement of often expensive software licenses. Furthermore, open-source software encourages reproducibility in research, promoting rigorous scientific procedure. In particular, Python programming has opened the door to a number of topics in volcanology (e.g., Bear-Crozier et al., 2012; Daggitt et al., 2014; Diaz Moreno et al., 2019; Gliβ et al., 2017; Peters and Oppenheimer, 2018), making the application of complex algorithms much more accessible to relatively inexperienced users. The work presented here makes use of two Python libraries recently developed for machine/deep learning applications: Albumentations (Buslaev et al., 2020) and Segmentation Models (Yakubovskiy, 2019). Such tools could encourage more geoscientists to explore the application of machine/deep learning in their own research in the coming years.

Herein, we present a semantic segmentation model, using a CNN, which identifies pixels in visible imagery that belong to an explosive volcanic plume. First, we step through the stages of model development, discussing model structure, model training and evaluation of model performance. We subsequently provide some examples of the model's application in volcanology, highlighting that it has the potential to automate the provision of valuable data (such as plume height, rise speed and direction of propagation) for use in volcano monitoring/ research.

1.1. Deep learning overview

Deep learning was designed to mimic the behaviour of the human brain, consisting of a number of layers of interconnected nodes to form an Artificial Neural Network (ANN). Each node has an associated weight and bias to transform its input into some output. In essence, through the use of training data, the goal is to train the network weights and biases such that, given a specific input, a network provides a desired output. Convolutional Neural Networks (CNNs) are common in image processing problems, where a number of the layers contain filters (or kernels) that are convolved with the data to extract features that can be useful in classification problems. Much more comprehensive overviews of deep learning and CNNs are available elsewhere (e.g., Goodfellow et al., 2016; LeCun et al., 2015).

In semantic segmentation a CNN typically consists of: (1) a contracting path (often referred to as the encoder or backbone), which decomposes the initial image into features, creating "feature maps" (predominantly using convolutional filters and max pooling layers); and (2) an expansive path (decoder), which gradually up-samples the data back to the original image size such that each pixel is classified individually in the final output (Fig. 1). For each down-sampling layer of the encoder, information can also be passed to the corresponding layer of the decoder using skip connections, therefore allowing the propagation of higher resolution features through the network; without these pathways, the down-sampling of the encoder would preclude the detection of finer features. This architecture differs from the more prominent image classification CNNs, which omit the decoder, therefore contracting the image into a single classification value.

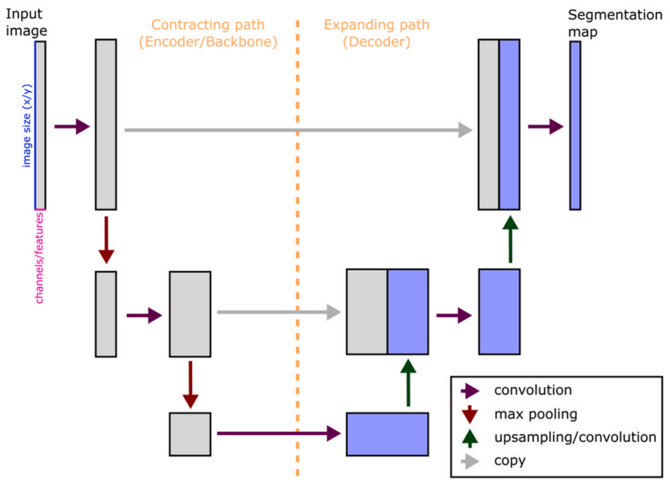


Fig. 1. A simplified diagram of the UNet architecture. Other semantic segmentation model architectures follow a number of similar principles. Note that the number of layers is dependent on the model backbone, therefore this figure is a generic representation and not based on any specific setup. Grey and blue boxes represent image/feature arrays, with the colours simply identifying the previous source of that block. The size of blocks in the horizontal direction represents the number of channels/features at that stage, which increases with depth in the contracting path. The size of blocks in the vertical direction represents the size of the image (x/y size), which decreases with depth in the contracting path, as coarser features are isolated. Arrows represent different operations applied to the arrays. A more detailed discussion of this architecture can be found in Ronneberger et al. (2015). (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

Looking at Fig. 1, most operations work by sliding a filter (or kernel matrix) across the image/feature map and applying some mathematical operation. The filter is typically much smaller than the image size, e.g. often of the order 2×2 or 3×3. In a convolution, each patch of the feature map is multiplied by a kernel and the sum of the result is taken; this results in extraction of features, such as vertical lines, horizontal lines, and much more complex features. Max pooling is simply extracting the maximum value within each patch, therefore resulting in down-sampling which highlights the most prominent feature in a patch. Up-sampling in its simplest form involves increasing the x/y resolution of a feature map by repeating rows and columns. Here, however, this is combined with a transposed convolution that results in a halving of the number of features in the feature map. Again, we highlight that much more thorough explanations of CNNs and UNet are available elsewhere (Goodfellow et al., 2016; LeCun et al., 2015; Ronneberger et al., 2015).

2. Methodology

2.1. Training data

Supervised deep learning models require correctly labelled data (training and validation data), allowing the model to fit its weights and biases to these data, therefore learning to predict the correct pixel labels from an input image. We manually labelled 120 images to identify all pixels of those images that were attributed to an explosive volcanic plume. Typically a much larger dataset (many 1000s at least) of training data would be used to train a CNN model (e.g., Zhu et al., 2020); however, due to the substantial time requirements for labelling images, we were restricted to this size of dataset. Specifically, the number of images labelled here was chosen following a number of preliminary tests, which found that fewer images led to quite poor model generalization during model validation. A larger training dataset is likely to reduce the error of the model predictions, enabling better generalization to images that the model has not previously seen. As we discuss later, future work to develop a much more comprehensive dataset could be of great use to the geoscience and volcanology communities.

Images were sourced from a combination of images/videos acquired on field campaigns and through internet searches. Due to licensing on some images, we are unfortunately unable to make the full training dataset available for further use. We do, however, suggest that the development of a large-scale freely available labelled dataset for use in volcanology could be of great benefit to the community; here, the principal aim of the study was to provide a proof-of-concept for semantic segmentation in this field, and to create useable models for real-world deployments by the community. Images were selected to cover a range of explosion types, volcanoes and eruption conditions; therefore, the selection was not entirely random. Table 1 summarises the volcanoes and eruption styles present in the image set. Whilst the set primarily consists of Strombolian (n = 63) and Vulcanian (n = 43) explosions, it does also contain a small number of other eruption styles (e.g. Sub-Plinian, Phreatic). 8 images with no explosion present were also included. These images all contained meteorological cloud, with the aim of exposing the model to features that it could confuse with a volcanic explosion; from these data the model should learn not to label such features as explosive plume. We should note, a model that consistently over-identifies explosive plumes is likely to be less useful than one which under-identifies smaller explosions; i.e., we believe it is preferable to have a model with a false-negative bias rather than a false-positive bias. Including more of such images could improve model performance under complex meteorological conditions; however, it would also lead to larger class imbalance (see later in this section), therefore we decided to limit the number of explosion-free images in our dataset. Furthermore, along with the images containing no explosion, all images with explosions contain a considerable portion of background from which the model will learn. Of the 112 images containing explosive plumes, 27 had a (mostly) clear blue sky background, 28 were overcast (relatively homogenous grey cloud), 44 had patchy clouds and 13 had a sea background (Stromboli). All images are daytime, or dusk/dawn scenes, and all were in the standard red-green-blue (RGB) format.

Defining explosive plume pixels, even manually, was a non-trivial task, due to the subjectivity of this classification. Indeed, different expert labelers would almost certainly label images in different ways. Nevertheless, it is likely that the bulk regions of plume would be agreed upon and that the subjectivity labelling only becomes significant close to plume boundaries; we therefore believe that in general the model will not be significantly impacted by the plume boundaries chosen herein. In general, pixels were only labelled as an explosive plume if they created a relatively substantial opaque region in the image; however, when fine ash was associated with the tail of an explosive plume or a fine ash Strombolian explosion, it was also included. Fig. 2 shows some plume scenarios and how they were manually labelled. Some ash-poor/water-rich explosive plumes were also included in the training dataset, however, focus was placed on ash-rich explosions, and the performance of the model on ash-poor explosions is not assessed herein.

Of the 120 labelled images, 6 (5%) were set aside as validation images. Validation images allow evaluation of the model's performance on images it has not been trained on, therefore giving an idea of the generalisation ability of the model. For the initial stage of training, rather than randomly selecting these images, they were specifically chosen to represent a wide range of conditions and plume appearances; this included one image that was explosion-free, to ensure that validation included the assessment of the model on this common scenario.

Table 1
Summary of volcanoes included in the study and the associated eruption styles included.

Volcano	Strombolian	Vulcanian	Vulcanian/Sub-Plinian	Sub-Plinian	Phreatic	Undefined	PDC	No Eruption	Total
Anak Krakatau	4	5	–	–	–	–	–	–	**9**
Arenal	2	–	–	–	–	–	–	3	**5**
Augustine	–	1	–	–	–	–	–	–	**1**
Calbuco	–	–	–	1	–	–	–	–	**1**
Chimboraza	–	2	–	–	–	–	–	–	**2**
Colima	–	6	–	–	–	–	–	–	**6**
El Reventador	–	3	–	–	–	–	–	–	**3**
Etna	5	1	–	–	–	–	–	–	**6**
Eyjafjallajökul	–	1	–	–	–	–	–	–	**1**
Fuego	5	1	–	–	–	–	–	–	**6**
Gunung Ibu	–	1	–	–	–	–	–	–	**1**
Ili Lewotolok	–	1	–	–	–	–	–	–	**1**
Karymsky	–	1	–	–	–	–	–	–	**1**
Kilauea	–	–	–	–	1	1[a]	–	–	**2**
Mahameru	–	1	–	–	–	–	–	–	**1**
Mayon	–	3	–	–	–	–	–	2	**5**
Merapi	–	–	–	–	–	–	1	–	**1**
Mount Agung	–	1	2	–	–	–	–	–	**3**
Mount Sinabung	–	1	–	–	–	–	–	–	**1**
Mount St. Helen's	1	–	–	–	–	–	–	–	**1**
Ol Doinyo Lengai	–	–	–	–	–	–	–	2	**2**
Poas	–	2	–	–	–	–	–	–	**2**
Popocatépetl	–	4	–	–	–	–	–	–	**4**
Sabancaya	–	1	–	–	–	–	–	–	**1**
Sakurajima	–	5	–	–	–	–	–	–	**5**
Santiaguito	1	–	–	–	–	–	–	–	**1**
Stromboli	34	–	–	–	–	–	–	–	**34**
Tavurvur	–	1	–	–	–	–	–	–	**1**
Telica	–	1	–	–	–	–	–	–	**1**
Gilgit, Phandar Lake	–	–	–	–	–	–	–	1	**1**
Yasur	11	–	–	–	–	–	–	–	**11**
Total	**63**	**43**	**2**	**1**	**1**	**1**	**1**	**8**	**120**

PDC - pyroclastic density current.
[a] Eruption caused by a rock fall into the lava lake.

