



Emulation of wildland fire spread simulation using deep learning

Frédéric Allaire, Vivien Mallet, Jean-Baptiste Filippi

► To cite this version:

Frédéric Allaire, Vivien Mallet, Jean-Baptiste Filippi. Emulation of wildland fire spread simulation using deep learning. *Neural Networks*, 2021, 141, pp.184-198. 10.1016/j.neunet.2021.04.006 . hal-03473257

HAL Id: hal-03473257

<https://hal.science/hal-03473257>

Submitted on 9 May 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial 4.0 International License

1 Emulation of wildland fire spread simulation 2 using deep learning

3 Frédéric Allaire^{A,*}, Vivien Mallet^A, and Jean-Baptiste Filippi^B

4 ^A Institut national de recherche en informatique et en automatique (INRIA), 2 rue
5 Simone Iff, Paris, France; Sorbonne Université, Laboratoire Jacques-Louis Lions,
6 France.

7 ^B Centre national de la recherche scientifique (CNRS), Sciences pour l'Environnement
8 – Unité Mixte de Recherche 6134, Università di Corsica, Campus Grossetti, Corte,
9 France.

10 * Email: frederic.allaire@inria.fr; corresponding author.

11 **Abstract:**

12 Numerical simulation of wildland fire spread is useful to predict the loca-
13 tions that are likely to burn and to support decision in an operational con-
14 text, notably for crisis situations and long-term planning. For short-term,
15 the computational time of traditional simulators is too high to be tractable
16 over large zones like a country or part of a country, especially for fire danger
17 mapping.

18 This issue is tackled by emulating the area of the burned surface returned
19 after simulation of a fire igniting anywhere in Corsica island and spreading
20 freely during one hour, with a wide range of possible environmental input
21 conditions. A deep neural network with a hybrid architecture is used to
22 account for two types of inputs: the spatial fields describing the surrounding
23 landscape and the remaining scalar inputs.

24 After training on a large simulation dataset, the network shows a satis-
25 factory approximation error on a complementary test dataset with a MAPE
26 of 32.8%. The convolutional part is pre-computed and the emulator is de-
27 fined as the remaining part of the network, saving significant computational
28 time. On a 32-core machine, the emulator has a speed-up factor of several
29 thousands compared to the simulator and the overall relationship between
30 its inputs and output is consistent with the expected physical behavior of
31 fire spread. This reduction in computational time allows the computation
32 of one-hour burned area map for the whole island of Corsica in less than a
33 minute, opening new application in short-term fire danger mapping.

34

35 **Keywords:** deep neural network, hybrid architecture, mixed inputs, numer-
36 ical simulation, fire growth prediction, Corsica

37

38 1 Introduction

39 A major purpose of mathematical modeling and numerical simulation of
40 wildland fire spread across land is to make relevant predictions and sup-
41 port long-term to short-term planning of firefighting actions. Fundamen-
42 tally, fire spread implies heat transfer at scales of the centimeter, which is
43 too computationally intensive to solve in operational conditions. Alterna-
44 tively, fire spread modeling can be approached by solving a front-tracking
45 problem where we focus on the propagation of the interface between burned
46 and not burned areas, aka the *fire front*, over a 2D domain that represents
47 the landscape. The growth of the burned surfaces from their initial state is
48 governed by equations involving a model of rate of spread (ROS), that is to
49 say the speed at which the flames advance, which is expressed as a function of
50 local environmental parameters. Among such solvers, marker methods con-
51 sist in discretizing the fire front by means of markers, which evolve in space
52 and time according to an underlying fire behavior model that determines the
53 speed at which the markers advance as well as other characteristics such as
54 reaction intensity. Notable examples of simulators using this method include

55 FARSITE (Finney, 1998), Prometheus (Tymstra, Bryce, Wotton, Taylor,
56 & Armitage, 2010), and Phoenix (Tolhurst, Shields, & Chong, 2008), that
57 are commonly used in the US, Canada, and Australia, respectively. Alter-
58 natively, level-set methods (e.g. Mallet, Keyes, & Fendell, 2009; Rochoux,
59 Ricci, Lucor, Cuenot, & Trouvé, 2014) can be used in simulations to track
60 the fire front, and other approaches were proposed to model fire spread, such
61 as cell-based simulations (e.g. Johnston, Kelso, & Milne, 2008) that adopt a
62 raster representation of the burned surface (see Sullivan, 2009b, for a detailed
63 review of simulation models). Most of these approaches allow to simulate a
64 fire propagating during more than an hour in a computational time of about
65 a minute or less.

66 Physical models of wildland fire spread (Sullivan, 2009a), more complex
67 and typically including heat transfer conservation laws, equations describing
68 combustion chemistry, etc. have also been developed. However, their use is
69 generally limited to research purposes, because the computational time for
70 simulations based on such models is prohibitory in an operational context,
71 even more so for large wildfires that may burn during several hours or even
72 days and scale up to thousands of hectares.

73 Evaluation of simulators of wildland fire spread can be carried out based
74 on the comparison of an observed burned surfaces with its simulated counter-
75 part. Several evaluation metrics have been proposed in wildland fire research,
76 for instance relying on how much the two surfaces intersect (e.g. Duff, Chong,
77 & Tolhurst, 2016; Filippi, Mallet, & Nader, 2013), on the distance between

78 the vertices of the two fire perimeters (Duff, Chong, Taylor, & Tolhurst, 2012;
79 Fujioka, 2002), or on information regarding the growth of the simulated and
80 observed burned surfaces over time (Filippi et al., 2013). These metrics can
81 be computed for observed fires, in which case the simulations can make use of
82 data known in hindsight, such as fire suppression actions or observed weather
83 variables (e.g. Duff et al., 2018; Salis et al., 2016), or be simply based on data
84 available at the time of fire start (e.g. Filippi, Mallet, & Nader, 2014). ROS
85 models, which may be involved in fire spread simulators, can also be evalu-
86 ated based on observations obtained from laboratory or outdoor experiments
87 or even from observed wildfires (Cruz & Alexander, 2014).

88 Given the complexity of wildland fires, there are significant uncertainty
89 sources in modeling that may lead to considerable difficulties in determining
90 the most appropriate decisions in an operational context (Thompson, Calkin,
91 Scott, & Hand, 2017). In particular, for prediction purposes, there is con-
92 siderable input uncertainty, which can refer to a range of possible values for
93 a given model parameter or data source, possibly due to the use of weather
94 forecasts or difficulty to estimate a single “best” value. In control theory,
95 parameter uncertainty can be expressed by means of uncertainty matrices in
96 the model (L. Zhou, Tao, Paszke, Stojanovic, & Yang, 2020) to design robust
97 control laws (Stojanovic, He, & Zhang, 2020; L. Zhou et al., 2020). The main
98 goal of firefighters, once a wildfire has started (i.e. in a “crisis” situation), is
99 to control the fire by means of suppression actions. These actions are difficult
100 to model, therefore, predictions of wildland fire spread using simulators usu-

ally represent free spread (i.e. firefighting actions are not accounted for, but non-burnable areas such as water bodies may halt the progression of the fire front). Uncertainty can still be accounted for in such predictions, usually by propagating the probability distributions of the uncertain inputs following a Monte Carlo (MC) approach, resulting in an ensemble of fire spread simulations (e.g. [Allaire, Filippi, & Mallet, 2020](#); [Finney, Grenfell, et al., 2011](#); [M.S. Pinto et al., 2016](#)).

There are several possible applications of simulators of wildland fire spread in an operational context. As previously mentioned, in a crisis situation, they can help in predicting where the fire will spread and optimizing the fire suppression actions and evacuation. Prior to crisis situations, fire spread simulations are a major component of risk assessment frameworks to determine what areas have the highest potential to host a large incident. Wildland fire risk quantification generally involves models describing ignition probability, the probability for a given location to be burned, and the consequences on the objects affected by fire such as properties, timber production, as well as the consequences on human lives, wildlife habitats, etc. Several studies focused on fire risk mapping at the regional or country scale ([Finney, McHugh, Grenfell, Riley, & Short, 2011](#); [Lautenberger, 2017](#); [Parisien et al., 2005](#)), where many fires are simulated to represent a fire season or year according to some probabilistic distribution of ignition and environmental conditions driving fire spread. This process may be repeated hundreds of thousands of times as part of a Monte Carlo method. The purpose of such maps is to help

124 in land management through the reduction of areas at risk in the long-term,
125 by setting up fire breaks and providing more firefighting resources such as
126 reservoirs, etc.

127 Regarding short-term planning, information for the next day or hours
128 about the areas where a fire is most likely to ignite and how far the re-
129 sulting fire may spread can be very useful in order to know what locations
130 should be monitored more closely and help in anticipating the distribution
131 of firefighting resources (firefighters, trucks, ...) across the territory. For this
132 purpose, one may focus on the quantification of “fire danger”, a term that
133 generally relates to the potential for ignition and spread of a fire at a given
134 location. Traditional fire danger indices, widely used for decision support in
135 an operational context, consist in a unitless value calculated essentially based
136 on weather variables. A notable example of such an index is the Canadian
137 Fire Weather Index (FWI [Van Wagner & Pickett, 1985](#)), used for generating
138 maps of predicted fire danger covering Europe and the Mediterranean area
139 as part of the European Forest Fire Information System (EFFIS)¹. Making
140 use of output burned surfaces of wildland fire spread simulations offers an
141 interesting alternative for quantifying fire danger: for instance, the resulting
142 fire size accounts not only for weather but also for terrain (i.e. elevation
143 and vegetation, which are also influential factors in fire spread) and can be
144 expressed in hectares, a more “concrete” quantity. Numerical simulations of
145 wildland fire spread could be used to generate high-resolution maps of fire

¹<https://effis.jrc.ec.europa.eu/>, last checked 2021.02.01

146 spread on the basis of weather forecasts; but this would require numerous
147 computations for different ignition locations, and the constraint on compu-
148 tational time would be too demanding even for simulators used for other
149 operational purposes. As a rough estimate for the region considered in the
150 present study, running one fire spread simulation with a computational of
151 one minute for each hectare of land would amount to a computational time
152 of 872,000 minutes (about 600 days) on a single processor, and even more
153 if an ensemble of simulations is considered for each hectare; which would be
154 too long even after distributing the computations on multiple processors.

155 In the aforementioned applications, and more particularly in short-term
156 fire danger mapping, a promising approach to reduce computational time
157 is to rely on an *emulator* (aka metamodel or surrogate model) to provide
158 an approximation of some quantity of interest derived from the simulator’s
159 output. The idea is to focus on this quantity and compute it much faster
160 with the emulator at the cost of some approximation error that should be
161 as low as possible. Emulation may be used in situations when a fire spread
162 model has high computational time and/or a lot of simulations or calls of a
163 given function are required. Still, emulators are rarely used in wildland fire
164 research even though their potential for reducing computational time of sim-
165 ulations appears desirable in this field. Examples include data assimilation of
166 a fire front via polynomial chaos (Rochoux et al., 2014), sensitivity analysis
167 through the computation of Sobol’ indices related to the area and shape of the
168 simulated burned surface with emulation by either Gaussian processes (GP)

169 or generalized polynomial chaos (Trucchia, Egorova, Pagnini, & Rochoux,
 170 2019), uncertainty quantification and computation of Sobol' indices regard-
 171 ing the ROS model of Rothermel (Rothermel, 1972) using high dimensional
 172 model representation methods (Liu, Hussaini, & Ökten, 2016), interpolation
 173 in a cell-based wildland fire spread simulator to quickly compute the val-
 174 ues of correction factors in the relationship between advection velocity and
 175 spread angle on the basis of pre-computed values obtained in a few given
 176 configurations using a Radial Basis Function (RBF) approach (Ghisu, Arca,
 177 Pellizzaro, & Duce, 2015). Another example outside the scope of fire spread
 178 is the emulation of some outputs of a fire emission model with GP (Katurji
 179 et al., 2015).

180 Machine learning (ML) is a very rapidly growing area of study whose
 181 methods have been used for prediction and decision-making purposes in a
 182 variety of scientific and technical fields (Jordan & Mitchell, 2015). As exem-
 183 plified by recent reviews, there has been an increasing interest in application
 184 of ML to engineering risk assessment (Hegde & Rokseth, 2020), emergency
 185 management (Chen, Liu, Bai, & Chen, 2017), and to a wide variety of topics
 186 in wildland fire science (Jain et al., 2020) as well.

187 Neural networks, in particular, appear promising to take into account the
 188 complexity of wildland fire spread. For instance, an application involving em-
 189 ulation is proposed in (T. Zhou, Ding, Ji, Yu, & Wang, 2020) where a radial
 190 basis function neural network (RBFNN) is trained to emulate the similarity
 191 index between an observed burned surface and its simulated counterpart as a

function of several ROS adjustment factors; a MC procedure is then applied to the emulator, providing parameter estimation of the adjustment factors for data assimilation of the simulated fire front. Other methods consist in using a convolutional neural network (CNN) as a surrogate for a wildland fire spread simulator to obtain a map of predicted burned areas (Hodges & Lattimer, 2019; Radke, Hessler, & Ellsworth, 2019). Data required to solve wildfire simulations have similarities with these involved in image processing as we are handling gridded maps of elevation and fuel parameters. As deep learning proved to be very appropriate to solve such image processing problems (Krizhevsky, Sutskever, & Hinton, 2012), it motivates the use of deep neural networks (DNNs) instead of traditional emulation techniques to approach emulation in wildland fire spread simulations.

