



Emulation of wildland fire spread simulation using deep learning

Frederic Allaire, Vivien Mallet, Jean Baptiste Filippi

► To cite this version:

Frederic Allaire, Vivien Mallet, Jean Baptiste Filippi. Emulation of wildland fire spread simulation using deep learning. 2021. hal-03142281

HAL Id: hal-03142281

<https://inria.hal.science/hal-03142281>

Preprint submitted on 15 Feb 2021

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.

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 support
41 long-term to short-term planning of firefighting actions. Fundamentally, fire
42 spread implies heat transfer at scales of the centimeter, which is too computa-
43 tionally intensive to solve in operational conditions. Alternatively, fire spread
44 modeling can be approached by solving a front-tracking problem where we
45 focus on the propagation of the interface between burned and not burned
46 areas, aka the *fire front*, over a 2D domain that represents the landscape.
47 The growth of the burned surfaces from their initial state is governed by
48 equations involving an model of rate of spread (ROS), that is to say the
49 speed at which the flames advance, which is expressed as a function of local
50 environmental parameters. Among such solvers, marker methods consist in
51 discretizing the fire front by means of markers, which evolve in space and
52 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), that are more
67 complex and typically include heat transfer conservation laws, equations de-
68 scribing combustion chemistry, etc., have also been developed. However, their
69 use is generally limited to research purposes, because the computational time
70 for simulations based on such models is prohibitory in an operational con-
71 text, even more so for large wildfires that may burn during several hours or
72 even days and scale up to thousands of hectares.

73 There are several possible applications of simulators of wildland fire spread
74 in an operational context. In a crisis situation, when a fire has just started,
75 they can help in predicting where the fire will spread and optimizing the fire
76 suppression actions and evacuation. Prior to crisis situations, fire spread sim-
77 ulations are a major component of risk assessment frameworks to determine

78 what areas have the highest potential to host a large incident. Wildland fire
79 risk quantification generally involves models describing ignition probability,
80 the probability for a given location to be burned, and the consequences on
81 the objects affected by fire such as properties, timber production, as well as
82 the consequences on human lives, wildlife habitats, etc. Several studies fo-
83 cused on fire risk mapping at the regional or country scale (Finney, McHugh,
84 Grenfell, Riley, & Short, 2011; Lautenberger, 2017; Parisien et al., 2005),
85 where many fires are simulated to represent a fire season or year according
86 to some probabilistic distribution of ignition and environmental conditions
87 driving fire spread, and this process may be repeated hundreds of thousands
88 of times as part of a Monte Carlo method. The purpose of such maps is
89 to help in land management through the reduction of areas at risk in the
90 long-term, by setting up fire breaks and providing more firefighting resources
91 such as reservoirs, etc.

92 Regarding short-term planning, information for the next day or hours
93 about the areas where a fire is most likely to ignite and how far the resulting
94 fire may spread can be very useful to know what locations should be moni-
95 tored more closely and help in anticipating the distribution of firefighting re-
96 sources (firefighters, trucks, ...) across the territory. In such cases, numerical
97 simulations of wildland fire spread could be used to generate high-resolution
98 maps of fire spread on the basis of weather forecasts; but this would require
99 numerous computations for different ignition locations, and the constraint
100 on computational time would be too demanding even for simulators used for

101 other operational purposes. As a rough estimate for the region considered in
102 the present study, running one fire spread simulation with a computational
103 of one minute for each hectare of land would amount to a computational time
104 of 872,000 minutes (about 600 days) on a single processor, and even more
105 if an ensemble of simulations is considered for each hectare; which would be
106 too long even after distributing the computations on multiple processors.

107 In the aforementioned applications, and more particularly in short-term
108 fire danger mapping, a promising approach to reduce computational time
109 is to rely on an *emulator* (aka metamodel or surrogate model) to provide
110 an approximation of some quantity of interest derived from the simulator's
111 output. The idea is to focus on this quantity and compute it much faster
112 with the emulator at the cost of some approximation error that should be
113 as low as possible. Emulation may be used in situations when a fire spread
114 model has high computational time and/or a lot of simulations or calls of
115 a given function is required, but emulators are rarely used in wildland fire
116 research even though their potential for reducing computational time of sim-
117 ulations appears desirable in this field. Examples include data assimilation of
118 a fire front via polynomial chaos (Rochoux et al., 2014); sensitivity analysis
119 through the computation of Sobol' indices related to the area and shape of
120 the simulated burned surface with emulation by either Gaussian processes
121 (GP) or generalized polynomial chaos (Trucchia, Egorova, Pagnini, & Ro-
122 choux, 2019); uncertainty quantification and computation of Sobol' indices
123 regarding the rate of spread (ROS) model of Rothermel (Rothermel, 1972)

124 using high dimensional model representation methods (Liu, Hussaini, & Ök-
125 ten, 2016); interpolation in a cell-based wildland fire spread simulator to
126 quickly compute the values of correction factors in the relationship between
127 advection velocity and spread angle on the basis of pre-computed values ob-
128 tained in a few given configurations using a Radial Basis Function (RBF)
129 approach (Ghisu, Arca, Pellizzaro, & Duce, 2015). Another example outside
130 the scope of fire spread is the emulation of some outputs of a fire emission
131 model with GP (Katurji et al., 2015).