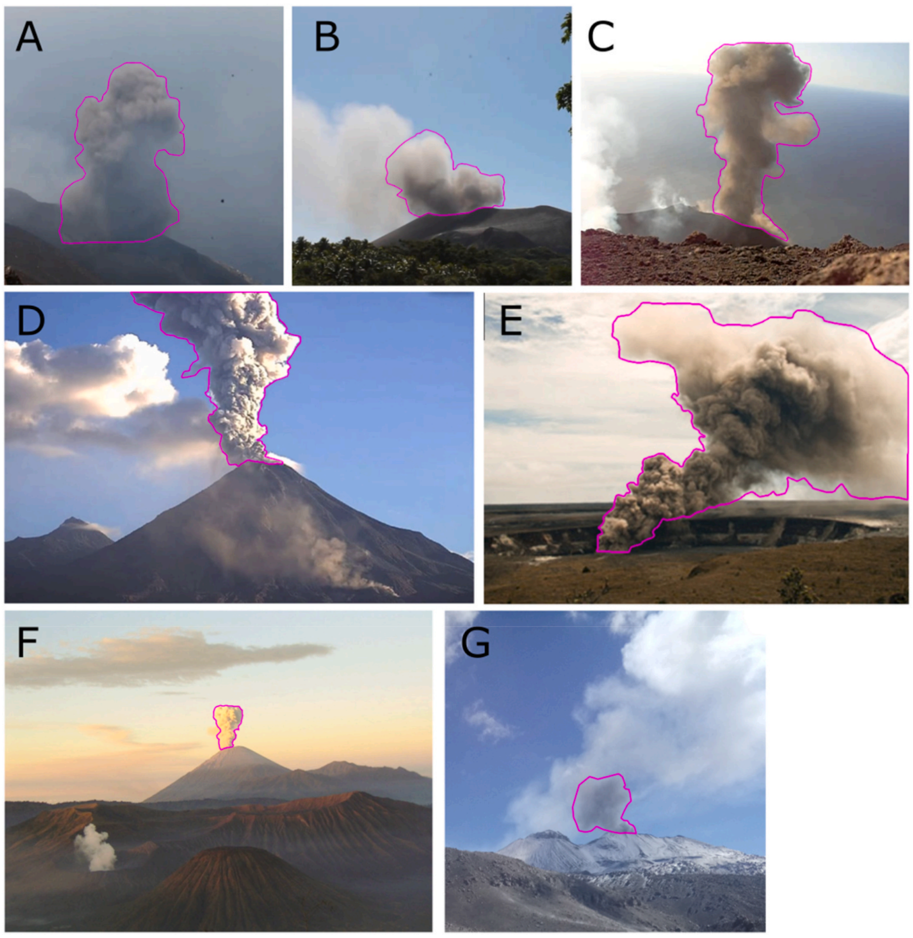


Fig. 2. Examples of explosive plume labelling. For clarity, only plume boundaries are outlined here, rather than overlaying the entire pixel classification. A) Strombolian explosion, Stromboli (T. Pering) - contains some subjectivity of plume location as it merges with the volcano flank. B) Strombolian explosion, Yasur (T. Pering) - as the explosion expands into passively degassed plume the explosive plume boundary becomes quite subjective and may be defined differently by different experts. C) Strombolian explosion, Stromboli (T. Pering). D) Vulcanian explosion, Colima (https://wiki.seg.org/wiki/File:Active_volcano.jpeg, *accessed April 26, 2022*). E) Rockfall triggered explosion, Kīlauea (Orr et al., 2013). F) Vulcanian explosion, Semeru (https://en.wikipedia.org/wiki/File:Mahameru-volcano.jpeg, *accessed April 26, 2022)*. G) Small Vulcanian explosion, Sabancaya (T. Wilkes).

More comprehensive *k*-fold cross-validation was performed in the latter stages of model evaluation, where a range of validation image sets were used (see Section 2.3). Data augmentation, critical for improving model performance when the volume of training data is limited (e.g., Shorten and Khoshgoftaar, 2019), was performed on the remaining 114 images, resulting in a training dataset of 684 images. Using the image augmentation Python library *Albumentations* (Buslaev et al., 2020), each training image and its associated label mask had 5 transformations applied to it, producing 5 new training images: horizontal flipping, zoom out (scale = −0.5), zoom in (scale = 0.5), rotate (20°), Gaussian noise ($\sigma^2 = 500$).

Following augmentation, class distributions were: class 0 = 6.1×10^7 pixels (not explosive plume), class 1 = 9.5×10^6 pixels (explosive plume), giving a ratio of ≈6:1. Krawczyk (2016) states that most research on imbalanced datasets focusses on ratios between 4:1 and 100:1, therefore our dataset is at the lower end of what might be considered imbalanced. Furthermore, some imbalance may again promote the false-negative bias, since there are more occurrences of the negative (0) class in our dataset. We therefore believe that the class imbalance is unlikely to cause significant problems in this work.

2.2. *Model training*

Model training is discussed in detail in the Appendix. In brief, models were constructed using the open-source Python library *Segmentation Models* (Yakubovskiy, 2019), based on the deep learning *Keras* and *Tensorflow* libraries, allowing simple implementation and examination of a range of model architectures and encoders. The four model architectures available are: UNet (Ronneberger et al., 2015), FPN (Lin et al., 2016), PSPNet (Zhao et al., 2017), LinkNet (Chaurasia and Culurciello, 2018). Each architecture can then be built with any of the 25 available backbones. Further to these configurations, a range of hyper-parameters (defining how the network is trained) were explored, to find the overall optimal model setup. Each model was initiated with weights pre-trained on the ImageNet database (Deng et al., 2009; Russakovsky et al., 2015), an approach termed transfer learning. Due to its efficiency, this method, which takes advantage of an encoder that has already been well trained on a generic dataset to extract important image features, is particularly popular when training deep learning models with a limited training dataset or restricted computing resources (e.g., Razavian et al., 2014; Shin et al., 2016; Tajbakhsh et al., 2016). Training time of a model is dependent on a number of factors, but typically took ≈2–4 h on a NVIDIA® GeForce GTX 850M (4 GB) GPU.

2.3. *Model evaluation*

The output of the model is an array of the same size as the input array, but with a single channel in the case of binary classifications such as this. The pixels contain a value between 0 and 1, indicating the model's confidence that pixel belongs to class 1. In this case, since there are only 2 classes, typically any value of >0.5 will be assigned as explosive volcanic plume, since the model predicts that it is more likely to be explosive plume (class 1) than anything else (class 0). This threshold value can be manually adjusted by the user; for instance, using a higher threshold will mean that fewer pixels are classified as explosive plume, with the model having a higher confidence that these pixels do indeed belong to that class.

Fig. 3 displays the confusion matrix, a technique that is commonly used in machine learning for summarising model performance. It defines the terms: true positive (*TP*), true negative (*TN*), false positive (*FP*), false

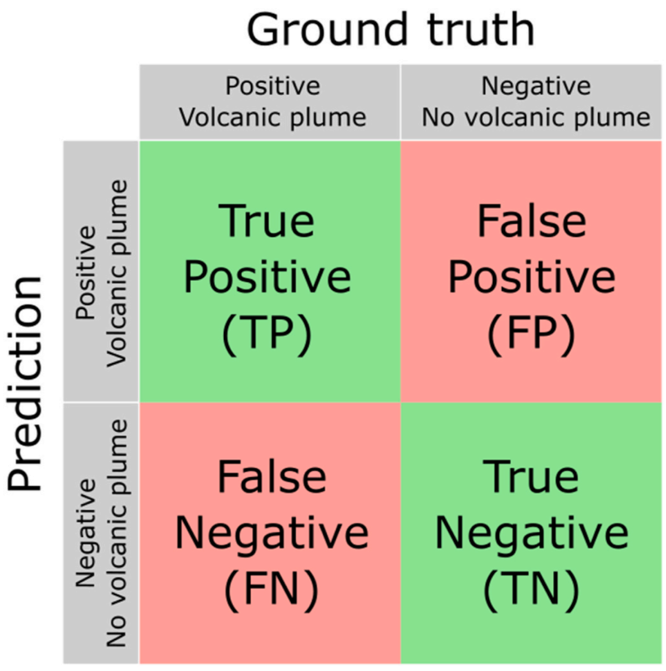


Fig. 3. Confusion matrix used to generate model evaluation metrics. True positives and true negatives indicate when the model's prediction for a pixel's classification is in line with the manually labelled ground truth data. Conversely, false positives and false negatives indicate where the model has incorrectly classified pixels.

negative (*FN*). A number of performance metrics can then be generated from the parameters of the confusion matrix. For instance, accuracy is the total number of correctly identified pixels ($TP + TN$) divided by the total number of pixels in an image ($TP + TN + FP + FN$). Herein, model performance was further quantitatively assessed using the Receiver Operating Characteristic curve (ROC) and its Area Under the Curve (AUC_{ROC}) (Bradley, 1997). We also use precision-recall curves, which provide an assessment for imbalanced datasets (Saito and Rehmsmeier, 2015), with the associated AUC score denoted as AUC_{prec} hereafter. The curves are based on parameters defined below, which can easily be calculated at a range of classification threshold values. The ROC curve is a plot of true positive rate (*TPR* hereafter; also referred to as sensitivity or recall elsewhere in the literature) versus false positive rate (*FPR*); the precision-recall curve is a plot of precision (*Pr*) versus *TPR*. Finally, Intersection-Over-Union (*IoU*), is commonly used for evaluating semantic segmentation performance (Rahman and Wang, 2016). All metrics are calculated from confusion matrix values as follows:

$$TPR = TP \,/\, (TP + FN) \tag{1}$$

$$FPR = FP \,/\, (FP + TN) \tag{2}$$

$$Pr = TP \,/\, (TP + FP) \tag{3}$$

$$IoU = TP \,/\, (TP + FP + FN) \tag{4}$$

Model performances, as calculated through these metrics, are presented in Table A1. ROC and precision-recall plots of the 4 model architectures are presented in Fig. A1. For a number of model runs, performances were incredibly similar, and whilst one model may have had a better AUC_{prec} score, the other may have presented a better AUC_{ROC} score. Absolute definition of the best model is therefore difficult, thus, along with the performance metrics we chose to also incorporate qualitative assessment of validation images when quantitative performances of models were very similar. For instance, a model that performed relatively well in the metric scores but which generated FPs in the explosion-free validation image was not selected for further investigation; we prefer a model that can accurately predict when no explosion is present in an image.

Following rapid model testing/validation, which was necessary to quickly determine models which could have good utility for this application, we subsequently performed a more detailed model validation on the models selected for final model deployment. This involved *k*-fold cross-validation (e.g., Goodfellow et al., 2016) whereby *k* sets of training/validation images are generated, with the model being trained and evaluated on each set. An average of performance metrics can then be calculated, providing a more thorough analysis of model performance. Here we used $k = 10$ with a 90/10% training/validation split (12 validation images per set) – this ensured that every labelled image was used exactly once as a validation image in the procedure. The sets were generated randomly, to avoid any bias from manual selection.