In the present study, a method is proposed for the estimation of wildland fire spread in a wide variety of environmental conditions with potential for application to fire danger mapping. The quantity of interest is the burned surface area in hectares provided by a wildland fire simulator and the core of the method consists in the emulation of this output quantity using a DNN with a hybrid architecture so that both 2D and scalar input data are processed by specific layers. The present study focuses on Corsica island but the method can be extended to other regions.

The numerical simulator of wildland fire spread that is used as basis of the present work is presented in Section 2 together with the characteristics of the simulations. The strategy used to obtain the emulator is described in

215 Section 3 and the results are provided and discussed in Section 4. Conclu-
216 sions of this work are summarized in Section 5 where some perspectives of
217 application of the emulator and possible extensions to the method are also
218 mentioned.

219 2 Simulation of wildland fire spread

220 In the present study, wildland fire spread simulations are carried out with
221 the numerical solver ForeFire (Filippi, Morandini, Balbi, & Hill, 2010). Fore-
222 Fire relies on a front-tracking method where the fire front is represented by
223 Lagrangian markers that are linked to each other by a dynamic mesh. The
224 interface is discretized using an ordered list of Lagrangian markers at given
225 locations on the surface of the Earth. The interface is then tracked by ad-
226 vecting all these markers at the propagation velocity of the front and by
227 ensuring that the list of markers still holds an accurate representation of
228 the interface. In this ordered list of markers, previous and next are defined
229 by convention in the indirect direction as in Figure 1. The outward normal
230 defines the direction of propagation from burning regions toward unburned
231 regions. Although fronts are allowed to contain islands of unburned fuel,
232 they must remain simple polygons (with no self-intersection).

233 A key aspect of the simulation is the computation of rate of spread (ROS),
234 that is to say the speed at which the flames advance. Several ROS models
235 were proposed in the scientific literature. The model used in present study is

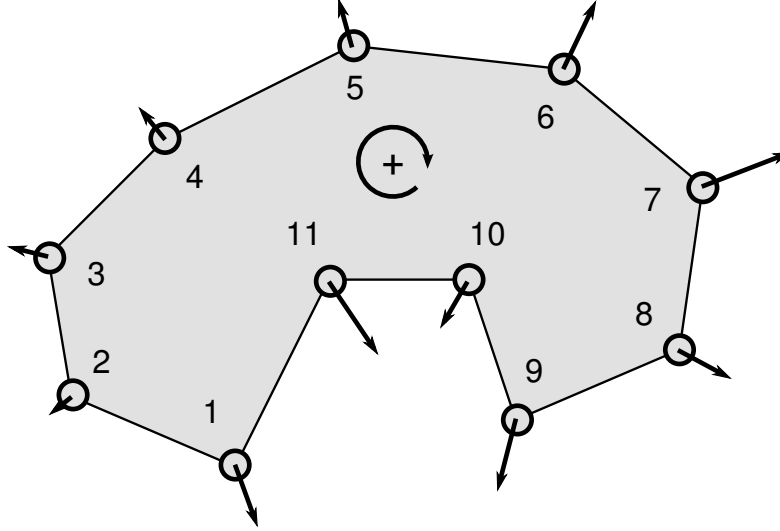
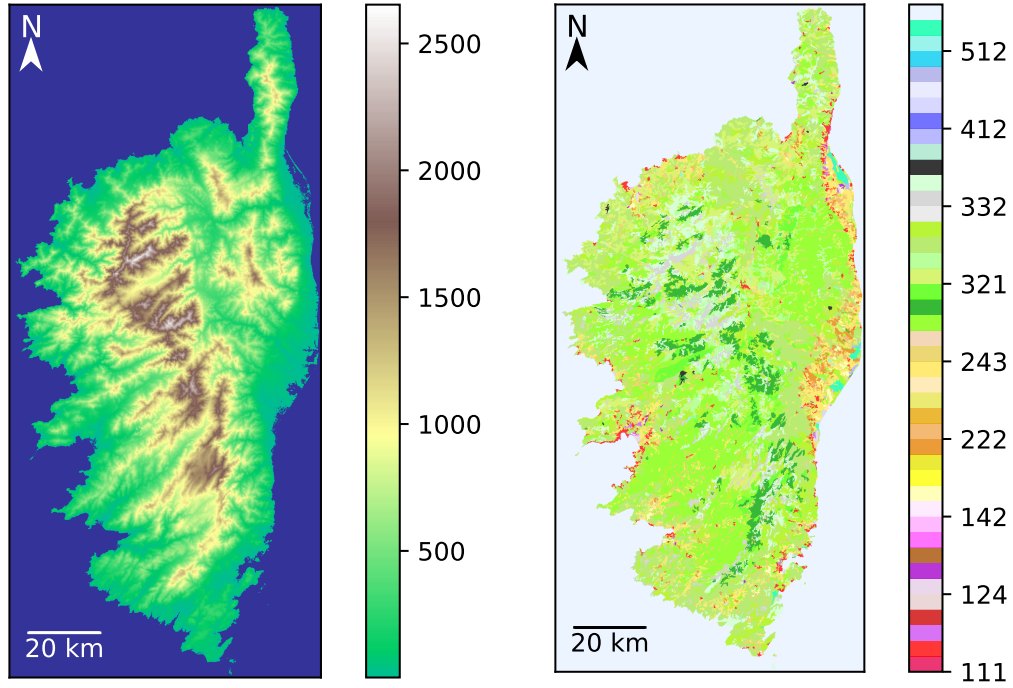


Figure 1: Example of a small fire front discretization with ordered markers. The gray area in the center represents the burned surface and its interface with the unburned locations (white) represents the fire front whose vertices (markers) expand outward along the normal to the front.

236 the model of Rothermel ([Rothermel, 1972](#)), which is commonly used by fire
 237 managers in the US. The ROS is expressed as a function of several environ-
 238 mental properties such as wind speed, terrain slope, fuel moisture content
 239 (FMC), and other fuel parameters characterizing the vegetation. A simula-
 240 tion mostly consists in the definition of an initial state of the fire front and
 241 the ROS is computed for the markers of the fire front based on underlying 2D
 242 fields, from which environmental properties are determined. ForeFire relies
 243 on a discrete event approach where most computations deal with the deter-
 244 mination of the time at which the markers will reach their next destination,

245 this destination being defined by a fixed spatial increment in the outward
246 normal. This discrete event approach includes other types of events such as
247 changes in the values of the layers, notably wind speed and FMC, additions
248 and removals of markers so that the fire front maintains a perimeter resolu-
249 tion in a given range during the simulation, and topology checks that may
250 induce front merging to ensure that the front keeps a physical representation.

251 The area of study is Corsica island, which is located south-east of France
252 in the Mediterranean sea. For fire simulation on this domain, 2D fields of
253 elevation and land use in raster format at approximately 80-m resolution
254 are used. These two fields are represented in Figures 2a and 2b. The land
255 use field comes from Corine Land Cover data (Feranec, Soukup, Hazeu, &
256 Jaffrain, 2016) coupled with data from the IGN (Institut Géographique Na-
257 tional) product BD TOPO® for road and drainage networks. The elevation
258 field is extracted from another IGN product: BD ALTI®, which originally
259 has a 25-m resolution. A fuel parameterization is used to assign reference
260 fuel parameters to each type of vegetation (referred to as “fuel type” in the
261 following) in the land use data for ROS computations. Data used for simula-
262 tion also include 2D fields of wind speed vectors at a resolution of 200 m that
263 were pre-computed for average wind speed vectors with the mass conserving
264 preconditioner from the atmospheric forecasting system Meso-NH (Lac et al.,
265 2018) to account for orographic effects. By specifying an average input wind
266 speed vector in the simulations, the underlying 2D wind field is simply ob-
267 tained from the pre-computed fields corresponding to the closest mean speed



(a) Elevation field.

(b) Land cover field.

Figure 2: Data maps of Corsica used to describe the landscape in ForeFire simulations; their spatial resolution is approximately 80 m.

(a) Locations with an altitude of 0 m or less (mostly maritime waters) are represented in blue.

(b) The color scheme corresponds to the classification of the Corine Land Cover

268 vectors.

269 In the present study, a simulation is always that of a fire with free spread
 270 during one hour. Another fixed input in the simulations is the initial fire
 271 front, which is an octagon with a surface area of 0.45 ha, corresponding to

an already-propagating fire, that must be located in areas classified as fuel (i.e. burnable vegetation) based on the land cover field.

Several inputs in the simulations may vary from a simulation to another. First are the coordinates of the center of the initial fire front, this point being referred to as the *ignition point*, that may be located in all fuel areas in Corsica. This “high-level” input is of major importance because it determines the location where the fire starts and the part of the spatial fields that will influence how the fire will spread. Next are the zonal and meridional coordinates of the “forcing” wind speed vector, in m s^{-1} , that both vary in $[-35, 35]$ on the condition that the wind speed norm be lower than 35 m s^{-1} . The FMC of dead fuel varies between 0.04 and 0.3. In contrast to these “raw” inputs, the remaining ones are perturbation coefficients that are applied to reference values of some fuel parameters. Perturbation in heat of combustion and particle density are additive and applied to a common reference value used for all fuel types, whereas perturbations in fuel height, fuel load or surface-volume ratio are multiplicative coefficients. For all the three latter parameters, each one of the 13 fuel types receives a specific perturbation coefficient. This amounts to 46 variable inputs in the simulations, whose information is summarized in Table 1, including the range of each variable.

The simulations are meant to be used for prevision of wildfire spread in Corsica before a fire starts, at any time, so the intervals of variation of the raw inputs were chosen to account for a wide variety of environmental conditions. Moreover, in this context, there is significant uncertainty in the

Input	Symbol	Unit	Type	Range	Constraint
Ignition point coordinates	(x, y)	m	Raw	Map of Corsica	Initial front in burnable area
Wind speed	(W_x, W_y)	m s^{-1}	Raw	$[-35, 35]^2$	Euclidean norm ≤ 35
Fuel moisture content (dead fuel)	m_c		Raw	$[0.04, 0.3]$	
Heat of combustion perturbation	ΔH	MJ kg^{-1}	Additive	$[-5, 5]$	
Particle density perturbation	ρ_p	kg m^{-3}	Additive	$[-300, 300]$	
Fuel height perturbations	h	m	Multiplicative	$[0.4, 1.6]^{13}$	
Fuel load perturbations	σ_f	kg m^{-2}	Multiplicative	$[0.4, 1.6]^{13}$	
Surface-volume ratio perturbations	S_v	m^{-1}	Multiplicative	$[0.4, 1.6]^{13}$	

Table 1: Variable scalar inputs in wildland fire spread simulations. In the case of perturbations, the symbol corresponds to the perturbed quantity, and the perturbation of this quantity can be either additive or multiplicative. The range indicates the boundaries of the domain of definition with two components for the wind and 13 components in the last three rows (one row per fuel type).

The intervals of variation account for uncertainty and variability of weather and ignition locations, as well as for uncertainty.

simulations. The weather forecasts used to predict wind speed and FMC are possible sources of uncertainty, so are model simplifications and the choice of a given fuel parameterization. Therefore, the intervals of variation of both raw inputs and fuel parameters also account for their uncertainty range. Some intervals follow those of a previous study that focused on uncertainty quantification (see notably Table 1 in [Allaire, Mallet, & Filippi, 2021](#)).

Finally, the quantity of interest in the present study is the area in hectares of the burned surface obtained at the end of the simulation, namely after a

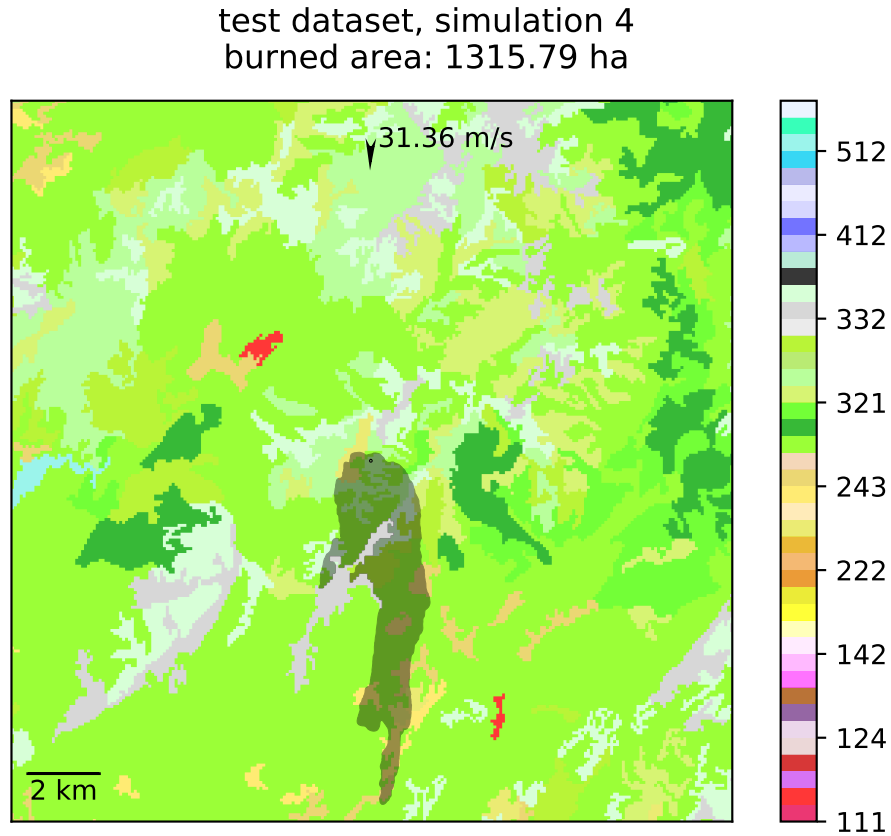


Figure 3: Example of a simulated burned surface after one hour returned by ForeFire.