132 Machine learning techniques have been used in a broad range of wild-
133 land fire science applications (Jain et al., 2020). Neural networks, in par-
134 ticular, appear promising to take into account the complexity of wildland
135 fire spread. For instance, an application involving emulation is proposed
136 in (Zhou, Ding, Ji, Yu, & Wang, 2020), where a radial basis function neu-
137 ral network (RBFNN) is trained to emulate the similarity index between
138 an observed burned surface and its simulated counterpart as a function of
139 several ROS adjustment factors; a Monte Carlo procedure is then applied
140 to the emulator, providing parameter estimation of the adjustment factors
141 for data assimilation of the simulated fire front. Other methods consist in
142 using a convolutional neural network (CNN) as a surrogate for a wildland
143 fire spread simulator to obtain a map of predicted burned areas (Hodges &
144 Lattimer, 2019; Radke, Hessler, & Ellsworth, 2019). Data required to solve
145 wildfire simulations have similarities with these involved in image process-
146 ing as we are handling gridded maps of elevation and fuel parameters. As

147 deep learning proved to be very appropriate to solve such image processing
148 problems (Krizhevsky, Sutskever, & Hinton, 2012), it motivates the use of
149 deep neural networks instead of traditional emulation techniques to approach
150 emulation in wildland fire spread simulations.

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

159 The numerical simulator of wildland fire spread that is used as basis of
160 the present work is presented in Section 2 together with the characteristics
161 of the simulations. The strategy used to obtain the emulator is described in
162 Section 3 and the results are provided and discussed in Section 4. Conclu-
163 sions of this work are summarized in Section 5, where some perspectives of
164 application of the emulator and possible extensions to the method are also
165 mentioned.

166 2 Simulation of wildland fire spread

167 In the present study, wildland fire spread simulations are carried out with
168 the numerical solver ForeFire (Filippi, Morandini, Balbi, & Hill, 2010). Fore-
169 Fire relies on a front-tracking method where the fire front is represented by
170 Lagrangian markers that are linked to each other by a dynamic mesh. The
171 interface is discretized using an ordered list of Lagrangian markers at given
172 locations on the earth’s surface. The interface is then tracked by advecting
173 all these markers at the propagation velocity of the front, and by ensuring
174 that the list of markers still holds an accurate representation of the interface.
175 In this ordered list of markers, previous and next are defined by conven-
176 tion in the indirect direction as in Figure 1. The outward normal defines
177 the direction of propagation from burning regions toward unburned regions.
178 Although fronts are allowed to contain islands of unburned fuel, they must
179 remain simple polygons (with no self-intersection). A key aspect of the sim-
180 ulation is the computation of rate of spread (ROS), that is to say the speed
181 at which the flames advance. Several ROS models were proposed in the
182 scientific literature; the model used in present study is the model of Rother-
183 mel (Rothermel, 1972), which is commonly used by fire managers in the US.
184 The ROS is expressed as a function of several environmental properties such
185 as wind speed, terrain slope, fuel moisture content (FMC) and other fuel
186 parameters characterizing the vegetation. A simulation mostly consists in
187 the definition of an initial state of the fire front and the ROS is computed