3. Results and discussion

For deployment, 3 trained models are freely available online (https://doi.org/10.15131/shef.data.17061509), due to the similarity in their performances on our validation data. Indeed, it is often the case that very different machine learning models can perform comparably and have similar applicability to a specific task (Zhao et al., 2015). It is possible that one model will work better in certain scenarios, therefore we encourage users to test each model for their specific deployment. From tests herein, however, we cannot make any significant distinction between the model performances or define specific scenarios in which one model might outperform the others. Confusion matrices for the 3 models are shown in Tables A4-A6, with their performance metrics presented in Table A2 and their cross-validation performances in Table A3.

The 3 models made available are: the UNet architecture (Ronneberger et al., 2015) with a DenseNet121 encoder (Huang et al., 2017); UNet with an Inceptionv3 encoder (Szegedy et al., 2015); FPN (Lin et al., 2016) with an EfficientNetB0 encoder (Tan and Le, 2019). As the name suggests, EfficientNetB0 is a much smaller network, with fewer trainable parameters than the other two models; therefore it is an especially good option for applications with limited computing resources (although this benefit is more significant during model training than it is during deployment). Nevertheless, in our 10-fold cross-validation evaluations it still performs comparably to the UNet-DenseNet121 and UNet-Inceptionv3 models, often outperforming the other two models. All models present average AUC_{ROC} and AUC_{prec} scores exceeding 0.99 (with a score of 1.0 representing a perfect model performance), with the exception of UNet-Inceptionv3 AUC_{prec} which still displays a high score of 0.9849 ± 0.0317 (± 1 standard deviation). Intersection-over-Union (IoU) results show more notable disparity, with EfficientNetB0 performing best here (0.9593 ± 0.0158), followed by DenseNet121 (0.9336 ± 0.0529) and finally Inceptionv3 (0.9155 ± 0.1109); however, the latter two model means are dragged down by one or two model run(s) out of the 10 in each cross-validation set, which is reflected in the high associated standard deviations.

Validation accuracies were very similar, with scores of 0.9798 ± 0.0055 (or $97.98 \pm 0.55\%$), 0.9722 ± 0.0171, and 0.9709 ± 0.0132 for the EfficientNetB0, Inceptionv3, and DenseNet121 models, respectively. We note, however, that these accuracies are almost certainly above that which can be expected from the models under extensive deployment, since the small number of validation images (due to the overall small number of labelled images available in this work) will not cover all image scenarios that may be encountered by users at volcanoes across the globe. Even with larger training datasets, since deep learning models are highly over-parameterised, overfitting to training data is a common problem in deep learning (e.g., Salman and Liu, 2019).

Below, for brevity, we present results and test deployments of only one of the 3 available models: UNet-DenseNet121. The other two models provided very similar pixel classifications for the test images examined in the sections below, with the exception of difficult meteorological conditions (see section 3.1.3) where we highlight the difference in model predictions.

The predicted mask of a representative validation image is displayed in Fig. 4; as with all subsequent discussion, this relates to the UNet-DenseNet121 model. In this case the model replicates the manual labelling very well, with an accuracy of 99.7%. Labelling differences are solely confined to small areas around the edges of the plume - subjective boundaries which could also be mislabelled by manual labelling in some cases.

3.1. Model applications

In this section we present results from model deployment on test data which the model has not been exposed to during the training procedure. Note, unlike the validation images, these test images do not have counterpart manually labelled segmentations, since quantification of model performance in this section is not strictly the aim. Here, we are presenting applications much like in a real-world deployment, where supplementary manual labels will not be available. We therefore do not further provide quantitative evaluations of model performance on these deployments.

3.1.1. Automated plume height and rise speed extraction

Information on ash plume injection height and trajectory are critical input parameters to ash dispersion models (Beckett et al., 2020; Scollo et al., 2008), which are used to inform decisions regarding aviation dangers from volcanic ash. Furthermore, explosion rise dynamics can provide interesting insights into the buoyant nature of the plume and, thus, potentially the source bulk density (Yamamoto et al., 2008). One application of our model is the automated retrieval of ash column height and its associated rise speed. Although not presented here, one further application may be to estimate eruptive mass and/or mass eruption rate (Scollo et al., 2019), albeit with the requirement of an assumed plume density and symmetry, the former of which may introduce significant uncertainties. This is a further parameter of use to ash dispersion models (Costa et al., 2016; Scollo et al., 2008), and therefore may warrant exploration in further works.

Here we use the model to analyse a video of an explosion on Sabancaya volcano, Peru, acquired on April 27, 2018, to calculate plume height and rise speed. The camera, with a vertical field of view (FOV) of 13.3°, was positioned 10 km from the Sabancaya summit. From this information, the height of pixels above the crater can be calculated using trigonometry, assuming the plume rises in a vertical plane above the crater (this assumption will of course introduce some error into the retrieval). A more rigorous calibration of the camera orientation can be made using the projected skyline of a digital elevation model (Scollo et al., 2019); however, this was not performed for the proof of concept test herein. The segmentation model is then applied to each image and the maximum height of the plume is found.

Fig. 5A shows the segmentation results. We note that in the final images, parts of the explosive plume are not labelled by the model, possibly as they appear white and have a strong resemblance to cloud at this point. However, generally in this example it can be seen that the model performs extremely well in identifying the plume and therefore provides reliable associated information on plume height and rise speed.

Fig. 5B shows the maximum plume height recorded in each frame of the Sabancaya imagery and the associated plume rise speed between successive images. The plume rises to ≈1250 m above Sabancaya's summit before extending beyond the FOV of this camera. Optimised setups for this application would ensure that the geometry was such that plumes don't rise beyond the vertical extent of the camera FOV; in such cases, a final atmospheric injection height, at which point further vertical movement of ash is negligible, may be estimated. The maximum rise speed in this sequence was measured as 6.9 m/s; this value is in agreement with buoyant plume velocities (6.7–23.8 m/s) found by Tournigand et al. (2017) for Strombolian/Vulcanian eruptions. The use of optical flow algorithms would provide more in-depth analysis of plume turbulence and speeds, but the simple method presented here provides easy access to the progression of the plume front, which can be useful for applications such as assessing the buoyant rise phase and transition to it from an initial gas thrust phase (e.g., Johnson et al., 2004).

3.1.2. Plume direction extraction

Similarly to extracting plume height, the plume drift (or explosive) direction can provide useful information for both ash dispersion models, which require accurate wind fields at the time of atmospheric injection (e.g., Beckett et al., 2020), and also potentially explosion dynamics in the subsurface (Fitzgerald et al., 2020; Gaudin et al., 2014). Again, this can be extracted from the model output by first generating an array where each pixel is labelled with an associated angle from the source pixel. We note that this 2D representation of a 3D dispersion is

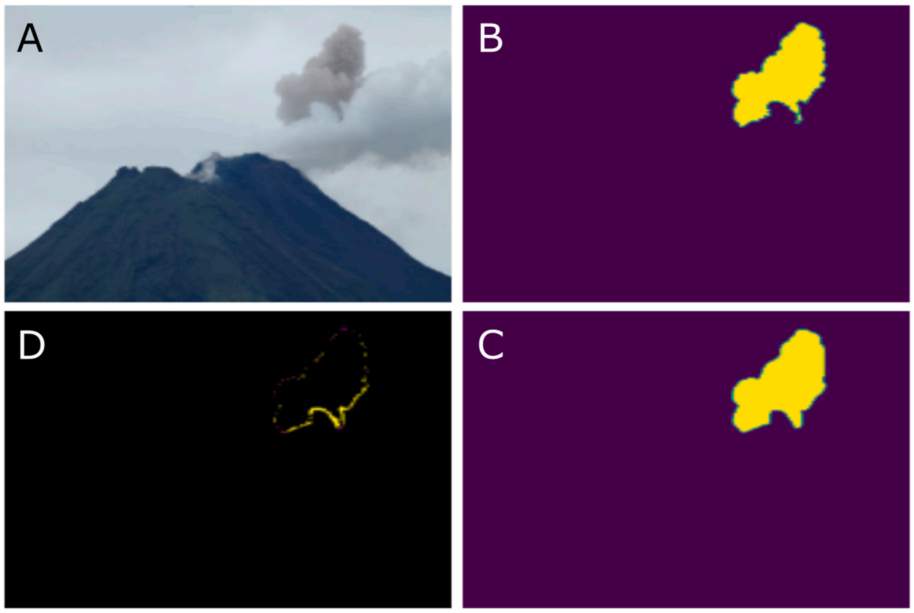


Fig. 4. Validation image of an explosive volcanic eruption at Arenal volcano. A) Raw image. B) Manually labelled plume mask. C) Predicted plume mask (yellow = class 1, purple = class 0). D) Difference image – false positives are displayed in yellow, false negatives are displayed in purple, correct classifications are black. There are very few false negatives in this image. False positives are confined to around the edges of the plume, therefore are not too problematic and could also be a result of poor manual labelling as well as model prediction errors. This image is relatively representative of all validation images, errors are primarily confined to the edges of the plumes. Model accuracy here is 0.9970. Image source: https://commons.wikimedia.org/wiki/File:Arenal_strombolian_eruption_2008.JPG. (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