The initial fire front of 0.45 ha is represented in black at the center of the figure and the final burned surface is the surrounding shaded shape. The input wind speed vector is represented by the arrow at the top. The simulated fire spread to the south, was partly blocked by mountains (in gray), but still burned 1316 ha.

Background colors correspond to the classification of the Corine Land Cover

303 free fire spread of one hour. An example of simulated surface is represented
304 in Figure 3.

305 It is possible with ForeFire to simulate any duration of fire and obtain
306 the state of the fire front at any moment between fire start and fire end.
307 Still, the simulated one-hour area alone could be a relevant information for
308 the firefighters as it provides an estimation of the potential of fire growth if
309 a fire that starts at a given location is not contained fast enough, one hour
310 being a typical time for a fire to be detected and firefighters to arrive on-site.

311 **3 Emulation with deep learning**

312 In the context of fire growth prediction mentioned in Section 2, the absence
313 of knowledge regarding the location of fire start and the uncertainty in the
314 simulation are considerable difficulties that need to be addressed. An intu-
315 itive method consists in running a large number of simulations for ignition
316 points all across the map, where some inputs are determined from weather
317 forecasts. This procedure may or may not include perturbations in the in-
318 puts other than ignition point coordinates to account for uncertainty; but in
319 any case, the time required to run all the desired simulations in operational
320 conditions is too high with usual numerical simulators such as ForeFire. This
321 motivates the use of an emulator to compute the area of the output simulated
322 burned surface in a reasonable amount of time, although with some error of
323 approximation. It is desirable to obtain an emulator that approximates this

324 quantity with high accuracy and has a significantly lower computational time
325 than that of the simulator, but it can be quite challenging for an emulator
326 to combine both properties.

327 **3.1 Design of experiments**

328 A common strategy to design an emulator consists in considering the simu-
329 lator as a “black-box” and build the emulator based on a synthetic dataset
330 of input and corresponding output. The first step of this strategy is to define
331 a design of experiments (DOE) to generate the datasets that will be used to
332 build the emulator and evaluate its approximation error. Given input dimen-
333 sion and model complexity in the present study, we expect a large number
334 of simulations ($\sim 10^5$ at the very least) will be required for an emulator to
335 have good accuracy.

336 The DOE relies on a Latin Hypersquare Sample (LHS) in $[0, 1]^{46}$, which
337 is a popular space-filling design. For all elements of the LHS, we apply an
338 affine transformation from $[0, 1]^{46}$ to the hyperrectangle whose boundaries
339 are defined by the ranges in Table 1. However, this procedure alone does
340 not account for the restrictions to the definition domain implied by the con-
341 straints on ignition point coordinates and wind speed norm. To include these
342 constraints, we generate a LHS with more members than n_{train} , the desired
343 number of training sample members, and keep only “valid” members, namely
344 those that satisfy the constraints after the affine transformation, so that the
345 resulting sample size is slightly lower than the target. The next step in the

346 constitution of the DOE is to generate a Sobol' sequence in $[0, 1]^{46}$, which is
 347 a low-discrepancy sequence. We complete the initial LHS (in $[0, 1]^{46}$) with
 348 members of the Sobol' sequence based on a discrepancy criterion, following
 349 the idea proposed in (Iooss, Boussouf, Feuillard, & Marrel, 2010) to obtain
 350 an optimal complementary design. A notable difference in the present study
 351 is that the first elements selected by the algorithm are used to complete the
 352 training sample only if they are valid (they are ignored otherwise). Then,
 353 when the target size n_{train} is reached, the next valid elements are used to form
 354 a test sample of size n_{test} . This procedure aims at selecting the points of the
 355 test sample so that they are located far from each other but also far from
 356 the points of the training sample, where the approximation error is expected
 357 to be higher.

358 Finally, based on the inputs of the training and test sample, the corre-
 359 sponding fire spread simulations are carried out as described in Section 2 and
 360 the resulting outputs complete the training and test datasets.

361 **3.2 Neural network architecture**

362 Several techniques can be considered for emulation. Simple statistical meth-
 363 ods such as linear regression based on the inputs in Table 1 would most likely
 364 lead to poor approximation because of the non-linearity of the model. Other
 365 methods such as those mentioned in Section 1 (i.e. Gaussian processes, poly-
 366 nomial chaos, high dimensional model reduction, radial basis functions) are
 367 interesting alternatives, however their computational requirements (regard-

ing time and/or memory space) can become prohibitory when there are both
a high dimension ($d = 46$) and a large sample size ($\geq 10^5$).

In this problem, the input variables presented in Table 1 can be expressed
as a vector of \mathbb{R}^{46} , including the coordinates (two scalars) of the ignition
point. While these coordinates do locate the origin of the fire, they are not
used directly to compute the ROS and simulate how the fire will spread from
there. Actually, the restriction of the simulation domain to the surface that is
burned after one hour identifies the part of the spatial fields of elevation and
fuel parameters that were used in the ROS computations. Therefore, this
information could be a better-suited emulator input than the coordinates
of the ignition point. Although the simulated burned surface is not known
beforehand, the fire will almost never spread further than 10 km in an hour,
so *a priori* it will be contained in a $20 \text{ km} \times 20 \text{ km}$ square centered around the
ignition point. If one considers the fields of elevation and of fuel parameters
 h , σ_f , and S_v restricted to this square, given their 80-m spatial resolution,
this amounts to four input fields of size 256×256 for emulation. This raises
the need for a method that is adapted to handle such high-dimensional data
as well as the remaining scalar inputs.

Neural network models appear suitable for emulation of fire spread simu-
lations, not only because they usually perform well when trained on a large
dataset, but also because they can handle several types of data. In partic-
ular, CNNs proved to be quite successful in the classification of 2D inputs
such as images (e.g. Krizhevsky et al., 2012), but also for regression (e.g. Xie,

391 Xing, Kong, Su, & Yang, 2015), which is our target. Here, the simulations
 392 are also significantly influenced by the other (scalar) inputs, notably wind
 393 speed and FMC, so a network with a hybrid architecture to process both
 394 types of inputs (2D and scalar) seems well suited to our problem. The term
 395 “hybrid” may have different meanings when it comes to neural networks. It
 396 can refer to the succession of multiple ensembles of layers, with each ensemble
 397 appearing like a given type of neural network, as in (Quang & Xie, 2016)
 398 where DNA sequences are first processed by a convolutional part then by a
 399 recurrent part. In the present study, this term is understood as the use of
 400 specific types of layers for each type of input, as proposed in (Yuan, Jiang,
 401 Li, & Huang, 2020) where image, sequential, and scalar/categorical inputs
 402 are first processed separately by the network.

403 We propose an emulator based on a DNN with a hybrid architecture. A
 404 convolutional part processes the four 2D fields of elevation and fuel parameters
 405 (prior to perturbation) h , σ_f , and S_v in a square surrounding the ignition
 406 point with a side of approximately 20 km, which corresponds to an input of
 407 shape (256, 256, 4). Another part of the network processes the vector of
 408 size 46 of scalar simulation inputs mentioned in Table 1. The “absolute”
 409 coordinates (x, y) of the ignition point are replaced by (δ_x, δ_y) , which are
 410 the coordinates of this point relatively to the center of the surrounding 2D
 411 fields. Also, both 2D and scalar inputs are scaled to $[-1, 1]$ through an affine
 412 transformation before being processed by the DNN.

413 The detailed architecture of the DNN is represented in both Figure 4

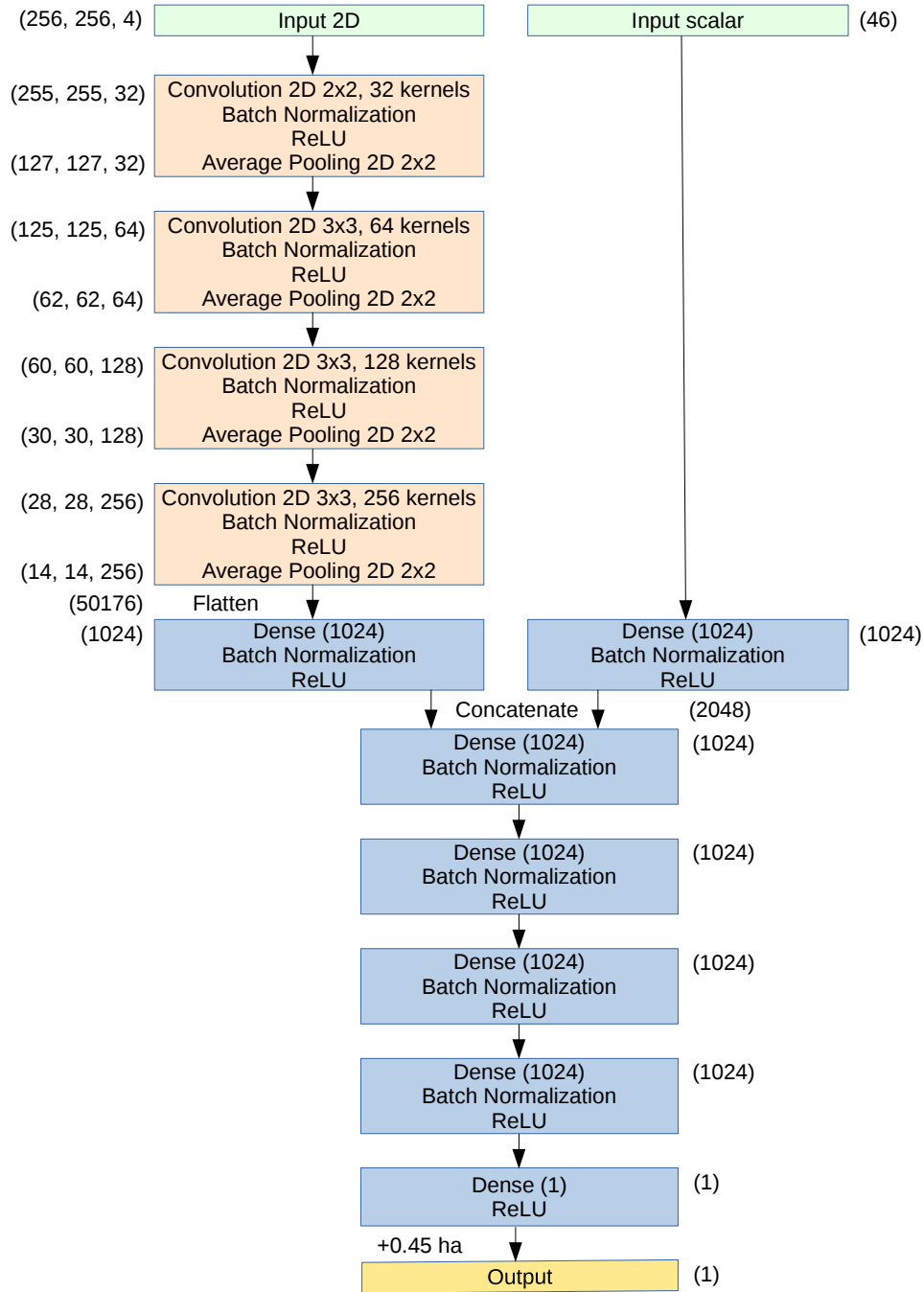


Figure 4: Neural network architecture. The numbers in brackets outside the boxes indicate the shape of the data as they are processed by the network. The architecture is considered “hybrid” because the DNN processes both 2D input corresponding to terrain data and scalar inputs. Processing is first carried out separately until concatenation after which both parts are mixed.