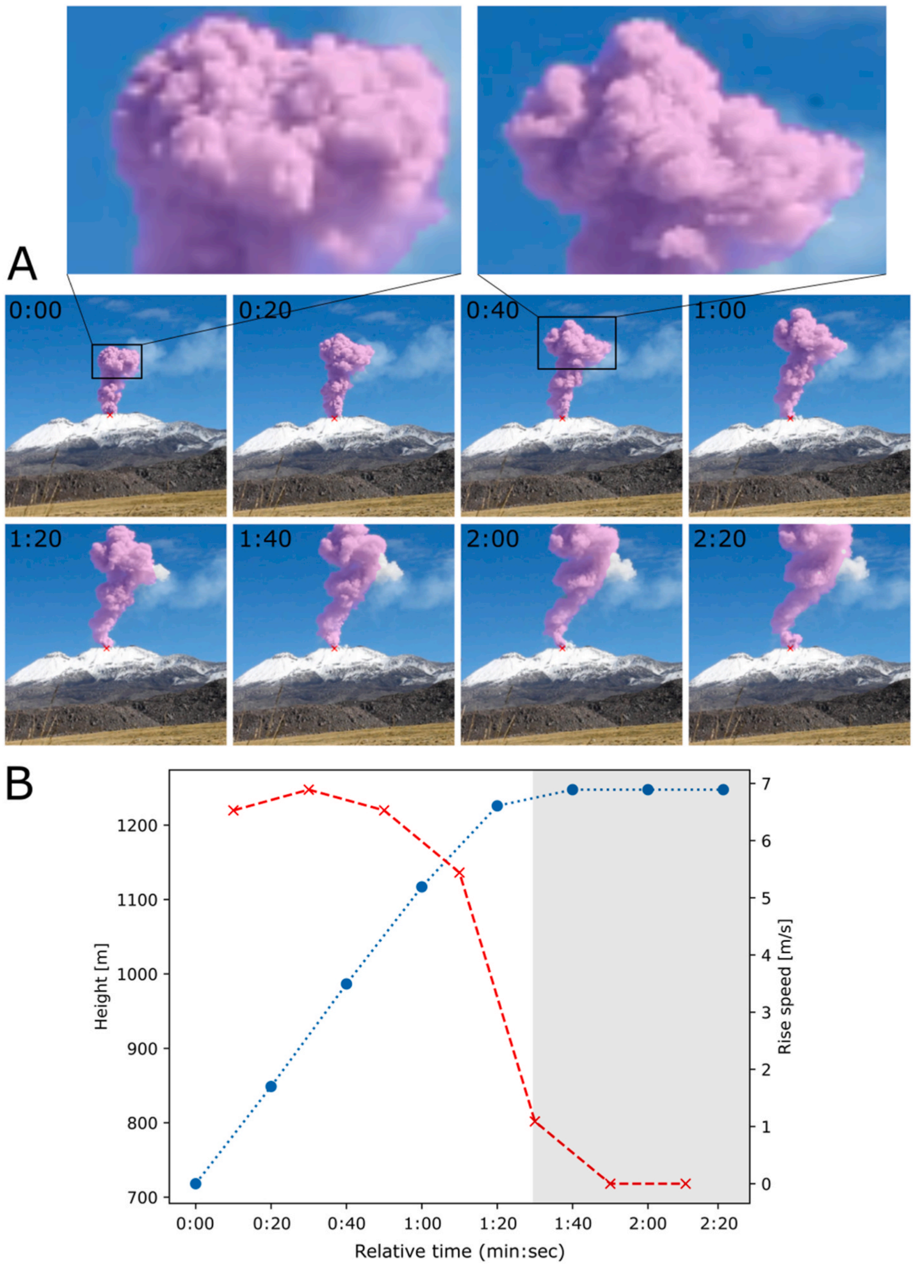


Fig. 5. A) Image sequence (frames extracted from video) of an explosion at Sabancaya volcano April 27, 2018. Time since initial frame shown below (minutes: seconds). Pixels classified by the model as being explosive volcanic plume are overlain with a semi-transparent pink colour; 2 zoomed regions highlight the fine-scale accuracy of the segmentation. The red cross marks the source coordinate used for plume height extraction and rise speed calculations. In the final frames part of the plume is not identified, to the right-hand side, however at this point this portion is beginning to detach from the upwardly rising plume and appears to contain very little visible ash. B) Resulting plume height (blue dots) and plume speed (red crosses) calculation. The rise speed is initially relatively stable (although the sequence misses the initial explosive burst) then starts to slow before extending beyond the FOV of the camera. The grey shaded region identifies values in which the plume has extended beyond the FOV of the camera. (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

sub-optimal, however it can still provide a reasonable estimate of plume motion as long as bulk motion is close to perpendicular to the camera viewing direction. Furthermore, it may be possible to combine recent advances in 3D plume reconstruction (Wood et al., 2019) with this segmentation model to extract accurate 3D insights into plume dispersion. For instance, Albadra et al. (2020) present 3D reconstruction of a highly condensed plume at Pacaya volcano that required accurate segmentation of the volcanic plume pixels. They present a number of segmentation techniques with varying degrees of accuracy, but could perhaps have benefitted from a well-trained CNN model to perform the segmentation. We note, however, that the model presented herein would not be directly applicable, due to the type of plume (passive degassing) and geometry of images (taken from a drone) used in that study.

This plume direction application was applied to an image sequence of a small Strombolian explosion at Yasur volcano, Vanuatu, acquired on July 8, 2018. The image sequence also contains a speckled cloud background, therefore providing a good test of the model's performance in slightly challenging conditions. Fig. 6 shows the results of plume propagation retrievals, identifying that the Strombolian explosion does not rise vertically but is instead drifting with the wind. From this viewing position the plume is not immediately visible at the point of explosion, since it occurs in a deep crater; it has therefore already transitioned to primarily drifting with ambient wind conditions in our images. Plume speed can again be extracted in this application, making it possible to distinguish between speeds at different orientations from the source.

This example also highlights the utility of thresholding the model predictions. Using the standard segmentation threshold of 0.5 for binary classification (i.e. a pixel is classified as containing explosive plume if the prediction probability is greater than 0.5) parts of the volcano flank are incorrectly identified as volcanic plume. However, through empirical tests we find that increasing the threshold of positive prediction to 0.96 removes the error in this case (Fig. 7). Interestingly, this area of flank is not wrongly classified as a false positive in all images of this Yasur sequence, only for the third image. It may be that for some applications a stricter threshold to minimise false positives is

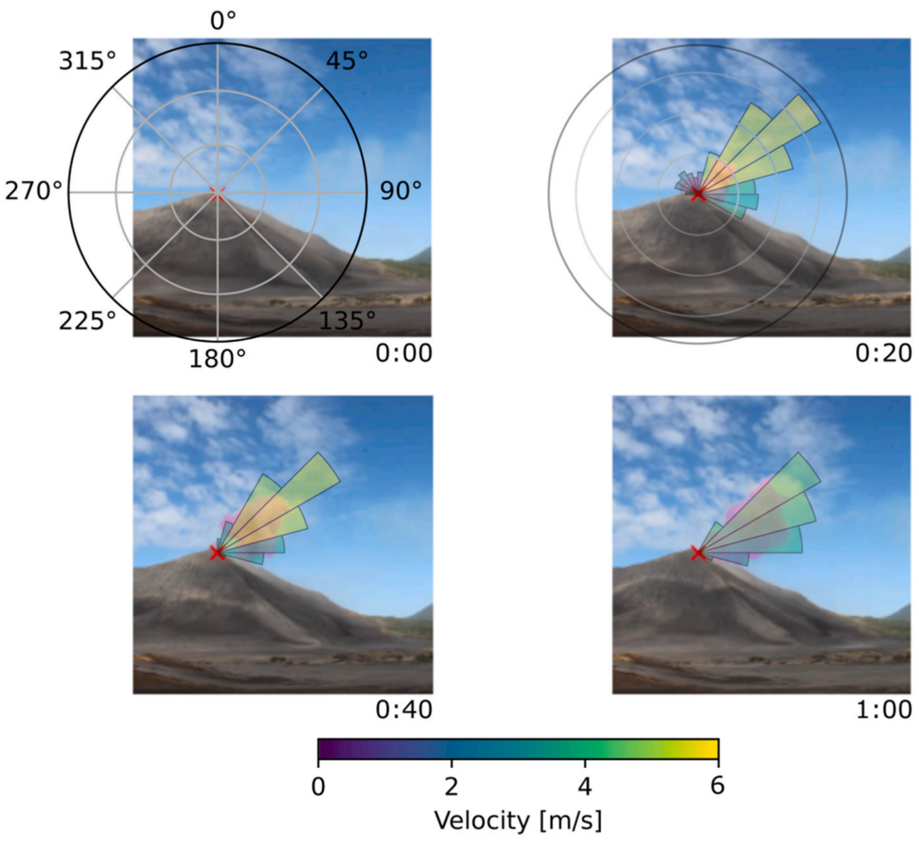


Fig. 6. Image sequence of a Strombolian explosion at Yasur volcano, Vanuatu, acquired on July 8, 2018. Relative times of each image are below (minutes: seconds). The explosive plume pixels are identified in pink. The orientation of plume pixels from the source coordinate is overlain as a histogram rose plot (size of bars represent number of pixels) with bar colours representing the maximum plume speed in that bin. Varying degrees of axis labels are included for improved clarity. Note, the scales on each plot are different, i.e. there are a much greater number of explosive plume pixels in the bottom right panel than in the top right. A classification threshold of 0.96 was used for all images here, to remove some false positives (see Fig. 7). (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

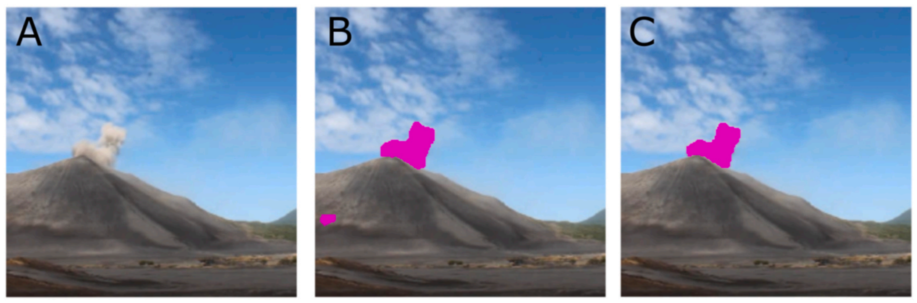


Fig. 7. An image of Yasur volcano with different levels of thresholding used for model classification. This image corresponds to the top right panel in Fig. 6, where a threshold of 0.96 was used throughout the retrievals. A) The raw image. B) The standard model prediction is used, corresponding to a probability threshold of 0.5 in binary classification such as this. C) A prediction threshold of 0.96 is used to therefore only classify explosive plume pixels if the model has a relatively high confidence in the classification. This value was manually tuned through testing, but appears to work well in a number of cases. In this case it removes the erroneous false positive that can be seen on the volcano flank in B, but does not adversely affect the identification of plume pixels.

advantageous, whilst in others, especially under good meteorological conditions, users may prefer to use a threshold closer to 0.5, allowing the capture of perhaps more subtle explosions for instance. In general, from our testing we would suggest that thresholding at >0.9 is likely to be most suitable in the majority of cases. As can be seen in Fig. 7, increasing the threshold to 0.96 does not significantly decrease the identification of the explosion in this case, but does remove the area of false positive classification; therefore, a higher threshold may significantly increase the model precision (eq. (3)). We note here that the pixel classifications made by the UNet-Inceptionv3 and FPN-EfficientNetb0 models are almost identical to the UNet-DenseNet121 classifications in Fig. 7; however, when using the standard predictive threshold of 0.5, neither falsely classifies patches of explosive plume on the volcano flank.

3.1.3. *Difficult segmentation conditions*

It must be acknowledged that there will undoubtedly be scenarios where semantic segmentation models fail, which relates to the generalisation properties of the model (e.g., Li et al., 2021; Salman and Liu, 2019). In this work we anticipate that certain scenarios in particular may exacerbate this issue, for example in images with difficult meteorological conditions. We might expect that when summit clouds begin to mix with the plume, and partially obscure it, the accuracy of the segmentation algorithm will suffer. Here, we test the performance of the model in one such scenario, from Yasur volcano, to evaluate its performance. This image sequence was extracted from a video acquired on July 9, 2018 during a field campaign.