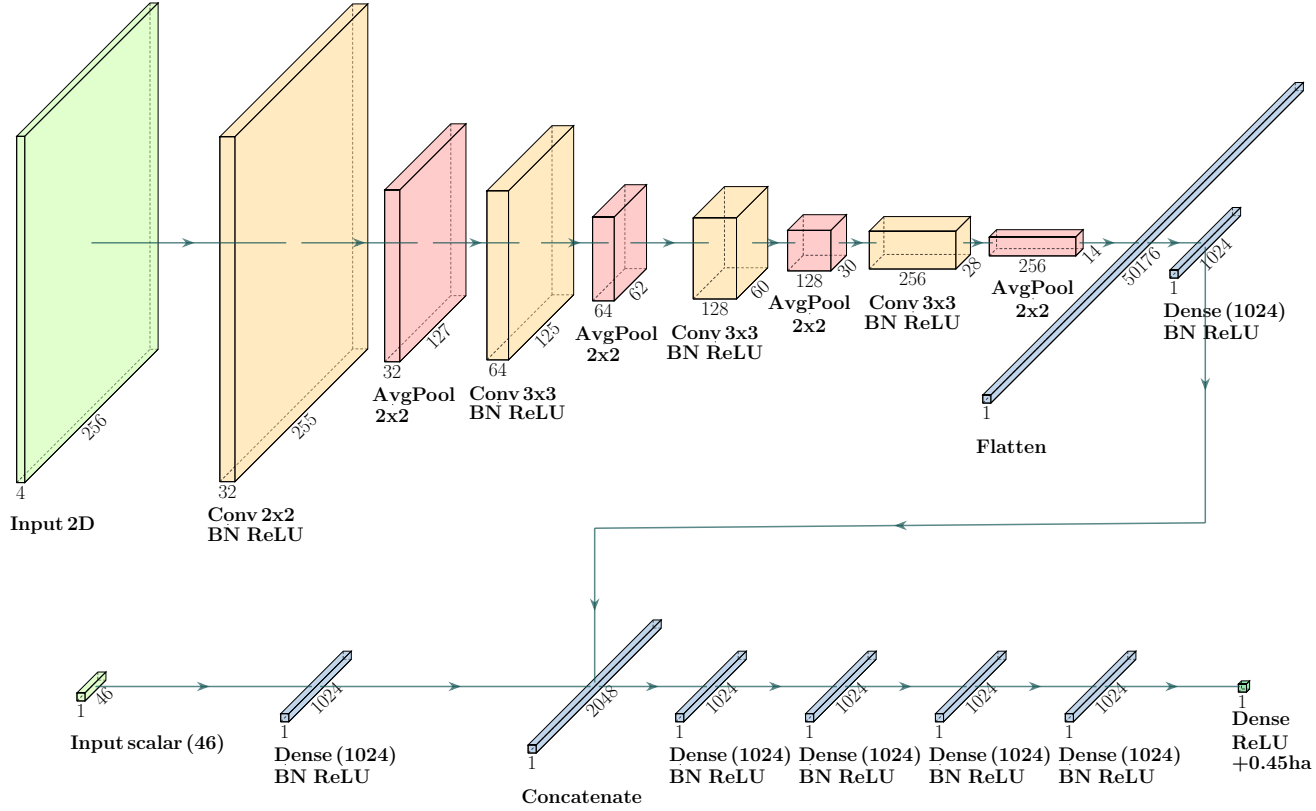


Figure 5: Representation of data processing in the neural network. The blocks indicate the shape of the data. The 2D input is derived from the four fields of elevation, and fuel parameters h , σ_f , and S_v . The 46 scalar inputs are derived from the simulation parameter inputs of Table 1. Conv: Convolution 2D; BN: Batch Normalization; AvgPool: Average Pooling 2D.

414 and Figure 5. The first figure is more focused on the processing layers (i.e.
415 convolutions, pooling, etc.), while the second figure represents the successive
416 shapes of the data as they are processed by the network.

417 First, convolutions with a 2x2 window are applied to the 2D inputs, fol-
418 lowed by a batch normalization layer, a Rectified Linear Unit (ReLU) acti-
419 vation and an average pooling layer with a 2x2 window. This succession of
420 layers is repeated three more times, with a 3x3 window for the convolutions
421 and more and more kernels. Convolutions are carried out without padding
422 nor stride, and the first two average pooling layers result in the edge of the
423 data being cropped, due to the odd input shape. Then, the output of these
424 four blocks of layers is flattened and goes through a block consisting of a fully
425 connected feed forward (aka dense) layer with 1024 output nodes, followed
426 by batch normalization and ReLU activation. As for the scalar input, it goes
427 through a similar block of layers. The output of these two blocks is con-
428 catenated and undergoes four similar blocks of layers. The intention behind
429 the application of the dense blocks before concatenation is to concatenate
430 vectors that have the same shape and potentially give similar importance to
431 the 2D part and the scalar part in this mixed architecture. Finally, a dense
432 layer followed by a ReLU activation and an increase of 0.45 ha (the minimum
433 simulated burned surface area, corresponding to a fire that does not spread)
434 are carried out, yielding the output of the network.

3.3 Accuracy metrics and training strategy

Among a dataset of size n , \mathbf{u}^i denotes the i -th set of simulation inputs, $y(\mathbf{u}^i)$ the resulting output, and $\tilde{y}(\mathbf{u}^i)$ the corresponding value returned by the emulator. Several metrics can be used to evaluate the accuracy of \tilde{y} , the emulator of function y . In this study, we use the mean absolute error (MAE), the mean absolute percentage error (MAPE), and the standardized mean square error (SMSE, cf. [Rasmussen & Williams, 2006](#)), which are respectively defined as follows:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |\tilde{y}(\mathbf{u}^i) - y(\mathbf{u}^i)|, \quad (3.1)$$

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^n \left| \frac{\tilde{y}(\mathbf{u}^i) - y(\mathbf{u}^i)}{y(\mathbf{u}^i)} \right|, \quad (3.2)$$

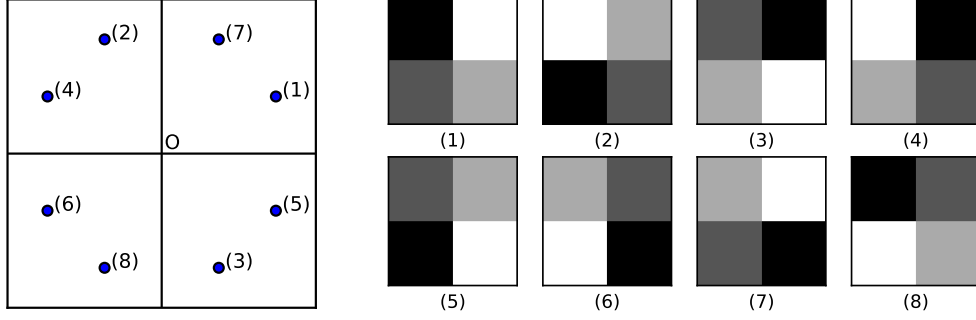
$$\text{SMSE} = \frac{\sum_{i=1}^n \left(\tilde{y}(\mathbf{u}^i) - y(\mathbf{u}^i) \right)^2}{\sum_{i=1}^n \left(y(\mathbf{u}^i) - \bar{y} \right)^2}, \quad (3.3)$$

where $\bar{y} = \frac{1}{n} \sum_{i=1}^n y(\mathbf{u}^i)$ is the sample mean of the emulated function. The SMSE can be seen as a mean squared error normalized by the sample variance of y , and would be equal to 1 if the emulator was a constant function equal to the sample mean \bar{y} . The lower these scores, the more accurate the emulator. The emulator can also be evaluated in terms of mean error, similarly to the MAE but without the absolute value, that will be referred to as “bias” in the following.

450 The accuracy metrics need to be computed for the test dataset as the
451 error is expected to be much lower for the training dataset, which is used to
452 determine the parameter values of the network. In order to quantify overfit-
453 ting, the accuracy metrics may also be computed for the training dataset.

454 The procedure used to train the network’s parameters relies on a MAE
455 loss function with an Adadelta optimizer (Zeiler, 2012), without regulariza-
456 tion based on the norm of the layer parameters.

457 To enrich the train dataset, a form of data augmentation is carried out:
458 over one epoch, each member of the training dataset is used exactly once,
459 but possibly after a geometric transformation (rotations or axial symme-
460 tries). The geometric transformation is applied to the 2D field inputs as well
461 as (W_x, W_y) , the wind speed vector, and (δ_x, δ_y) , the relative coordinates of
462 the ignition point. There is a 0.5 probability of having no transformation,
463 whereas the other transformations (seven different non-identity applications)
464 each have a 1/14 probability of being applied, all of them being represented
465 in Figure 6. We know that in such a configuration, the simulated burned sur-
466 face would be the same, so this allows us to enrich the dataset (virtually, by a
467 factor of eight) without running additional ForeFire simulations, and might
468 limit overfitting (Shorten & Khoshgoftaar, 2019) since it allows for more
469 possible configurations than described in Section 2. Note that data augmen-
470 tation is only used during training. Also, with the synthetic datasets there
471 is no need to split the training dataset to obtain a validation dataset, since
472 the test dataset was designed specifically to evaluate accuracy, as explained



(a) Transformations applied to a point. (b) Transformations applied to a matrix.

Figure 6: Geometric transformations used for data augmentation during training.

Considering (a) an initial point or vector in the plan with origin O or (b) a matrix in an initial state (1), 8 possible transformations are considered, resulting in state (n), with $n \in \{1, \dots, 8\}$. Identity transformation, leading to (1), has a 0.5 probability of being applied during training. The probability is 1/14 for all other transformations. (2): 90° rotation, (3): -90° rotation, (4): y axis symmetry, (5): x axis symmetry, (6): 180° rotation, (7): y=x axis symmetry, (8): y=-x axis symmetry.

Applying one of these transformations to the input data would result in a similarly transformed simulated burned surface, so the output area in hectares is invariant to these transformations.

473 in Section 3.1. The accuracy metrics of the network are simply computed for
474 the test dataset at the end of each epoch during training.

475 3.4 Extraction of the actual emulator

476 The DNN presented in Section 3.2 relies on many convolutions that can be
477 computed much faster with high-performance graphics cards. Even if one is
478 not equipped with such resources, it is possible to compute the output of the
479 DNN even faster once the network has been trained.

480 To achieve this goal, the final layer of the convolutional part of the
481 network (of size 1024), before concatenation with the scalar part, is pre-
482 computed. Indeed, due to the spatial resolution of the elevation and land
483 cover fields of approximately 80 m, there is a finite amount of possibilities for
484 the 2D input and the subsequent layers up to the end of the convolutional
485 part, which will take the same values as long as the ignition point is located
486 in a given cell of side ~ 80 m. In the present case, there are $\sim 1.2 \times 10^6$
487 possibilities for Corsica.

488 The actual emulator consists in the remaining part of the DNN and its
489 inputs are the pre-computed final layer of the convolutional part as well as
490 the scalar vector of size 46. This part of the network only involves some
491 dense blocks and a concatenation of the two parts of the network, that can
492 be computed much faster—even on a machine without specific acceleration.

493 3.5 Implementation

494 Python scripts are used to process the data, generate the training and test
495 datasets, build and evaluate the DNN. Keras library, which is a high-level
496 neural networks API that is running on top of TensorFlow, is used for building
497 the DNN.

498 Training and accuracy evaluation of the DNN up to the retrieval of the
499 actual emulator are carried out on a GPU accelerated compute node. The
500 computational time of the actual emulator is evaluated on a machine with
501 32 CPU.

502 The size of the datasets are $n_{\text{train}} = 5 \times 10^6$ and $n_{\text{test}} = 10^4$. Training
503 is carried out with data augmentation as explained in 3.3 for 100 epochs
504 with batches of size 400, and the hyperparameters of the Adadelta optimizer
505 are a decay rate of 0.95, a conditioning constant ϵ of 10^{-7} , and a learning
506 rate of 0.3, which is an extra factor in the right-hand term of Equation (14)
507 in (Zeiler, 2012). The weights of the network are initialized using default
508 TensorFlow arguments, therefore the weights of Dense and Conv2D layers
509 are initialized following a Glorot uniform initializer (cf. Equation (16) in
510 Glorot & Bengio, 2010).

511 The same procedure is also applied to smaller training dataset of size
512 $n_{\text{train}} \in \{10^5, 10^6\}$, each with a specific dataset of size $n_{\text{test}} = 10^4$ gener-
513 ated as explained in Section 3.1, to investigate the influence of n_{train} on the
514 approximation error.

515 4 Results and discussion

516 The computational time of a simulation (with ForeFire) of wildland fire
517 spread took an average of approximately 25 s. This time highly depends
518 on the input of the simulation and can range from about 0.1 second to more
519 than an hour. Overall, the larger the simulated burned surface, the more
520 computations are carried out during the simulation. Running all the simula-
521 tions in the training and test datasets would have taken about 4 years if it
522 were not for distributed computing: with several multi-CPU machines, for
523 a total of about 150 CPU cores, the computations were completed in about
524 10 days. Given the simulation settings presented in Section 2, the obtained
525 burned surface areas range from 0.45 ha to 24 804.4 ha among the training
526 dataset. Some statistics of this output in the training dataset are presented
527 in Table 2. The high variance of the simulation output is consistent with that
528 of computational time. The minimum output corresponds to the area of the
529 initial burned surface and is obtained in a few simulations (approximately
530 half a thousandth) where the FMC is very close to the moisture of extinction
531 (0.3) in the ROS model, leading to a fire that almost does not spread. Similar
532 statistics are obtained with the test dataset, except for the maximum output
533 (14 403.7 ha). The test dataset, having a much lower size than that of the
534 training dataset, is less representative of the tail of the output distribution,
535 hence the lower maximum.