Fig. 8 shows segmentation results for an image sequence using a threshold of 0.96 – a threshold that was shown to work well on the previous image sequence (Fig. 6). Here, we display results from all 3 models available for deployment, since each shows significantly different pixel classifications. In every case the model predictions for this sequence are quite poor. Large areas of non-plume are falsely labelled as plume and the majority of plume pixels are not identified as such. We note, however, that manual labelling of this scene would also

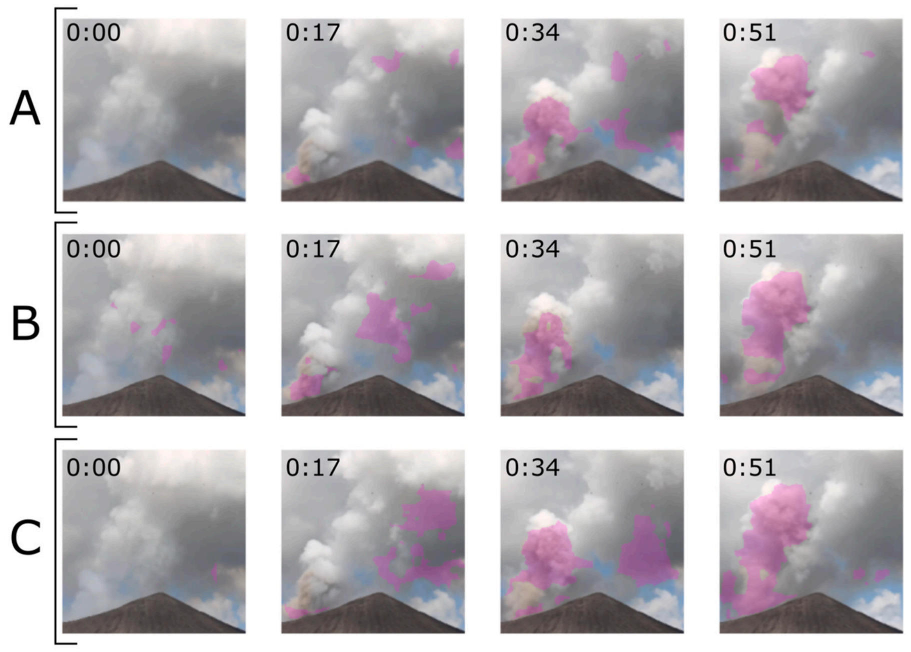


Fig. 8. An image sequence from Yasur volcano (July 9, 2018), Vanuatu, representing sup-optimal conditions for model segmentation of explosive volcanic plumes. Set A) UNet-DenseNet121; Set B) UNet-Inceptionv3; Set C) FPN-EfficientNetB0. Images were segmented using a model probability threshold of 0.96. Relative times of each image are in the top left corner (minutes:seconds). Imagery shows cloudy conditions where the explosive plume mixes and is partially obscured by low-lying cloud. All 3 models correctly identify parts of the explosion, especially in the final image of the sequence. However, each model also predicts relatively large areas of false positives; these areas are different for each model. Note that manual labelling of this scene would also be difficult, with a high subjectivity of what pixels should be classified as explosive volcanic plume.

be difficult – it is not immediately clear which pixels should be labelled as explosive plume, with a high level of subjectivity meaning that different experts would likely produce considerably different labels for these images. Nevertheless, it is clear that some regions show significant false positives that would lead to errors in any automated analysis.

This presents another example of where thresholding the images, with a higher prediction probability required for plume classification, could be helpful. However, this is not a complete solution to poor model performance, since here the model's probability output predicts that some areas of cloud are more likely to be plume than some areas of actual plume; thus, no threshold level will solve the problem. In reality only better model training, in particular a larger training dataset, would truly solve this issue by improving model generalisation to scenes that it hasn't seen before. Large-scale databases exist for both image classification, e.g. ImageNet (Deng et al., 2009), and semantic segmentation, e.g. COCO (Lin et al., 2014), applications; however, whilst the COCO database contains 91 classes, it does not include any volcanic phenomena. We therefore suggest that a community effort to develop such a database in volcanology, and indeed the broader geosciences, could be a worthwhile endeavour, to promote the development of further machine learning models that can aid research in this field.

With the current model, it may be preferable to attempt to make the model completely ignore scenes like this, restricting the false positives in favour of increased false negatives. We find that using a threshold of 0.9995 markedly reduces the number of false positives in Fig. 8 image sequence, whilst maintaining reasonable predictions for the more optimal images at Sabancaya and Yasur presented in Figs. 5 and 7, albeit with the introduction of more false negatives, especially in the later images of the Sabancaya plume. By combining this model with the work of Witsil and Johnson (2020), who present an ANN model for classifying volcano images, it may be possible to first exclude poor visibility images and run only pertinent images through our model; thus, providing more confidence in the subsequent segmentation results. Indeed, we emphasise that these two models are complementary tools for automated image analysis, as opposed to being competing or mutually exclusive. Alternatively, it should be possible to train a segmentation model to identify and segment cloud as an additional class. Primarily, this would require the development of a more comprehensive training dataset that contains cloud labels, which requires investment of further time for manual labelling. This could additionally be extended to further classes, such as other volcanic phenomena (e.g. lava flows, lava domes). We further acknowledge that explosions styles that were not present or poorly represented in the training dataset (e.g. Plinian eruptions, pyroclastic density currents) may not be successfully segmented by these models and should be the subject of further investigations. Again, a more substantial training dataset could incorporate more of such phenomena.

4. Concluding remarks

We have presented what we believe to be the first use of CNNs for semantic segmentation of explosive volcanic plumes. The model was trained on a manually labelled set of 120 images (augmented to produce a full training dataset of 684 images after setting aside 6 images for validation) and developed using open source Python packages. We have made 3 trained models available (https://doi.org/10.15131/shef.data.17061509) to promote their use for volcano research and monitoring purposes. Each of these models achieved average validation accuracies of >97% during 10-fold cross-validation; however, we do suggest that this is almost certainly an over-estimation of model performance in general, due to the relatively small training/validation dataset that will not capture all scenarios to which this model could be applied.

We present model applications for automated plume rise speed, height and direction determination, which could have great utility in volcano monitoring/research. In particular, we highlight that these data are important input parameters for volcanic ash dispersion models, therefore playing an important role in risk mitigation for aviation. Future work could also explore the combination of these data with other data streams, such as infrasound and seismics, to perform multi-parametric studies into explosion dynamics.

When deploying the models on a more difficult cloud-covered scene, we highlight that they fail to perform accurately. We therefore finally note that this work, and the generalisation capabilities of the models presented herein, would strongly benefit from a larger training dataset. However, manual labelling of imagery would require considerable effort/time to go beyond the 120 images labelled herein. As deep learning and computer vision continue to become more important in the geosciences, we propose that generating a comprehensive labelled image dataset of volcanological phenomena may be a very worthwhile undertaking.

Authorship contribution statement

T. C. Wilkes: devised the research idea, performed model development/testing/application, and wrote the manuscript. T. D. Pering: acquired the model test data and contributed to the manuscript. A. J. S. McGonigle: devised the research idea and contributed to the manuscript.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

T.C.W. was supported by a Leverhulme Trust Early Career Fellowship (ECF-2020-107). We would like to thank everyone involved in organizing and running the 6th workshop for the Network for Observation of Volcanic and Atmospheric Change (NOVAC), in Peru, who made the acquisition of the Sabancaya explosion video possible. We thank two anonymous reviewers, whose comments and suggestions have greatly improved the quality of the manuscript.

Appendix

Training and validation images were prepared by resizing to 320 × 320 for feeding into the CNN, except in the case of the FPN architecture, where memory limitations meant that a size of 224 × 224 was used, and PSPNet, where image dimensions must be divisible by 48, therefore 240 × 240 was used. Note that no step to preserve aspect ratio during resizing was taken; however, images with notably large ratios were first cropped to a ratio close to 1:1. We highlight that, as Fig. 4 shows, these slight distortions do not seem to significantly impact the performance of the model. All images were then normalised to a range between 0 and 1, as is common practice in deep learning tasks. Due to loading convention in the OpenCV library (Bradski, 2000), there is some confusion over images being passed to models in Red-Green-Blue (RGB) or the flipped channel Blue-Green-Red (BGR) format. We found the BGR format resulted in slightly more accurate models, although differences were negligible, therefore all models were trained with BGR images. The load_and_process_image function provided in the utils. py file, which can be found in our model repository (https://doi.org/10.15131/shef.data.17061509), automatically controls this channel flipping to provide users with a straightforward means of preparing their data.

The convolutional neural network (CNN) was built using the Segmentation Models API (Yakubovskiy, 2019), which provides access to 4 model architectures (UNet, FPN, Linknet, PSPNet) with 25 backbones (encoders) available for each architecture. Furthermore, within these 100 models there is the requirement to optimise hyper-parameters too, which define exactly how the model is trained (e.g., number of epochs, learning rate, loss function). Since computation time restricts the ability to perform an exhaustive sweep of all model networks and associated hyper-parameters we conducted tests in a restricted manner, with the aim of empirically finding the model most suited to volcanic plume segmentation. Note that it is very likely that a number of model designs will work for any application, and there is not necessarily a “perfect” solution; the goal is therefore to find solutions that work appropriately for a given application, rather than exhaustively looking for the best.

We identified the core model architecture likely to be most suitable by configuring each of the 4 architectures with a VGG16 backbone, pre-trained with ImageNet weights (Deng et al., 2009; Russakovsky et al., 2015), then training the model on our data. VGG16 (Simonyan and Zisserman, 2015) was chosen in this first step as it is generally a popular encoder (e.g., Shin et al., 2016). 50 epochs were run to train the decoder, using the stochastic optimiser Adam (Kingma and Ba, 2015), a batch size of 8, and learning rate, α, of 10^{-2}; fine tuning of all parameters (with a trainable encoder) was then performed for a further 20 epochs at $\alpha = 10^{-5}$. Following the transfer learning approach, significant encoder training should not be necessary, since its weights and biases were initiated with values from training on the ImageNet database; the values are therefore expected to be capable of extracting important features from a wide range of visible images. It is not within the scope of this paper to provide a detailed explanation of each of these hyper-parameters and their role in model training, since they are well documented in most Deep Learning textbooks (e.g., Goodfellow et al., 2016). In all cases, the model training and validation accuracies had reached a plateau by the end of this training, i.e., further training would not improve the model performance and could lead to overfitting.

In each epoch the model was trained on the full dataset, both original and augmented, to ensure that a large amount of data was available in each pass. In some cases this may lead to overfitting of the model, as the model sees the same image (although with some form of augmentation applied) multiple times during an epoch. However, due to the limited size of the labelled dataset, in this case we found that allowing the model to fit to all available data in each epoch was preferable. Furthermore, from the validation accuracy it appears that this method did not lead to over-fitting of the model.

All 4 model architectures produced similar validation accuracies (97.1–99.2%), suggesting that all have good potential for this application (Table A1). ROC and precision-recall curves for the 4 architectures on the VGG16 backbone are shown in Figure A1. FPN displayed the highest $\mathrm{AUC}_{\mathrm{ROC}}$ and $\mathrm{AUC}_{\mathrm{prec}}$ scores (see section 2.3 for explanation of these metrics), with UNet achieving the second highest scores. For subsequent tests we therefore selected these two architectures. Of course, the encoder utilised will also have a significant impact on model performance, so we acknowledge that the testing of architectures here was not completely exhaustive and that PSPNet and LinkNet may achieve higher performances with the use of other encoders.

In the second stage of model identification a selection of encoders were tested to identify the optimal for this task. Again, it was not possible to comprehensively test all 25 encoders due to computation time. For instance, since MobileNet achieves a lower accuracy than other models in most segmentation tasks (Howard et al., 2017) we did not test this network. The 6 encoders tested (Table A1) gave a good spread of options available and the high performance of a number of these suggests that they will perform well in this application.

Finally, a small set of additional hyper-parameter optimisation tests were performed. Again, due to computation time, an exhaustive grid search for this optimisation was not possible, however, it has been shown that random search can perform adequately in most cases (Bergstra and Bengio, 2012). Furthermore, typically not all hyper-parameters will have a significant effect on the final model performance, thus small tests can elucidate which, if any, are significant in this design, with subsequent optimisation focussing on just these parameters (Bergstra and Bengio, 2012).

Table A2 displays the full suite of tests run for finding reasonable optimisations of hyper-parameters and model architecture. Note that performance in a number of model setups is very good (validation accuracy >0.99, AUCs >0.99), therefore there are likely a number of model setups that would be suitable for this application. Therefore, an exhaustive search of hyper-parameter space, whilst clearly requiring an unreasonable amount of computational time for this work, may also not yield much improvement on the results presented here.

The final hyper-parameters for model deployment were selected through a combination of their evaluation metric score and also a qualitative inspection of the validation image predictions. For example, the binary cross-entropy loss function in the UNet-DenseNet121 model scores the highest AUC_{prec} for this model architecture; however, it predicts a small region of false positives in the plume-free validation image, therefore the Jaccard loss function is preferred. Each model setup used for deployment is shown in bold in Table A2.

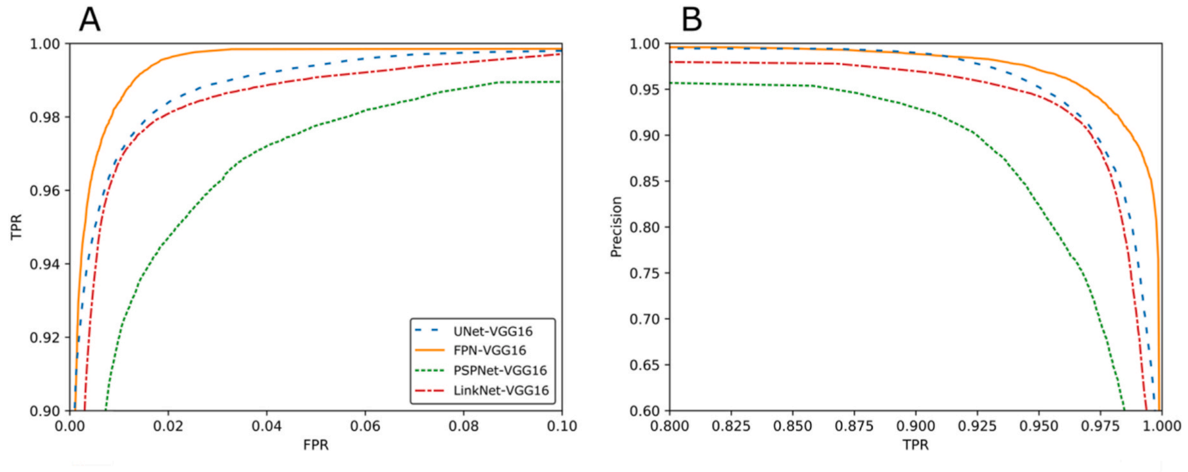


Fig. A1. (A) Receiver operator characteristic and (B) precision-recall curves for the 4 model architectures tested in this work, each with the VGG16 backbone. Typically curves are shown with axes ranges of 0–1, but due to the relative similarity in performances we have isolated the corner region of the curves where the greatest disparity can be seen. Better performing models will be closer to the left and top axes for A and right and top axes for B. Associated evaluation results are shown in Table A1.

Table A1
The evaluation of all model architectures tested. Evaluation metrics are defined in Section 2.3. Hyper-parameters were kept constant, as the values outlined in Supplementary Material A1.

Architecture	Backbone (encoder)	Batch size	Training Accuracy	Validation Accuracy	Sensitivity	Precision	IoU	AUC_{ROC}	AUC_{prec}
UNet	VGG16	8	0.9939	0.9875	0.9714	0.9058	0.8822	0.9978	0.9880
	resnet34	8	0.9939	0.9905	0.9780	0.9275	0.9086	0.9973	0.9929
	seresnet34	8	0.9943	0.9885	0.9690	0.9169	0.8908	0.9962	0.9897
	densenet121	4	0.9915	0.9920	0.9749	0.9621	0.9388	0.9981	0.9953
	efficientnetb0	8	0.9943	0.9930	0.9771	0.9519	0.9312	0.9992	0.9951
	inceptionv3	8	0.9955	0.9946	0.9729	0.9715	0.9459	0.9989	0.9962
FPN	VGG16	8	0.9939	0.9922	0.9673	0.9530	0.9233	0.9985	0.9924
	resnet34	8	0.9929	0.9935	0.9714	0.9611	0.9347	0.9992	0.9954
	seresnet34	8	0.9925	0.9932	0.9706	0.9592	0.9321	0.9988	0.9943
	densenet121	8	0.9934	0.9942	0.9703	0.9699	0.9420	0.9976	0.9947
	efficientnetb0	8	0.9941	0.9947	0.9804	0.9655	0.9472	0.9994	0.9972
	inceptionv3	8	0.9950	0.9911	0.9638	0.9452	0.9128	0.9981	0.9924
PSPNet	VGG16	8	0.9936	0.9706	0.9597	0.7839	0.7589	0.9898	0.9541
Linknet	VGG16	8	0.9934	0.9851	0.9751	0.8830	0.8636	0.9963	0.9780

Table A2
Hyper-parameter evaluations for the 3 model architectures chosen. Bold shows the configuration used for model deployment.

Architecture	Backbone (encoder)	Batch size	Epochs (main)	Epochs (fine tune)	Optomiser	Loss function	α	$α_{fine\_tune}$	Training Accuracy	Validation Accuracy	Sensitivity	Precision	IoU	AUC_{ROC}	AUC_{prec}
UNet	densenet121	4	50	20	adam	Jaccard	0.01	1.00E-05	0.9915	0.9920	0.9749	0.9621	0.9388	0.9981	0.9953
		1	50	20	adam	Jaccard	0.01	1.00E-05	0.9950	0.9681	0.9751	0.8830	0.8805	0.9963	0.9780
		4	50	20	adam	Jaccard	0.10	1.00E-05	0.9907	0.9877	0.9614	0.9150	0.8826	0.9978	0.9878
		4	50	20	SGD	Jaccard	0.01	1.00E-05	0.9922	0.9912	0.9677	0.9425	0.9138	0.9989	0.9936
		4	50	0	adam	Jaccard	0.01	–	0.9905	0.9945	0.9800	0.9632	0.9447	0.9994	0.9962
		4	**50**	**20**	**RMSProp**	**Jaccard**	**0.01**	**1.00E-05**	**0.9939**	**0.9950**	**0.9766**	**0.9718**	**0.9497**	**0.9992**	**0.9969**
		4	50	20	adam	BCE	0.01	1.00E-05	0.9900	0.9940	0.9732	0.9648	0.9399	0.9997	0.9970
		4	50	50	adam	Jaccard	0.01		0.9917	0.9919	0.9589	0.9568	0.9191	0.9980	0.9929

(continued on next page)

Table A2 (*continued*)

Architecture	Backbone (encoder)	Batch size	Epochs (main)	Epochs (fine tune)	Optomiser	Loss function	α	α_{fine_tune}	Training Accuracy	Validation Accuracy	Sensitivity	Precision	IoU	AUC_{ROC}	AUC_{prec}
								1.00E-03							
		4	100	0	adam	Jaccard	0.01	–	0.9943	0.9936	0.9620	0.9710	0.9351	0.9990	0.9946
		8	50	20	adam	Jaccard	0.01	1.00E-05	0.9955	0.9946	0.9729	0.9715	0.9459	0.9989	0.9962
		1	50	20	adam	Jaccard	0.01	1.00E-05	0.9934	0.8072	0.9362	0.8126	0.7699	0.9917	0.9558
		8	50	20	adam	Jaccard	0.10	1.00E-05	0.9878	0.9387	0.9754	0.9624	0.9396	0.9991	0.9956
		8	50	20	SGD	Jaccard	0.01	1.00E-05	0.9892	0.9848	0.9686	0.8844	0.8597	0.9964	0.9817
UNet	inceptionv3	8	50	0	adam	Jaccard	0.01	–	0.9957	0.9936	0.9787	0.9556	0.9361	0.9987	0.9934
		8	50	20	RMSProp	Jaccard	0.01	1.00E-05	0.9965	0.9928	0.9777	0.9490	0.9289	0.9987	0.9944
		8	**50**	**20**	**adam**	**BCE**	**0.01**	**1.00E-05**	**0.9959**	**0.9941**	**0.9786**	**0.9606**	**0.9409**	**0.9993**	**0.9966**
		8	50	50	adam	Jaccard	0.01	1.00E-03	0.9881	0.9885	0.9042	0.9747	0.8834	0.9978	0.9904
		8	100	0	adam	Jaccard	0.01	–	0.9968	0.9936	0.9691	0.9650	0.9362	0.9965	0.9922
FPN	efficientnetb0	8	50	20	adam	Jaccard	0.01	1.00E-05	0.9941	0.9947	0.9804	0.9655	0.9472	0.9994	0.9972
		1	50	20	adam	Jaccard	0.01	1.00E-05	0.9912	0.9658	0.7058	0.9209	0.6655	0.8985	0.8789
		8	50	20	adam	Jaccard	0.10	1.00E-05	0.9879	0.9932	0.9717	0.9586	0.9326	0.9979	0.9928
		8	50	20	SGD	Jaccard	0.01	1.00E-05	0.9789	0.9830	0.9667	0.8708	0.8454	0.9971	0.9828
		8	50	0	adam	Jaccard	0.01	–	0.9911	0.9936	0.9791	0.9554	0.9363	0.9993	0.9956
		8	**50**	**20**	**RMSProp**	**Jaccard**	**0.01**	**1.00E-05**	**0.9947**	**0.9952**	**0.9838**	**0.9667**	**0.9515**	**0.9991**	**0.9974**
		8	50	20	adam	BCE	0.01	1.00E-05	0.9914	0.9938	0.9806	0.9565	0.9387	0.9996	0.9968
		8	50	50	adam	Jaccard	0.01	1.00E-03	0.9954	0.9940	0.9741	0.9642	0.9401	0.9983	0.9956
		8	100	0	adam	Jaccard	0.01	–	0.9941	0.9893	0.9729	0.9206	0.8976	0.9964	0.9844

SGD - Stochastic gradient descent, RMSProp - Root mean squared propogation, BCE - Binary cross-entropy, α - learning rate.

Table A3
Performance metrics of cross-validation model runs.