536 Most simulations result in a burned surface of less than 1000 ha, which is

Mean	Std	Minimum	Q1	Median	Q3	Maximum
455.7 ha	782.0 ha	0.45 ha	52.6 ha	181.0 ha	517.7 ha	24 804.4 ha

Table 2: Statistics of the output simulated burned surface area among the training dataset of size 5×10^6 .

Std: Standard deviation; Q1: first quartile; Q3: third quartile.

The output has high variance, arguably making “relative” error metrics such as the MAPE and SMSE (cf. Equations (3.2) and (3.3), respectively) better suited for expressing the performance of the emulator regarding approximation error.

537 realistic for a fire that spreads freely during one hour. Still, a non-negligible
538 amount of simulations result in burned surfaces that are most certainly bigger
539 than what would be observed in reality. This amount would probably be
540 higher were it not for non-burnable zones that significantly contribute to limit
541 fire spread in some cases. This is mostly due to the fact that the simulations
542 rely on simplifying assumptions where wind speed and FMC are constant
543 in time and the DOE allows these inputs to vary in very large intervals.
544 Therefore, it is not surprising to obtain a very large burned surface in a
545 simulation where the wind speed is extremely high, the FMC extremely low,
546 and no unburnable zone is reached during a whole hour of spread. Although
547 somewhat unrealistic, the extremely high values of simulated burned surfaces
548 were not removed from the dataset. This might make the emulation more
549 difficult but the ability to discriminate between a wide range of situations,

550 even extreme ones, is relevant in wildland fire spread.

551 Carrying out one epoch took a bit less than three hours, so training, which
552 consisted in 100 epochs, lasted for about 10 days. The evolution of the MAE
553 over training of the DNN for these 100 epochs is reported in Figure 7. At a
554 given epoch, the predicted values for both test and training datasets result
555 from the model obtained at the epoch’s end. Due to high computational time,
556 the MAE was only computed for the training dataset (without applying data
557 augmentation) at the first epoch and every five epoch starting from the fifth.
558 On the one hand, the MAE for the test dataset decreases overall until it
559 reaches 81.5 ha after about 78 epochs, after which it oscillates around that
560 value. On the other hand, the MAE for the training dataset decreases overall,
561 faster than the MAE of the test dataset. Therefore, while both scores are
562 almost identical at the start, the gap between the two increases with the
563 number of epochs.

564 The main objective is to have low generalization error, which is measured
565 here using the error metrics for the test dataset. In high-dimensional cases,
566 it is possible to observe a significant gap in error between the training and
567 test datasets when training neural networks (see for instance Advani, Saxe,
568 & Sompolinsky, 2020). It is the case in this study with the original model
569 (ForeFire) relying on high-dimensional input data and being highly non-
570 linear. Note that the neural network can be interpreted here as a substitute
571 for an interpolator in high dimension, without the constraint to coincide with
572 the training dataset at the training points.

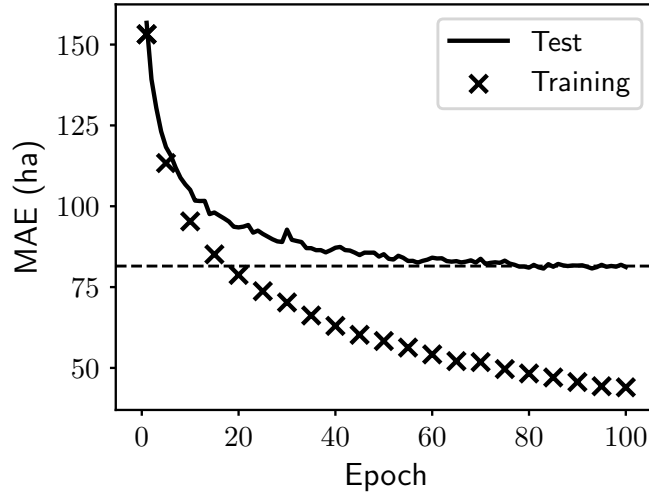


Figure 7: MAE (loss function) over training. The solid curve represents the MAE for the test dataset, while the crosses represent the MAE computed for the training dataset at the end of the first epoch and after every five epochs starting from the fifth. The horizontal dotted line corresponds to MAE=81.5 ha.

Both metrics decrease and this decrease is faster for the training dataset, yet the MAE for the test dataset does not increase and seems to keep around 81.5 ha after about 75 epochs. No significant decrease in the test error is expected after more epochs, so the network after 94 epochs, which has a SMSE on the test dataset of 6.0% (the best over all 100 epochs) is selected for emulation.

573 It is unlikely that carrying out more training epochs would result in a
574 significant decrease of the error metrics for the test dataset. Consequently,

575 the model with the best SMSE over the test set, which was obtained at the
 576 end of the 94-th epoch, was selected to define the emulator. The emulator
 577 with the best MAE was not selected because its MAE was only slightly lower
 578 (80.7 ha instead of 81.2 ha), whereas the other scores were all better for the
 579 model with the best SMSE. Even though our loss function (the MAE) may
 580 seem high, an absolute error of about 80 ha is at the same time very high
 581 for a small simulated burned surface of about 10 ha and very small for larger
 582 ones of about 1000 ha. Consequently, “relative” error metrics such as the
 583 MAPE and SMSE (cf. Equations (3.2) and (3.3), respectively) are arguably
 584 better suited for expressing the performance of the emulator regarding ap-
 585 proximation error.

Model \ Metric	MAE	MAPE	SMSE	Bias
Mean of training	461.9 ha	2266%	100.0%	2.2 ha
Linear regression without threshold	387.9 ha	1239%	73.9%	−4.8 ha
Linear regression with 0.45 ha threshold	361.1 ha	493.3%	72.3%	21.9 ha
DNN after 100 epochs	81.2 ha	33.5%	6.2%	−13.1 ha
Emulator (from DNN after 94 epochs)	81.2 ha	32.8%	6.0%	−6.5 ha

Table 3: Model error on test dataset of size 10^4 .

The approximation is poor using the three most simple models (constant and
 linear regression with or without threshold), whereas the DNN trained using
 a large training dataset shows good approximation.

586 The error metrics of the emulator are reported in Table 3 and Table 4, re-

Model \ Metric	MAE	MAPE	SMSE	Bias
Mean of training	461.5 ha	2139%	100.0%	0 ha
Linear regression without threshold	389.9 ha	1185%	73.7%	0 ha
Linear regression with 0.45 ha threshold	365.6 ha	493.4%	72.3%	24.3 ha
DNN after 100 epochs	44.0 ha	23.8%	1.2%	−7.6 ha
Emulator (from DNN after 94 epochs)	45.1 ha	23.2%	1.2%	−0.9 ha

Table 4: Model error on training dataset of size 5×10^6 .

For the three most simple models, the approximation metrics are almost the same to the ones computed on the test dataset (cf. Table 3), whereas the DNN has lower error for the training dataset than for the test dataset.

spectively relating to the test dataset and the training dataset. These metrics are also computed for three simple models for comparison: 1) a model that consists in always predicting the mean simulated burned surface of the training dataset (455.7 ha), 2) a linear regression model fitted using the training dataset based on the 46 inputs of Table 1, 3) same as the previous model, but applying a 0.45 ha minimum threshold (the minimum simulated area) to avoid non-physical output. The metrics for the DNN with the parameters obtained at the end of training are also reported. Although a MAE of 81.2 ha might seem high, it is much lower compared to that of the three simple models (461.9 ha with the mean, 361.1 ha for the linear regression with threshold). The SMSE of 6.0% means that 94.0% of the variance in the test dataset output is explained by the emulator, which is very good given the

599 range of variation in simulation inputs. The relative error is also satisfactory
 600 with a MAPE of 32.8% on the test dataset, especially when compared to
 601 that of the simple models (2266.0% using the mean, 493.3% using linear re-
 602 gression with threshold). As for computational time on a 32-CPU machine,
 603 the outputs for the test dataset are obtained in about half a second with the
 604 emulator against 56 s with the whole DNN, which corresponds to a speed-
 605 up by a factor of about 100. Also, the corresponding ForeFire simulations
 606 would have been obtained in about two hours with parallel computations
 607 on the 32-CPU machine, meaning that the emulator allows a speed-up by
 608 about 15,000 times. For a dataset where the simulated burned surface tends
 609 to be higher, the average computational time with ForeFire could be higher.
 610 This is not the case for the emulator, for which computational time does not
 611 depend on the output fire size, meaning that the resulting speed-up factor
 612 would be higher.

$n_{\text{train}} \setminus \text{Metric}$	MAE	MAPE	SMSE	Bias
10^5 (best SMSE after 34 epochs)	182.0 ha	89.1%	25.4%	−22.6 ha
10^6 (best SMSE after 26 epochs)	127.5 ha	49.9%	13.3%	−16.3 ha
5×10^6 (best SMSE after 94 epochs)	81.2 ha	32.8%	6.0%	−6.5 ha

Table 5: DNN error on complementary test dataset (always of size $n_{\text{test}} = 10^4$) with variable training dataset size n_{train} .

The larger the training dataset of the DNN, the better the approximation.

613 The influence of n_{train} on the resulting approximation error of the DNN

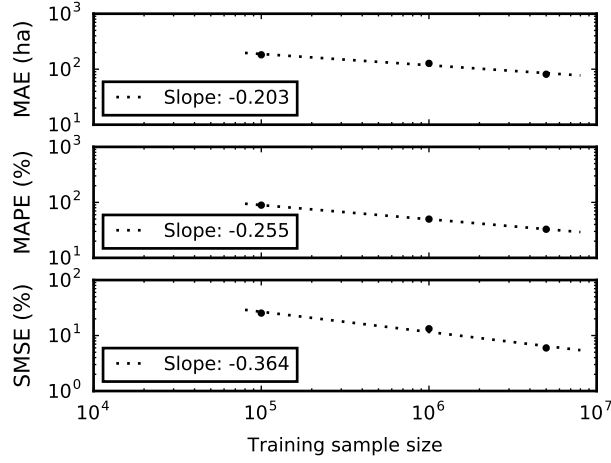


Figure 8: Plot of the error metrics against n_{train} the size of the training dataset (cf. values reported in Table 5), in log scale.

Linear regression of the logarithms (dotted lines) results in a good fit with the data, suggesting a slowly decreasing trend of the form $(n_{\text{train}})^{-\alpha}$ with $\alpha \in [0.2, 0.37]$, depending on the metric.

614 based on the error metrics for the test dataset is presented in Table 5. The
615 MAE in the test dataset did not seem to decrease after a few tens of epochs
616 for smaller training datasets (this was also the case for $n_{\text{train}} = 5 \times 10^6$),
617 and the model with best SMSE over 100 epochs, which had a MAE close to
618 the best value obtained over the 100 epochs, was selected. Figure 8 shows
619 the MAE, MAPE, and SMSE from Table 5 plotted against n_{train} . Although
620 there are only three points for each metric, linear regression of the logarithms
621 suggests that the metrics decrease following a trend of the form $(n_{\text{train}})^{-\alpha}$ with
622 $\alpha \in [0.2, 0.37]$, depending on the metric, which is quite slow. For instance,

623 assuming this trend using the values of the slopes reported in Figure 8 to
 624 specify α , reducing the MAE from 81.2 ha to 50 ha (resp. the MAPE from
 625 32.8% to 20% and the SMSE from 6.0% to 2.0%) would require to increase
 626 n_{train} by approximately a factor 10 (resp. 7 and 20).