Model	Validation set	Training Accuracy	Validation Accuracy	Sensitivity	Precision	IOU	AUC_{ROC}	AUC_{prec}
Unet-DenseNet121	1	0.9924	0.9600	0.9933	0.9428	0.9368	0.9996	0.9975
	2	0.9932	0.9418	0.9704	0.9873	0.9584	0.9996	0.9984
	3	0.9937	0.9656	0.9929	0.9579	0.9514	0.9998	0.9984
	4	0.9929	0.9824	0.9809	0.9809	0.9625	0.9994	0.9983
	5	0.9934	0.9777	0.9638	0.9926	0.9570	0.9993	0.9984
	6	0.9934	0.9786	0.9819	0.9220	0.9065	0.9989	0.9945
	7	0.9934	0.9832	0.9862	0.9722	0.9592	0.9997	0.9973
	8	0.9929	0.9841	0.9745	0.9854	0.9607	0.9990	0.9977
	9	0.9940	0.9589	0.9894	0.7892	0.7826	0.9984	0.9872
	10	0.9939	0.9767	0.9878	0.9724	0.9608	0.9997	0.9985
	Mean	**0.9933**	**0.9709**	**0.9821**	**0.9503**	**0.9336**	**0.9993**	**0.9966**
	SD	*0.0005*	*0.0132*	*0.0094*	*0.0576*	*0.0529*	*0.0004*	*0.0033*
Unet-Inceptionv3	1	0.9956	0.9834	0.9765	0.9922	0.9690	0.9993	0.9987
	2	0.9962	0.9615	0.9974	0.9619	0.9595	0.9999	0.9994
	3	0.9959	0.9743	0.9881	0.9881	0.9765	0.9999	0.9995
	4	0.9950	0.9811	0.9857	0.9872	0.9733	0.9999	0.9994
	5	0.9960	0.9835	0.9905	0.9819	0.9727	0.9999	0.9993
	6	0.9942	0.9907	0.9716	0.9907	0.9628	0.9999	0.9990
	7	0.9916	0.9524	0.9863	0.7364	0.7289	0.9934	0.9581
	8	0.9944	0.9832	0.9515	0.9962	0.9480	0.9999	0.9991
	9	0.9763	0.9325	0.9964	0.6854	0.6837	0.9860	0.8974
	10	0.9965	0.9793	0.9922	0.9877	0.9801	0.9999	0.9993
	Mean	**0.9932**	**0.9722**	**0.9836**	**0.9308**	**0.9155**	**0.9978**	**0.9849**
	SD	*0.0058*	*0.0171*	*0.0132*	*0.1109*	*0.1054*	*0.0044*	*0.0317*
FPN-EfficientNetb0	1	0.9946	0.9808	0.9767	0.9883	0.9655	0.9992	0.9984
	2	0.9949	0.9660	0.9925	0.9699	0.9628	0.9998	0.9986
	3	0.9947	0.9762	0.9847	0.9854	0.9705	0.9995	0.9987
	4	0.9947	0.9790	0.9925	0.9695	0.9625	0.9997	0.9987
	5	0.9947	0.9824	0.9905	0.9768	0.9678	0.9997	0.9982
	6	0.9946	0.9837	0.9778	0.9321	0.9128	0.9975	0.9893
	7	0.9942	0.9791	0.9773	0.9832	0.9612	0.9987	0.9975

(*continued on next page*)

Table A3 (*continued*)

Model	Validation set	Training Accuracy	Validation Accuracy	Sensitivity	Precision	IOU	AUC_{ROC}	AUC_{prec}
	8	0.9941	0.9878	0.9921	0.9694	0.9620	0.9984	0.9979
	9	0.9945	0.9816	0.9866	0.9741	0.9614	0.9996	0.9985
	10	0.9949	0.9813	0.9861	0.9798	0.9665	0.9995	0.9983
	Mean	**0.9946**	**0.9798**	**0.9857**	**0.9729**	**0.9593**	**0.9992**	**0.9974**
	SD	*0.0003*	*0.0055*	*0.0061*	*0.0150*	*0.0158*	*0.0007*	*0.0027*

Table A4
Confusion matrix for UNet-densenet121 for the 6 validation images. Values are number of pixels.

		Ground-truth		
		Plume	No plume	*Total*
Prediction	Plume	58189	1033	*59222*
	No Plume	1057	554121	*555178*
	Total	*59246*	*555154*	

Table A5
Confusion matrix for UNet-inceptionv3 for the 6 validation images. Values are number of pixels.

		Ground-truth		
		Plume	No plume	*Total*
Prediction	Plume	57914	1545	*59459*
	No Plume	1332	553609	*554941*
	Total	*59246*	*555154*	

Table A6
Confusion matrix for FPN-efficientnetb0 for the 6 validation images. Values are number of pixels. Note, due to memory limitations, this network was trained/validated on lower resolution images relative to the UNet models in Table A3 and A4 (see Appendix for more details). Therefore, the total number of pixels in this matrix is lower.

		Ground-truth		
		Plume	No plume	*Total*
Prediction	Plume	28666	361	*29027*
	No Plume	363	271666	*272029*
	Total	*29029*	*272027*	

References

Albadra, A., Wood, K., Berthoud, L., Calway, A., Watson, M., Thomas, H., Richardson, T., Liu, E., Chigna, G., 2020. Determining the three-dimensional structure of a volcanic plume using Unoccupied Aerial System (UAS) imagery. J. Volcanol. Geoth. Res. 407, 106731 https://doi.org/10.1016/j.jvolgeores.2019.106731.

Alexander, D., 2013. Volcanic ash in the atmosphere and risks for civil aviation: a study in European crisis management. Int. J. Disaster Risk Sci. 4, 9–19. https://doi.org/10.1007/s13753-013-0003-0.

Bear-Crozier, A.N., Kartadinata, N., Heriwaseso, A., Nielsen, O., 2012. Development of python-FALL3D: a modified procedure for modelling volcanic ash dispersal in the Asia-Pacific region. Nat. Hazards 64, 821–838. https://doi.org/10.1007/s11069-012-0273-7.

Beckett, F.M., Witham, C.S., Leadbetter, S.J., Crocker, R., Webster, H.N., Hort, M.C., Jones, A.R., Devenish, B.J., Thomson, D.J., 2020. Atmospheric dispersion modelling at the London VAAC: a review of developments since the 2010 eyjafjallajökull volcano ash cloud. Atmosphere 11. https://doi.org/10.3390/atmos11040352.

Bergen, K.J., Johnson, P.A., De Hoop, M.V., Beroza, G.C., 2019. Machine learning for data-driven discovery in solid Earth geoscience. Science 80, 363. https://doi.org/10.1126/science.aau0323.

Bergstra, J., Bengio, Y., 2012. Random search for hyper-parameter optimization. J. Mach. Learn. Res. 13, 281–305.

Bombrun, M., Jessop, D., Harris, A., Barra, V., 2018. An algorithm for the detection and characterisation of volcanic plumes using thermal camera imagery. J. Volcanol. Geoth. Res. 352, 26–37. https://doi.org/10.1016/j.jvolgeores.2018.01.006.

Bradley, A.P., 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recogn. 30, 1145–1159. https://doi.org/10.1016/S0031-3203(96)00142-2.

Bradski, G., 2000. The OpenCV Library. Dr. Dobb's Journal of Software Tools.

Brown, S.K., Jenkins, S.F., Sparks, R.S.J., Odbert, H., Auker, M.R., 2017. Volcanic fatalities database: analysis of volcanic threat with distance and victim classification. J. Appl. Volcanol. 6 https://doi.org/10.1186/s13617-017-0067-4.

Buslaev, A., Iglovikov, V.I., Khvedchenya, E., Parinov, A., Druzhinin, M., Kalinin, A.A., 2020. Albumentations: fast and flexible image augmentations. Information 11, 1–20. https://doi.org/10.3390/info11020125.

Campion, R., Delgado-Granados, H., Legrand, D., Taquet, N., Boulesteix, T., Pedraza-Espitía, S., Lecocq, T., 2018. Breathing and coughing: the extraordinarily high degassing of popocatépetl volcano investigated with an SO2 camera. Front. Earth Sci. 6 https://doi.org/10.3389/feart.2018.00163.

Cassidy, M., Cole, P.D., Hicks, K.E., Varley, N.R., Peters, N., Lerner, A.H., 2015. Rapid and slow: varying magma ascent rates as a mechanism for Vulcanian explosions. Earth Planet Sci. Lett. 420, 73–84. https://doi.org/10.1016/j.epsl.2015.03.025.

Chaurasia, A., Culurciello, E., 2018. LinkNet: exploiting encoder representations for efficient semantic segmentation. In: 2017 IEEE Visual Communications and Image Processing. https://doi.org/10.1109/VCIP.2017.8305148. VCIP 2017 2018-Janua, 1–4.

Costa, A., Suzuki, Y.J., Cerminara, M., Devenish, B.J., Ongaro, T.E., Herzog, M., et al., 2016. Results of the eruptive column model inter-comparison study. J. Volcanol. Geoth. Res. 326, 2–25. https://doi.org/10.1016/j.jvolgeores.2016.01.017.

Daggitt, M.L., Mather, T.A., Pyle, D.M., Page, S., 2014. AshCalc-a new tool for the comparison of the exponential, power-law and Weibull models of tephra deposition. J. Appl. Volcanol. 3, 1–8. https://doi.org/10.1186/2191-5040-3-7.

David, C.H., Famiglietti, J.S., Yang, Z.-L., Habets, F., Maidment, D.R., 2016. A decade of RAPID-Reflections on the development of an open source geoscience code. Earth Space Sci. 3, 226–244. https://doi.org/10.1002/2015EA000142.

Dempsey, D.E., Cronin, S.J., Mei, S., Kempa-Liehr, A.W., 2020. Automatic precursor recognition and real-time forecasting of sudden explosive volcanic eruptions at

Whakaari, New Zealand. Nat. Commun. 11, 1–8. https://doi.org/10.1038/s41467-020-17375-2.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L., 2009. ImageNet: a large-scale hierarchical image database. In: IEEE Conference on Computer Vision and Pattern Recognition. IEEE, pp. 248–255. https://doi.org/10.1109/CVPRW.2009.5206848, 2009.

Diaz Moreno, A., Bueno Rodriguez, A., Zuccarello, L., Woolam, J., Titos Luzón, M., Benitez, C., Álvarez, I.S., Prudencio, J., De Angelis, S., Ibáñez, J.~M., 2019. PICOSS: Python interface for the classification of seismic signals. In: AGU Fall Meeting Abstracts. NG21B-0954.

Dye, B.C., Morra, G., 2020. Machine learning as a detection method of Strombolian eruptions in infrared images from Mount Erebus, Antarctica. Phys. Earth Planet. In. 305, 106508 https://doi.org/10.1016/j.pepi.2020.106508.

Fee, D., Izbekov, P., Kim, K., Yokoo, A., Lopez, T., Prata, F., Kazahaya, R., Nakamichi, H., Iguchi, M., 2017. Eruption mass estimation using infrasound waveform inversion and ash and gas measurements: evaluation at Sakurajima Volcano, Japan. Earth Planet Sci. Lett. 480, 42–52. https://doi.org/10.1016/j.epsl.2017.09.043.

Fenner, D., Ruempker, G., Li, W., Chakraborty, M., Faber, J., Koehler, J., Stoecker, H., Srivastava, N., 2021. AWESAM: A Python Module for Automated Volcanic Event Detection Applied to Stromboli.

Fitzgerald, R.H., Kennedy, B.M., Gomez, C., Wilson, T.M., Simons, B., Leonard, G.S., Matoza, R.S., Jolly, A.D., Garaebiti, E., 2020. Volcanic ballistic projectile deposition from a continuously erupting volcano: Yasur Volcano, Vanuatu. Volcanica 3, 183–204. https://doi.org/10.30909/vol.03.02.183204.

Gaudin, D., Moroni, M., Taddeucci, J., Scarlato, P., Shindler, L., 2014. Pyroclast Tracking Velocimetry: a particle tracking velocimetry-based tool for the study of Strombolian explosive eruptions. J. Geophys. Res. Solid Earth 119, 5369–5383. https://doi.org/10.1002/2014JB011096.

Gerst, A., Hort, M., Aster, R.C., Johnson, J.B., Kyle, P.R., 2013. The first second of volcanic eruptions from the Erebus volcano lava lake, Antarctica-Energies, pressures, seismology, and infrasound. J. Geophys. Res. Solid Earth 118, 3318–3340. https://doi.org/10.1002/jgrb.50234.

Gliß, J., Stebel, K., Kylling, A., Dinger, A., Sihler, H., Sudbø, A., 2017. Pyplis - a Python software toolbox for the analysis of SO2 camera data. Implications in geosciences. Geosciences 7, 134. https://doi.org/10.3390/geosciences7040134.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.

Hort, M., Avard, G., Gast, E., Markmann, P., Scharff, L., 2018. Monitoring the explosive activity of turrialba volcano, Costa Rica, using Doppler radar and webcam observations. In: EGU General Assembly Conference Abstracts EGU General Assembly Conference Abstracts, p. 9845.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. MobileNets: efficient convolutional neural networks for mobile vision applications. arXiv. https://doi.org/10.48550/ARXIV.1704.04861.

Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q., 2017. Densely connected convolutional networks. In: Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017 2017-Janua, pp. 2261–2269. https://doi.org/10.1109/CVPR.2017.243.

Johnson, J.B., Harris, A.J.L., Sahetapy-Engel, S.T.M., Wolf, R., Rose, W.I., 2004. Explosion dynamics of pyroclastic eruptions at Santiaguito Volcano. Geophys. Res. Lett. 31, 1–5. https://doi.org/10.1029/2003gl019079.

Kingma, D.P., Ba, J.L., 2015. Adam: a method for stochastic optimization. 3rd international conference on learning representations. ICLR 2015 - Conf. Track Proc. 1–15.

Krawczyk, B., 2016. Learning from imbalanced data: open challenges and future directions. Prog. Artif. Intell. 5, 221–232. https://doi.org/10.1007/s13748-016-0094-0.

LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521, 436–444. https://doi.org/10.1038/nature14539.

Li, D., Yang, J., Kreis, K., Torralba, A., Fidler, S., 2021. Semantic segmentation with generative models: semi-supervised learning and strong out-of-domain generalization. CoRR. https://doi.org/10.1109/cvpr46437.2021.00820.

Lin, T-.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S., 2016. Feature pyramid networks for object detection. arXiv. https://doi.org/10.48550/ARXIV.1612.03144.

Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C. L., 2014. Microsoft COCO: common objects in context, pp. 740–755. https://doi.org/10.1007/978-3-319-10602-1_48. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 8693 LNCS.

Manley, G.F., Mather, T.A., Pyle, D.M., Clifton, D.A., Rodgers, M., Thompson, G., Roman, D.C., 2021. Machine learning approaches to identifying changes in eruptive state using multi-parameter datasets from the 2006 eruption of Augustine Volcano, Alaska. J. Geophys. Res. Solid Earth. https://doi.org/10.1029/2021jb022323.

Matoza, R.S., Chouet, B.A., Jolly, A.D., Dawson, P.B., Fitzgerald, R.H., Kennedy, B.M., et al., 2022. High-rate very-long-period seismicity at Yasur volcano, Vanuatu: source mechanism and decoupling from surficial explosions and infrasound. Geophys. J. Int. 230, 392–426. https://doi.org/10.1093/gji/ggab533.

Orr, T.R., Thelen, W.A., Patrick, M.R., Swanson, D.A., Wilson, D.C., 2013. Explosive eruptions triggered by rockfalls at Kīlauea volcano, Hawai'i. Geology 41, 207–210. https://doi.org/10.1130/G33564.1.

Peters, N., Oppenheimer, C., 2018. Plumetrack: flux calculation software for UV cameras. Comput. Geosci. 118, 86–90. https://doi.org/10.1016/j.cageo.2018.05.014.

Rahman, M.A., Wang, Y., 2016. Optimizing intersection-over-union in deep neural networks for image segmentation. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Porikli, F., Skaff, S., et al. (Eds.), Advances in Visual Computing. Springer International Publishing), Cham, pp. 234–244.

Razavian, A.S., Azizpour, H., Sullivan, J., Carlsson, S., 2014. CNN features off-the-shelf: an astounding baseline for recognition. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog. Workshops 512–519. https://doi.org/10.1109/CVPRW.2014.131.

Ren, C.X., Peltier, A., Ferrazzini, V., Rouet-Leduc, B., Johnson, P.A., Brenguier, F., 2020. Machine learning reveals the seismic signature of eruptive behavior at piton de la Fournaise volcano. Geophys. Res. Lett. 47, 1–11. https://doi.org/10.1029/2019GL085523.

Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: convolutional networks for biomedical image segmentation. IEEE Access 9, 16591–16603. https://doi.org/10.1109/ACCESS.2021.3053408.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L., 2015. ImageNet large scale visual recognition challenge. Int. J. Comput. Vis. 115, 211–252. https://doi.org/10.1007/s11263-015-0816-y.

Saito, T., Rehmsmeier, M., 2015. The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PLoS One 10, 1–21. https://doi.org/10.1371/journal.pone.0118432.

Salman, S., Liu, X., 2019. Overfitting Mechanism and Avoidance in Deep Neural Networks.

Scollo, S., Folch, A., Costa, A., 2008. A parametric and comparative study of different tephra fallout models. J. Volcanol. Geoth. Res. 176, 199–211. https://doi.org/10.1016/j.jvolgeores.2008.04.002.

Scollo, S., Prestifilippo, M., Bonadonna, C., Cioni, R., Corradini, S., Degruyter, W., et al., 2019. Near-real-time tephra fallout assessment at Mt. Etna, Italy. Rem. Sens. 11, 1–18. https://doi.org/10.3390/rs11242987.

Shin, H.C., Roth, H.R., Gao, M., Lu, L., Xu, Z., Nogues, I., Yao, J., Mollura, D., Summers, R.M., 2016. Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. IEEE Trans. Med. Imag. 35, 1285–1298. https://doi.org/10.1109/TMI.2016.2528162.

Shorten, C., Khoshgoftaar, T.M., 2019. A survey on image data augmentation for deep learning. J. Big Data 6. https://doi.org/10.1186/s40537-019-0197-0.

Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition. 3rd International Conference on Learning Representations. ICLR 2015 - Conf. Track Proc. 1–14.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., 2015. Going deeper with convolutions. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn. 1–9. https://doi.org/10.1109/CVPR.2015.7298594, 07-12-June.

Taddeucci, J., Palladino, D.M., Camaldo, C., Scarlato, P., Palldino, D.M., Camaldo, C., Scarlato, P., 2012. High-speed Imaging of Strombolian Eruptions Reveals Gas-Pyroclast Interactions in Volcanic Jets, vol. 14, p. 9132. https://doi.org/10.1002/2015GL064874.Received.

Tajbakhsh, N., Shin, J.Y., Gurudu, S.R., Hurst, R.T., Kendall, C.B., Gotway, M.B., Liang, J., 2016. Convolutional neural networks for medical image analysis: full training or fine tuning? IEEE Trans. Med. Imag. 35, 1299–1312. https://doi.org/10.1109/TMI.2016.2535302.

Tan, M., Le, Q.V., 2019. EfficientNet: rethinking model scaling for convolutional neural networks. In: 36th International Conference on Machine Learning. ICML, pp. 10691–10700, 2019 2019-June.

Tournigand, P.Y., Taddeucci, J., Gaudin, D., Peña Fernández, J.J., Del Bello, E., Scarlato, P., et al., 2017. The initial development of transient volcanic plumes as a function of source conditions. J. Geophys. Res. Solid Earth 122, 9784–9803. https://doi.org/10.1002/2017JB014907.

Tournigand, P.Y., Fernández, J.J.P., Taddeucci, J., Perugini, D., Sesterhenn, J., Palladino, D.M., 2019. Time evolution of transient volcanic plumes: insights from fractal analysis. J. Volcanol. Geoth. Res. 371, 59–71. https://doi.org/10.1016/j.jvolgeores.2018.12.007.

Valade, S.A., Harris, A.J.L., Cerminara, M., 2014. Plume Ascent Tracker: interactive Matlab software for analysis of ascending plumes in image data. Comput. Geosci. 66, 132–144. https://doi.org/10.1016/j.cageo.2013.12.015.

Witsil, A.J.C., Johnson, J.B., 2020. Volcano video data characterized and classified using computer vision and machine learning algorithms. Geosci. Front. https://doi.org/10.1016/j.gsf.2020.01.016.

Witt, T., Walter, T.R., Müller, D., Guðmundsson, M.T., Schöpa, A., 2018. The relationship between lava fountaining and vent morphology for the 2014–2015 holuhraun eruption, Iceland, analyzed by video monitoring and topographic mapping. Front. Earth Sci. 6 https://doi.org/10.3389/feart.2018.00235.

Wood, K., Thomas, H., Watson, M., Calway, A., Richardson, T., Stebel, K., Naismith, A., Berthoud, L., Lucas, J., 2019. Measurement of three dimensional volcanic plume properties using multiple ground based infrared cameras. ISPRS J. Photogrammetry Remote Sens. 154, 163–175. https://doi.org/10.1016/j.isprsjprs.2019.06.002.

Yakubovskiy, P., 2019. Segmentation models. GitHub Repository. https://github.com/qubvel/segmentation_models.

Yamada, T., Ueda, H., Mori, T., Tanada, T., 2019. Tracing volcanic activity chronology from a multiparameter dataset at Shinmoedake Volcano (Kirishima), Japan. J. Disaster Res. 14, 687–700. https://doi.org/10.20965/jdr.2019.p0687.

Yamamoto, H., Watson, I.M., Phillips, J.C., Bluth, G.J.S., 2008. Rise dynamics and relative ash distribution in vulcanian eruption plumes at Santiaguito Volcano, Guatemala, revealed using an ultraviolet imaging camera. Geophys. Res. Lett. 35, 1–5. https://doi.org/10.1029/2007GL032008.

Zhao, T., Jayaram, V., Roy, A., Marfurt, K.J., 2015. A comparison of classification techniques for seismic facies recognition. Interpretation 3. https://doi.org/10.1190/INT-2015-0044.1. SAE29–SAE58.

Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J., 2017. Pyramid scene parsing network. In: Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017 2017-Janua, pp. 6230–6239. https://doi.org/10.1109/CVPR.2017.660.

Zhu, Y., Zhang, Zhongyue, Wu, C., Zhang, Zhi, He, T., Zhang, H., Manmatha, R., Li, M., Smola, A., 2020. Improving Semantic Segmentation via Self-Training arXiv.