627 For more insight regarding the approximation of the selected model, the
 628 emulator output for each member of the test dataset is plotted against the
 629 actual values of simulated burned area in Figure 9. The vast majority of the
 630 emulated values are close to their simulated counterparts and 9,332 out of
 631 10,000 are at most either twice higher or half lower. In 157 cases, the emulator
 632 returns the minimum value of 0.45 ha, while the actual simulated value may
 633 go up to 10 ha. This corresponds to the apparent “black vertical bar” at the
 634 lower left of the graph in Figure 9a. There are 29 simulations for which the
 635 emulated burned area is at least five times lower (11 of them being equal to
 636 0.45 ha) and 43 simulations for which the emulated value is at least five times
 637 higher. In the latter cases, most of the simulated burned surfaces are small
 638 (≤ 10 ha in 32 simulations out of 43), which usually contributes to a higher
 639 relative error, but not all of them. In some of these cases of overprediction
 640 by the emulator, there is a relatively small area close to the ignition point
 641 in the main direction of fire spread that seems to considerably slow down
 642 the fire. The emulator probably has difficulty when it comes to accounting
 643 for some particular configurations of the underlying fuel and altitude fields,
 644 especially small non-burnable areas, given that the convolutional part of the
 645 DNN reduces the size of inputs by a factor of 256 when processing it for the

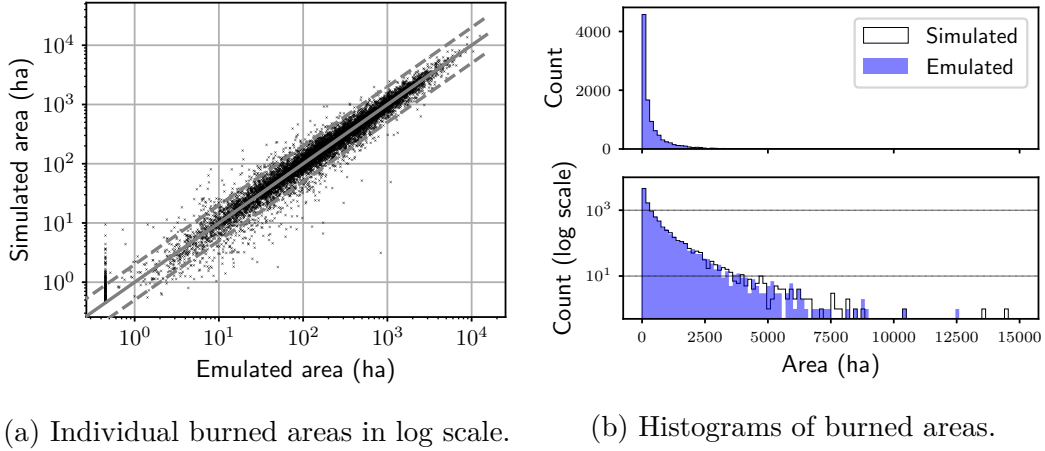


Figure 9: Comparison between the burned area simulated by ForeFire and the corresponding emulator output over the test dataset of size 10^4 .

(a) The solid oblique gray line corresponds to a perfect match and the dotted lines correspond to an error by a factor of 0.5 and 2.

(b) Black contour: histogram of simulated areas; blue surface: histogram of emulated areas. Both top and bottom figures represent the same distributions, they share the same abscissa axis but the bottom figure has its ordinate in log scale.

Most of the emulated values are at most either twice higher or half lower, resulting in good error metrics (MAE=81.2 ha, MAPE=32.8%), yet the individual error is quite high for a few members. The distributions of both samples are close.

emulator (from 262,144 to 1024). Overall, the individual errors lead to similar distributions of burned area. The emulator has a small bias of -6.5 ha and,

as shown in Figure 9b, the histogram of emulated burned areas is slightly
less dispersed (standard deviation of 752.9 ha against 782.5 ha).

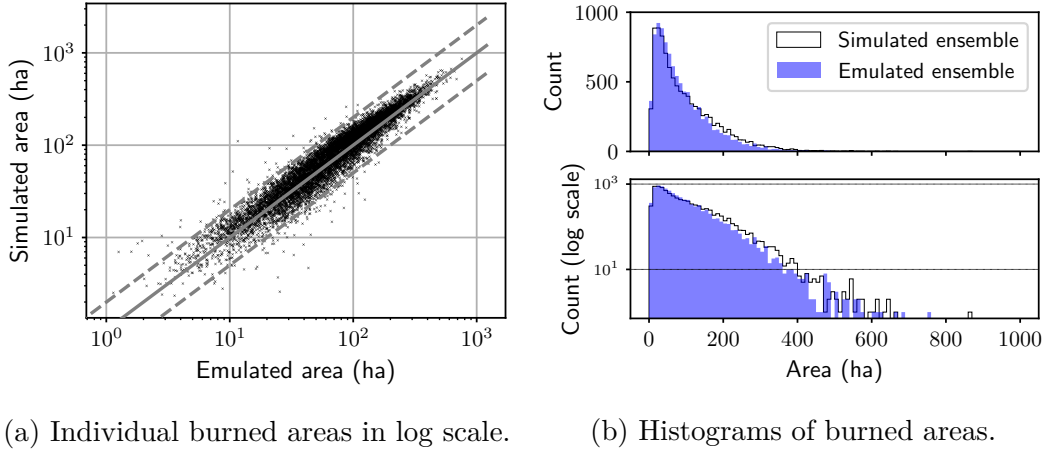


Figure 10: Comparison between the ensemble of burned areas simulated by ForeFire for the fire case of Calenzana and their emulated counterparts.

(a) The solid gray line corresponds to a perfect match and the dotted lines correspond to an error by a factor of 0.5 and 2.

(b) Black contour: histogram of simulated areas; blue surface: histogram of emulated areas. Both top and bottom figures represent the same distributions, they share the same abscissa axis but the bottom figure has its ordinate in log scale.

The inputs have smaller variations than for the test dataset, yet most emulated values fall into the range of half to twice the simulated value as it was the case for the test dataset (cf. Figure 9), leading to good error metrics (MAE=18.7 ha, MAPE=22.7%).

650 The emulator is also evaluated with an ensemble of ForeFire simulations
 651 that correspond to a real Corsican fire that occurred near Calenzana during
 652 summer 2017 and burned about 120 ha. Most of the spread for this fire took
 653 place during the first hour after ignition. For this case, some reference inputs
 654 are defined from weather predictions and a presumed ignition point is iden-
 655 tified, as explained in (Allaire et al., 2020). Then, an ensemble of perturbed
 656 simulations is generated, for which the inputs presented in Table 1 follow a
 657 calibrated distribution that was obtained in a previous study (Allaire et al.,
 658 2021) with $\beta = 1/2$. It should be noted that the resulting ensemble of burned
 659 surface areas in the present study is not the same as in (Allaire et al., 2021)
 660 because supplementary inputs were variable in the previous study (such as
 661 perturbations in the times of fire start and fire end, which could make the sim-
 662 ulated fire duration different from one hour). The 10,000 simulated burned
 663 surface areas of the ensemble are compared to their emulated counterparts
 664 in Figure 10. Similarly to the test dataset, most emulated values fall into the
 665 range of half to twice the simulated value, leading to a MAPE of 22.7%. A
 666 MAE of 18.7 ha is obtained and individual errors result in a distribution of
 667 the emulator output that is less dispersed than that of the simulated output,
 668 as shown in Figure 10b, with a bias of -9.6 ha and a standard deviation of
 669 77.7 ha against 86.1 ha.

670 The overall agreement between simulation and emulation is good for this
 671 simulated fire case and the simulations were computed in 20 minutes, whereas
 672 the emulator predictions only took a bit more than a second. The speed-up

673 factor is about 1000 this time, which is lower than the several thousands
674 obtained for the test dataset. This is explained by the lower simulation time
675 for this fire case (20 min instead of about two hours for the test dataset, while
676 both datasets have the same size). This performance is quite promising for
677 application to ensemble forecasting, but care should be taken as propagation
678 of uncertainty leads to different output distributions according to the model
679 (either ForeFire or its emulator) used.

680 Linked to the approximation error of the emulator is the influence of the
681 inputs on the output. A desirable property of the emulator is the ability to
682 behave in a similar way as ForeFire so that it keeps the main characteristics
683 of the fire spread model, namely a burned area that, overall, increases with
684 wind speed and decreases with FMC, while the surrounding 2D fields of
685 altitude and fuel can either favor or block fire spread. Perturbations of
686 fuel parameters are expected to have less influence, especially those of fuel
687 parameters that are applied to a specific fuel type (h, σ_f, S_v) . Also, the
688 ROS is proportional to heat of combustion ΔH , which is a global parameter,
689 so positive perturbations of this quantity will increase the burned area and
690 negative ones will decrease it.

691 Given the complexity of the emulator, one may approach it as a black-box
692 and estimate the overall influence of its inputs with Shapley additive expla-
693 nations (SHAP, cf. [Lundberg & Lee, 2017](#)), a feature attribution method.
694 The features we focus on are the inputs of the emulator, namely the 1024
695 “position” scalars linked to the 2D fields surrounding the ignition point stem-

696 ming from the convolutions and the remaining 46 scalar inputs. Approximate
 697 SHAP values are computed for each member of the test dataset by means
 698 of expected gradients. This procedure leads to exact SHAP values when the
 699 model to explain is linear and the features are independent. While these
 700 assumptions are not verified with the emulator, the original algorithm for
 701 the computation of the exact SHAP values is too computationally expensive,
 702 whereas this method allows to compute approximate values in a reasonable
 703 amount of time. Although these results should be taken with care, they can
 704 still be used for a qualitative analysis and should provide some insight on
 705 the overall input influence over a whole dataset. For each member of the test
 706 dataset, the expected gradient is estimated based on a subset of size 50,000
 707 sampled randomly from the training dataset. Given that the 1024 position
 708 scalars are difficult to interpret and expected to have little individual influ-
 709 ence on the output due to their correlation, we consider the sum of their
 710 SHAP values, which is identified via a fictitious variable named “Position”.
 711 The approximate SHAP values obtained for 12 of the 47 resulting variables
 712 are summarized in Figure 11.

713 The values obtained for each of the 10,000 test members represented in
 714 Figure 11b indicate a good overall agreement with the main characteristics of
 715 the fire behavior model. High FMC (m_c) tends to decrease the output while
 716 low FMC tends to increase it. High positive SHAP values for the coordinates
 717 of wind speed (W_x and W_y) are obtained for extreme values of these inputs,
 718 i.e. close to either -35 m s^{-1} or 35 m s^{-1} (in blue and red, respectively) while

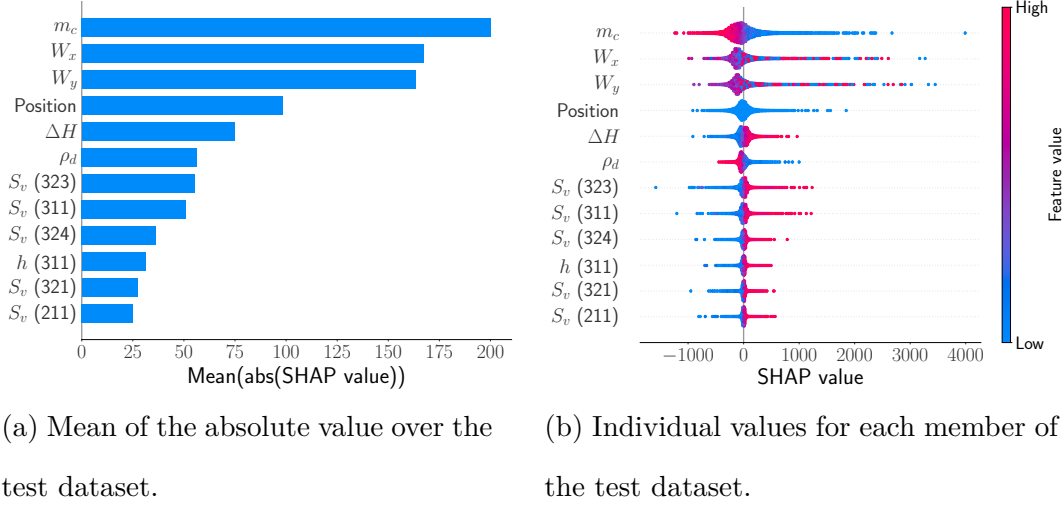


Figure 11: Approximate SHAP values associated with the emulator computed for the test dataset, using the training dataset as basis. The SHAP values corresponding to the 1024 inputs resulting from the convolutional part of the DNN are summed up and this sum is identified as “Position” in the figure. Only the 12 most overall influential inputs, as ranked in (a), are represented.

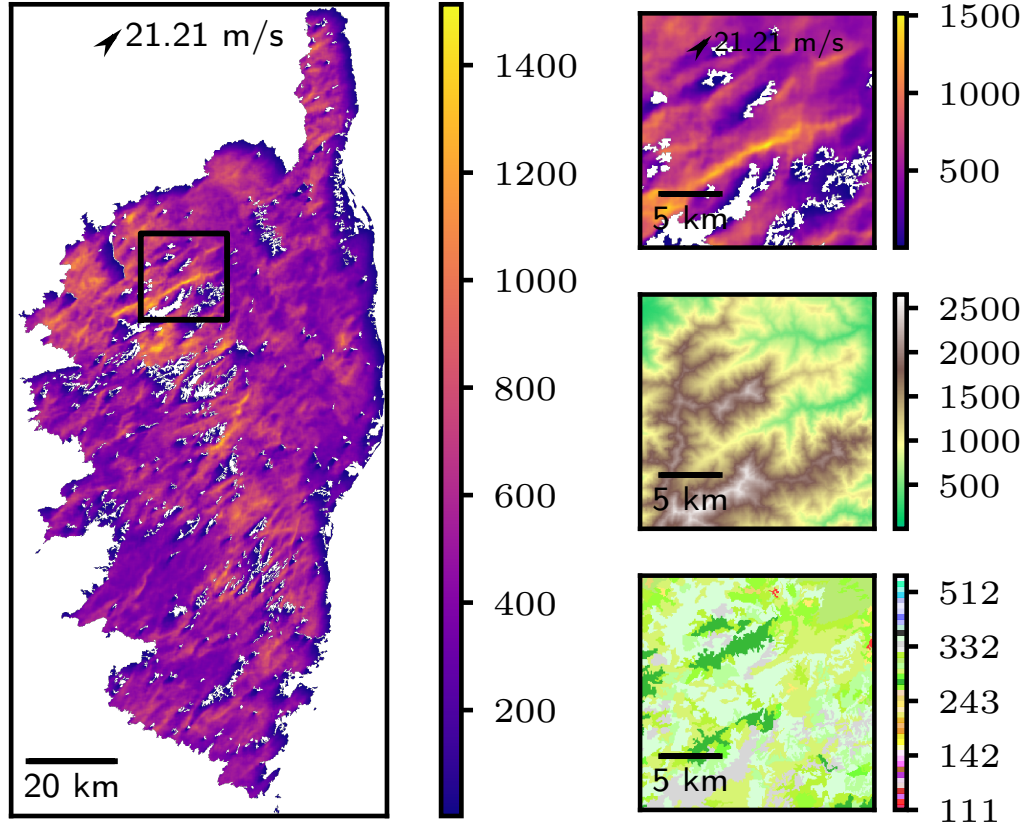
(b) The color indicates the value of the input for each member, while the SHAP value is read in abscissa.

Qualitatively, the influence of the inputs expressed by the SHAP values corresponds to typical behavior of fire spread, both in terms of ranking and in terms of values for individual members (e.g. overall, low FMC m_c leads to a high SHAP value and high FMC leads to a low SHAP value). This suggests that the input-output relationship of the emulator is similar to that of the simulator.

719 the negative values are obtained for intermediate values (close to 0 m s^{-1}).
 720 SHAP values associated to the perturbation of ΔH are also consistent with
 721 our expectations. Regarding the rankings of the inputs when looking at the
 722 absolute SHAP values averaged over the test dataset in Figure 11a, the three
 723 most influential inputs are the FMC and the two coordinates of the wind
 724 speed vector. Position is ranked fourth, perturbations on fuel parameters
 725 that affect all fuel types (ΔH and ρ_d) are ranked fifth and sixth, and the
 726 remaining ranks are attributed to the other perturbations of fuel parameters
 727 as well as δ_x and δ_y (ranked last). Interestingly, when the positional inputs
 728 are not summed, their individual influence is quite low: the 54th scalar of
 729 the vector of size 1024 is the highest ranked at rank 32 only. Although we
 730 only have an approximation of SHAP values, these results are qualitatively
 731 the ones we would expect from fire spread simulations and indicate that the
 732 emulator has an overall relationship between inputs and output that is fairly
 733 consistent with typical behavior of wildland fire spread.

734 The “physical” behavior of the emulator is also analyzed through the lens
 735 of fire danger mapping in Figure 12 that represents the response surface of the
 736 emulator where the ignition point varies in Corsica, whereas the other inputs
 737 are fixed to $m_c = 0.13$, $(W_x, W_y) = (15, 15) \text{ m s}^{-1}$, and no perturbation on
 738 fuel parameters.

739 This mapping involves $\sim 1.2 \times 10^6$ emulator computations, which are
 740 carried out in about 40 s only. Values lower than 200 ha can be observed
 741 toward the south-west of non-burnable areas (mostly water bodies, rocky



(a) Map of emulated burned area on the entire Corsica island. (b) Zoom in the black square represented in (a).

Figure 12: Map of the area (in hectares) of the burned surface predicted by the emulator with variable ignition point in Corsica. The other inputs are a wind speed vector of $(15, 15) \text{ m s}^{-1}$ represented with a black arrow, a FMC of 0.13, and no perturbation on fuel parameters. The spatial resolution is approximately 80 m; white pixels correspond to non-burnable locations in the simulations.

(b) From top to bottom: burned area (ha), altitude (m), land cover.

The “potential” area at a given ignition point is clearly lower when unburnable locations are close to the ignition point in the direction of the wind speed vector. This is consistent with “physical” behavior of wildland fire spread.

742 mountain tops over 1800 m with no vegetation, and urban areas), while most
743 of the other ignition points are associated to values higher than 300 ha. This
744 is consistent with the input wind speed vector pointing to the north-east.
745 Also, there is a fairly high spatial variation of the emulated burned area that
746 goes up to about 1500 ha. The smaller region shown in Figure 12b presents
747 some of the highest emulated values. Comparison with the underlying 2D
748 fields of altitude and fuel used in the simulations does not reveal clear in-
749 fluence of either one of these fields on the emulated output (except for the
750 ignition points to the south-west of non-burnable locations). An animated
751 version of Figure 12a with varying wind is available as Supplementary mate-
752 rial. Considering that the approximation errors of the emulator are relatively
753 low, it appears that, overall, the map generated using the emulator highlights
754 locations where ignition would induce larger burned areas.

755 5 Conclusions

756 The basis for the present study was simulations of wildland fire spread with
757 the numerical solver ForeFire using the underlying ROS model of Rothermel.
758 These simulations represented free fire spread during one hour from a small
759 initial burned surface located at all possible areas in Corsica island. The
760 terrain was represented by 2D fields of fuel and altitude at approximately
761 80-m resolution in the simulations. Some environmental input parameters,
762 namely FMC, wind speed, and perturbation of fuel parameters, were also

763 allowed to vary in a wide range. ForeFire simulations can be computed in a
764 reasonable amount of time, yet too high for applications that require a large
765 number of simulations on a daily basis. This motivated the use of an emulator
766 in order to faster compute an approximation of the output simulated burned
767 area (in hectares).

768 The proposed approach consisted in training a DNN used for regression.
769 The network has a hybrid architecture to deal with 2D fields of environmental
770 parameters and with scalar inputs. On the one hand, the 2D fields are
771 restricted to a square of 20 km side centered around the ignition point to filter
772 out information that is, for the most part, not used during the simulation,
773 and these fields go through convolutional blocks due to their similarity to
774 images. On the other hand, the remaining scalar inputs go through a dense
775 block and are concatenated with last layer of the convolutional part. Then,
776 the rest of the network consists in more dense blocks. Training was carried
777 out with a large dataset of size 5×10^6 obtained from a LHS sample, which
778 could be augmented during training, and a complementary test sample of
779 size 10^4 was obtained from a low-discrepancy sequence.

780 The DNN achieved good approximation of burned surface area simulated
781 by ForeFire. The last layer of the convolutional part of the DNN for all
782 fuel cells ($\sim 1.2 \times 10^6$) of the map of Corsica for which ignition is possible
783 in the simulation is pre-computed. This allows to reduce computational
784 time since the resulting positional information can be used together with
785 the scalar inputs to run computations with only the remaining part of the

786 DNN, which was chosen as emulator of burned surface area. The emulator
787 showed satisfactory performance. In the test dataset, it explains 94.0% of
788 the variance of the output, it has a MAPE of 32.8%. Also, compared to the
789 ForeFire simulations for fire danger mapping, the emulator computations
790 are carried out thousands of times faster on a 32-CPU machine. Finally,
791 the overall influence of the inputs on emulator output seems consistent with
792 typical behavior of wildland fire spread.

793 Preliminary results suggest that the emulator is suited to ensemble pre-
794 dictions and fire danger mapping, notably due to the considerable speed-up
795 factor. For instance, 1.2 million ForeFire simulations requiring 25 s on aver-
796 age would be computed in more than 10 days on a 32-CPU machine, while
797 this took about 40 s with the emulator, that is to say more than 20,000 times
798 faster.

799 Even though the DNN was trained for Corsica using ForeFire, the method
800 presented in this work could be applied to other regions and/or similar fire
801 spread simulators. In this method, the two most computationally expensive
802 steps are 1) running the simulations required for the training dataset and 2)
803 training the DNN. Thanks to high-performance computing resources (multi-
804 CPU machines for the simulations and a GPU-accelerated core for training),
805 both steps only took about 10 days in the present study. In other cases with
806 bigger territories (e.g. at the scale of a country), one can expect that an
807 even larger training dataset will be required to obtain comparable approx-
808 imation error. Also, if the spatial resolution of the data maps is different,

one may consider adapting the convolutional part. Regarding data size, the available RAM also poses a constraint on the batch size used during training. For these reasons, the authors strongly recommend using high-performance computing resource to apply the method and starting with a relatively small training dataset before moving on to a larger dataset. In spite of the high computational time for those two steps, once the DNN is trained and the emulator is obtained, the resulting speed-up factor should be worthwhile.

A major research perspective consists in evaluating the emulator for use in ensemble predictions and fire danger mapping, but now in a more extensive manner. In particular, actual weather forecasts that cover the whole island will be used to generate fire danger maps for every hour (at least) of a given day. This process can be carried by considering several real fire cases or an entire fire season. Depending on the ability of the emulator to quickly identify the locations with higher fire danger ahead of time, it could provide valuable help in an operational context.

Another perspective is to investigate how the DNN compares with other approximation techniques (regression or interpolation), but their application can be quite computationally expensive with large training datasets and may require to carry out data reduction on high-dimensional inputs. One may also focus on the neural network architecture to either increase its performance or extend its application to more scenarios of wildland fire spread simulations. A first extension could be to consider more simulation outputs, for instance the burned surface area every ten minutes after ignition. In this case, the DNN

could yield a vector output that represents burned areas at different forecast times (instead of a single scalar) where each component could be expressed as the sum of the previous component plus a positive quantity. Similarly, inputs such as wind speed vector and FMC could vary during the simulation time. This would entail more possibilities in simulated scenarios, making the emulator more relevant for simulations of fires spreading during 1 hour or more, provided that it is trained with realistic weather time series, the definition of which is not obvious. As for network architecture, upsampling layers could be considered hoping that they would re-constitute a good raster approximation of the burned surface. This burned surface could either be used directly as output (as in [Hodges & Lattimer, 2019](#)) or as the layer previous to the final output node estimating the number of hectares burned. Also, multi-dimensional recurrent neural networks ([Graves, Fernández, & Schmidhuber, 2007](#)) could be considered as substitute for the convolutional part of the DNN. Regardless of the complexity of the emulator, the main properties to pursue remain the same: low approximation error and reduction in computational time.

Acknowledgments

Funding: This work was supported by the Agence Nationale de la Recherche, France [grant number ANR-16-CE04-0006 FIRECASTER].

853 This work was carried out using HPC resources from GENCI-IDRIS (Grant
854 2020-AD011011630).

References

- Advani, M. S., Saxe, A. M., & Sompolinsky, H. (2020). High-dimensional dynamics of generalization error in neural networks. *Neural Networks*, *132*, 428–446. 10.1016/j.neunet.2020.08.022
- Allaire, F., Filippi, J.-B., & Mallet, V. (2020). Generation and evaluation of an ensemble of wildland fire simulations. *International Journal of Wildland Fire*, *29*(2), 160–173. <https://doi.org/10.1071/wf19073>
- Allaire, F., Mallet, V., & Filippi, J.-B. (2021). Novel method for a posteriori uncertainty quantification in wildland fire spread simulation. *Applied Mathematical Modelling*, *90*, 527–546. <https://doi.org/10.1016/j.apm.2020.08.040>
- Chen, N., Liu, W., Bai, R., & Chen, A. (2017). Application of computational intelligence technologies in emergency management: a literature review. *Artificial Intelligence Review*, *52*(3), 2131–2168. 10.1007/s10462-017-9589-8
- Cruz, M. G., & Alexander, M. E. (2014). Uncertainty in model predictions of wildland fire rate of spread. In *Advances in forest fire research* (pp. 466–477). Imprensa da Universidade de Coimbra. 10.14195/978-989-26-0884-6_54
- Duff, T. J., Cawson, J. G., Cirulis, B., Nyman, P., Sheridan, G. J., & Tolhurst, K. G. (2018). Conditional performance evaluation: Using wild-fire observations for systematic fire simulator development. *Forests*,

877 9(4). Retrieved from <https://www.mdpi.com/1999-4907/9/4/189>
878 10.3390/f9040189

879 Duff, T. J., Chong, D. M., Taylor, P., & Tolhurst, K. G. (2012). Procrustes
880 based metrics for spatial validation and calibration of two-dimensional
881 perimeter spread models: A case study considering fire. *Agricultural*
882 *and Forest Meteorology*, 160, 110 - 117. 10.1016/j.agrformet.2012.03
883 .002

884 Duff, T. J., Chong, D. M., & Tolhurst, K. G. (2016). Indices for the evaluation
885 of wildfire spread simulations using contemporaneous predictions and
886 observations of burnt area. *Environmental Modelling & Software*, 83,
887 276 - 285. 10.1016/j.envsoft.2016.05.005

888 Feranec, J., Soukup, T., Hazeu, G., & Jaffrain, G. (2016). *European landscape*
889 *dynamics: Corine land cover data*. Boca Raton, USA: CRC Press.

890 Filippi, J.-B., Mallet, V., & Nader, B. (2013). Representation and evaluation
891 of wildfire propagation simulations. *International Journal of Wildland*
892 *Fire*, 23, 46–57. 10.1071/WF12202

893 Filippi, J.-B., Mallet, V., & Nader, B. (2014). Evaluation of forest fire mod-
894 els on a large observation database. *Natural Hazards and Earth Sys-*
895 *tem Sciences Discussions*, 2(11), 3077–3091. Retrieved from [https://](https://www.nat-hazards-earth-syst-sci.net/14/3077/2014/)
896 www.nat-hazards-earth-syst-sci.net/14/3077/2014/ 10.5194/
897 nhessd-2-3219-2014

898 Filippi, J.-B., Morandini, F., Balbi, J. H., & Hill, D. R. (2010). Discrete event
899 front-tracking simulation of a physical fire-spread model. *SIMULA-*

900 *TION*, 86(10), 629–646. <https://doi.org/10.1177/0037549709343117>

901 Finney, M. A. (1998). *Farsite: Fire area simulator-model development and*
 902 *evaluation*. Res. Pap. RMRS-RP-4, Revised 2004, Ogden, UT: U.S.
 903 Department of Agriculture, Forest Service, Rocky Mountain Research
 904 Station. 47 p.

905 Finney, M. A., Grenfell, I. C., McHugh, C. W., Seli, R. C., Trethewey, D.,
 906 Stratton, R. D., & Brittain, S. (2011, Apr 01). A method for ensem-
 907 ble wildland fire simulation. *Environmental Modeling & Assessment*,
 908 16(2), 153–167. 10.1007/s10666-010-9241-3

909 Finney, M. A., McHugh, C. W., Grenfell, I. C., Riley, K. L., & Short,
 910 K. C. (2011). A simulation of probabilistic wildfire risk components
 911 for the continental united states. *Stochastic Environmental Research*
 912 *and Risk Assessment*, 25(7), 973–1000. [https://doi.org/10.1007/](https://doi.org/10.1007/s00477-011-0462-z)
 913 [s00477-011-0462-z](https://doi.org/10.1007/s00477-011-0462-z)

914 Fujioka, F. M. (2002, 01). A new method for the analysis of fire spread
 915 modeling errors. *International Journal of Wildland Fire*, 11, 193–203.
 916 10.1071/WF02004

917 Ghisu, T., Arca, B., Pellizzaro, G., & Duce, P. (2015). An optimal cel-
 918 lular automata algorithm for simulating wildfire spread. *Environ-*
 919 *mental Modelling & Software*, 71, 1–14. [https://doi.org/10.1016/](https://doi.org/10.1016/j.envsoft.2015.05.001)
 920 [j.envsoft.2015.05.001](https://doi.org/10.1016/j.envsoft.2015.05.001)

921 Glorot, X., & Bengio, Y. (2010, 13–15 May). Understanding the difficulty of
 922 training deep feedforward neural networks. In Y. W. Teh & M. Titter-

ington (Eds.), *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (Vol. 9, pp. 249–256). Chia Laguna Resort, Sardinia, Italy: JMLR Workshop and Conference Proceedings.

Graves, A., Fernández, S., & Schmidhuber, J. (2007). *Multi-dimensional recurrent neural networks*. arXiv preprint, arXiv:0705.2011.

Hegde, J., & Rokseth, B. (2020, February). Applications of machine learning methods for engineering risk assessment – a review. *Safety Science*, 122, 104492. 10.1016/j.ssci.2019.09.015

Hodges, J. L., & Lattimer, B. Y. (2019). Wildland fire spread modeling using convolutional neural networks. *Fire Technology*, 55(6), 2115–2142. <https://doi.org/10.1007/s10694-019-00846-4>

Iooss, B., Boussouf, L., Feuillard, V., & Marrel, A. (2010). Numerical studies of the metamodel fitting and validation processes. *International Journal On Advances in Systems and Measurements*, 3, 11–21. Retrieved from <https://hal.archives-ouvertes.fr/hal-00444666>

Jain, P., Coogan, S. C., Subramanian, S. G., Crowley, M., Taylor, S. W., & Flannigan, M. D. (2020). A review of machine learning applications in wildfire science and management. *Environmental Reviews*. <https://doi.org/10.1139/er-2020-0019>

Johnston, P., Kelso, J., & Milne, G. J. (2008). Efficient simulation of wildfire spread on an irregular grid. *International Journal of Wildland Fire*, 17, 614–627. <https://doi.org/10.1071/WF06147>

Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, per-

spectives, and prospects. *Science*, 349(6245), 255–260. 10.1126/
science.aaa8415

Katurji, M., Nikolic, J., Zhong, S., Pratt, S., Yu, L., & Heilman, W. E.
(2015). Application of a statistical emulator to fire emission modeling.
Environmental Modelling & Software, 73, 254–259. [https://doi.org/](https://doi.org/10.1016/j.envsoft.2015.08.016)
10.1016/j.envsoft.2015.08.016

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification
with deep convolutional neural networks. In F. Pereira, C. J. C. Burges,
L. Bottou, & K. Q. Weinberger (Eds.), *Advances in neural information*
processing systems 25 (pp. 1097–1105). Curran Associates, Inc.

Lac, C., Chaboureau, J.-P., Masson, V., Pinty, J.-P., Tulet, P., Escobar, J.,
... Wautelet, P. (2018). Overview of the meso-NH model version 5.4
and its applications. *Geoscientific Model Development*, 11(5), 1929–
1969. 10.5194/gmd-11-1929-2018

Lautenberger, C. (2017). Mapping areas at elevated risk of large-scale
structure loss using monte carlo simulation and wildland fire model-
ing. *Fire Safety Journal*, 91, 768–775. (Fire Safety Science: Proceed-
ings of the 12th International Symposium) [https://doi.org/10.1016/](https://doi.org/10.1016/j.firesaf.2017.04.014)
j.firesaf.2017.04.014

Liu, Y., Hussaini, M. Y., & Ökten, G. (2016). Accurate construction of
high dimensional model representation with applications to uncertainty
quantification. *Reliability Engineering & System Safety*, 152, 281–295.
<https://doi.org/10.1016/j.ress.2016.03.021>

- 969 Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting
970 model predictions. In I. Guyon et al. (Eds.), *Advances in neural in-*
971 *formation processing systems 30* (pp. 4765–4774). Curran Associates,
972 Inc.
- 973 Mallet, V., Keyes, D., & Fendell, F. (2009). Modeling wildland fire propa-
974 gation with level set methods. *Computers & Mathematics with Appli-*
975 *cations*, 57(7), 1089–1101. <https://doi.org/10.1016/j.camwa.2008.10>
976 .089
- 977 M.S. Pinto, R., Benali, A., C.L. Sá, A., Fernandes, P. M., M.M. Soares,
978 P., Cardoso, R. M., ... M.C. Pereira, J. (2016, 07). Probabilistic
979 fire spread forecast as a management tool in an operational setting.
980 *SpringerPlus*, 5, 1205. 10.1186/s40064-016-2842-9
- 981 Parisien, M., Kafka, V., Hirsch, K., Todd, J., Lavoie, S., & Maczek, P. (2005).
982 *Mapping wildfire susceptibility with the BURN-P3 simulation model*.
983 Natural Resources Canada, Information Report NOR-X-405, Canadian
984 Forest Service, Northern Forestry Centre, Edmonton, Alberta.
- 985 Quang, D., & Xie, X. (2016). DanQ: a hybrid convolutional and recurrent
986 deep neural network for quantifying the function of DNA sequences.
987 *Nucleic Acids Research*, 44(11), e107–e107. [https://doi.org/10.1093/](https://doi.org/10.1093/nar/gkw226)
988 [nar/gkw226](https://doi.org/10.1093/nar/gkw226)
- 989 Radke, D., Hessler, A., & Ellsworth, D. (2019). FireCast: Leveraging deep
990 learning to predict wildfire spread. In *Proceedings of the twenty-eighth*
991 *international joint conference on artificial intelligence, IJCAI-19* (pp.

4575–4581). International Joint Conferences on Artificial Intelligence
Organization. <https://doi.org/10.24963/ijcai.2019/636>

Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian processes for
machine learning*. the MIT Press.

Rochoux, M. C., Ricci, S., Lucor, D., Cuenot, B., & Trouvé, A. (2014).
Towards predictive data-driven simulations of wildfire spread – part
i: Reduced-cost ensemble kalman filter based on a polynomial chaos
surrogate model for parameter estimation. *Natural Hazards and Earth
System Sciences*, 14(11), 2951–2973. <https://doi.org/10.5194/nhess-14-2951-2014>

Rothermel, R. C. (1972). *A mathematical model for predicting fire spread
in wildland fuels*. Res. Pap. INT-115. Ogden, UT: U.S. Department of
Agriculture, Intermountain Forest and Range Experiment Station. 40
p.

Salis, M., Arca, B., Alcasena, F., Arianoutsou, M., Bacciu, V., Duce, P.,
... Spano, D. (2016, 06). Predicting wildfire spread and behavior in
mediterranean landscapes. *International Journal of Wildland Fire*, 25,
1015-1032. 10.1071/WF15081

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmen-
tation for deep learning. *Journal of Big Data*, 6(1). <https://doi.org/10.1186/s40537-019-0197-0>

Stojanovic, V., He, S., & Zhang, B. (2020). State and parameter joint estima-
tion of linear stochastic systems in presence of faults and non-Gaussian

noises. *International Journal of Robust and Nonlinear Control*, 30(16),
6683–6700. 10.1002/rnc.5131

Sullivan, A. L. (2009a). Wildland surface fire spread modelling, 1990 -
2007. 1: Physical and quasi-physical models. *International Journal of*
Wildland Fire, 18(4), 349–368. <https://doi.org/10.1071/wf06143>

Sullivan, A. L. (2009b). Wildland surface fire spread modelling, 1990 -
2007. 3: Simulation and mathematical analogue models. *International*
Journal of Wildland Fire, 18(4), 387–403. <https://doi.org/10.1071/wf06144>

Thompson, M., Calkin, D., Scott, J. H., & Hand, M. (2017). Uncertainty
and probability in wildfire management decision support: An example
from the united states. In K. Riley, P. Webley, & M. Thompson (Eds.),
Natural hazard uncertainty assessment: Modelling and decision support
(first ed., Vol. 223, p. 31-41). American Geophysical Union.

Tolhurst, K., Shields, B., & Chong, D. (2008). Phoenix: Development and
application of a bushfire risk management tool. *Australian Journal of*
Emergency Management, 23(4), 47–54.

Trucchia, A., Egorova, V., Pagnini, G., & Rochoux, M. (2019). On the
merits of sparse surrogates for global sensitivity analysis of multi-scale
nonlinear problems: Application to turbulence and fire-spotting model
in wildland fire simulators. *Communications in Nonlinear Science and*
Numerical Simulation, 73, 120–145. [https://doi.org/10.1016/j.cnsns](https://doi.org/10.1016/j.cnsns.2019.02.002)
.2019.02.002

- 1038 Tymstra, C., Bryce, R., Wotton, B., Taylor, S., & Armitage, O. (2010).
 1039 *Development and structure of prometheus: the canadian wildland fire*
 1040 *growth simulation model*. Natural Resources Canada, Information Re-
 1041 port NOR-X-417, Canadian Forest Service, Northern Forestry Centre,
 1042 Edmonton, Alberta. 88 p.
- 1043 Van Wagner, C. E., & Pickett, T. L. (1985). *Equations and fortran program*
 1044 *for the canadian forest fire weather index system*. Forestry Technical
 1045 Report No. 33. Ottawa, Environment Canada, Canadian Forestry Ser-
 1046 vice, Petawawa National Forestry Institute.
- 1047 Xie, Y., Xing, F., Kong, X., Su, H., & Yang, L. (2015). Beyond classification:
 1048 Structured regression for robust cell detection using convolutional neu-
 1049 ral network. In N. Navab, J. Hornegger, W. M. Wells, & A. F. Frangi
 1050 (Eds.), *Medical image computing and computer-assisted intervention –*
 1051 *MICCAI 2015* (pp. 358–365). Cham: Springer International Publish-
 1052 ing.
- 1053 Yuan, Z., Jiang, Y., Li, J., & Huang, H. (2020). *Hybrid-dnns: Hybrid deep*
 1054 *neural networks for mixed inputs*. arXiv preprint arXiv:2005.08419.
- 1055 Zeiler, M. D. (2012). *Adadelta: An adaptive learning rate method*. arXiv
 1056 preprint arXiv:1212.5701.
- 1057 Zhou, L., Tao, H., Paszke, W., Stojanovic, V., & Yang, H. (2020, Septem-
 1058 ber). PD-type iterative learning control for uncertain spatially inter-
 1059 connected systems. *Mathematics*, 8(9), 1528. 10.3390/math8091528
- 1060 Zhou, T., Ding, L., Ji, J., Yu, L., & Wang, Z. (2020). Combined estimation of

1061 fire perimeters and fuel adjustment factors in FARSITE for forecasting
1062 wildland fire propagation. *Fire Safety Journal*, 116, 103167. [https://](https://doi.org/10.1016/j.firesaf.2020.103167)
1063 doi.org/10.1016/j.firesaf.2020.103167