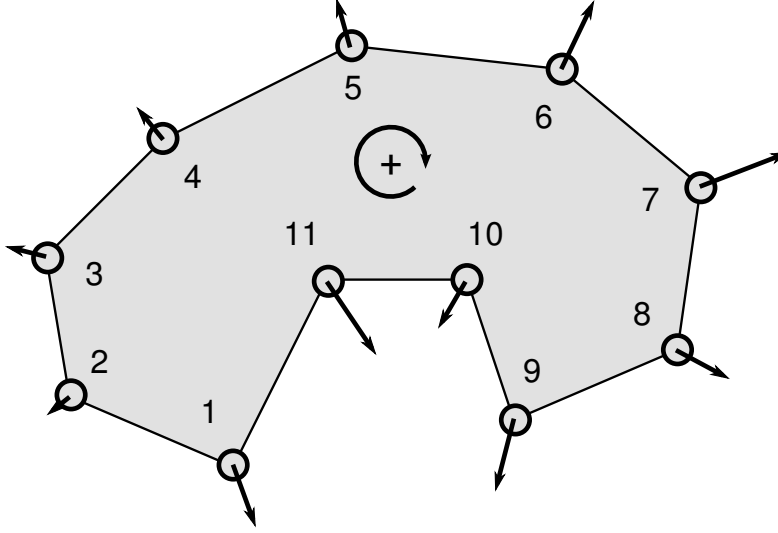


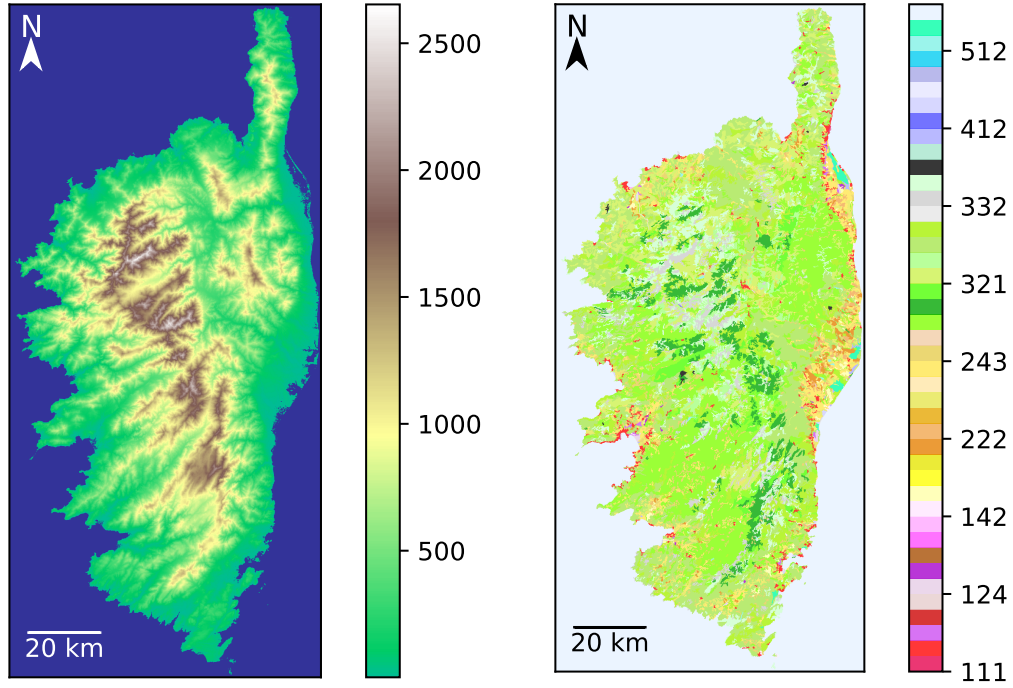
Figure 1: Example of a small fire front discretization with ordered markers.

188 for the markers of the fire front based on underlying 2D fields from which
 189 environmental properties are determined. ForeFire relies on a discrete event
 190 approach where most computations deal with the determination of the time
 191 at which the markers will reach their next destination, this destination being
 192 defined by a fixed spatial increment in the outward normal. This discrete
 193 event approach includes other types of events such as changes in the values of
 194 the layers, notably wind speed and FMC, additions and removals of markers
 195 so that the fire front maintains a perimeter resolution in a given range during
 196 the simulation, and topology checks that may induce front merging to ensure
 197 that the front keeps a physical representation.

198 The area of study is Corsica island, which is located south-east of France
 199 in the Mediterranean sea. For fire simulation on this domain, 2D fields of

elevation and land use in raster format at approximately 80-m resolution are used, and represented in Figures 2a and 2b, respectively. The land use field comes from Corine Land Cover data (Feranec, Soukup, Hazeu, & Jaffrain, 2016) coupled with data from the IGN (Institut Géographique National) product BD TOPO[®] for road and drainage networks. The elevation field is extracted from another IGN product: BD ALTI[®], which has originally a 25-m resolution. A fuel parameterization is used to assign reference fuel parameters to each type of vegetation (referred to as “fuel type” in the following) in the land use data for ROS computations. Data used for simulation also include 2D fields of wind speed vectors at a resolution of 200 m that were pre-computed for average wind speed vectors with the mass conserving preconditioner from the atmospheric forecasting system Meso-NH (Lac et al., 2018) to account for orographic effects. By specifying an average input wind speed vector in the simulations, the underlying 2D wind field is simply obtained from the pre-computed fields corresponding to the closest mean speed vectors.

In the present study, a simulation is always that of a fire with free spread (firefighting actions are not accounted for, but non-burnable areas such as water bodies may halt the progression of the fire front) during one hour. Another fixed input in the simulations is the initial fire 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.



(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

223 Several inputs in the simulations may vary from a simulation to another.
 224 First are the coordinates of the center of the initial fire front, this point
 225 being referred to as the *ignition point*, that may be located in all fuel areas in
 226 Corsica. This “high-level” input is of major importance because it determines
 227 the location where the fire starts, and the spatial fields that will influence
 228 how the fire will spread. Next are the zonal and meridional coordinates
 229 of the “forcing” wind speed vector, in m s^{-1} , that both vary in $[-35, 35]$
 230 on the condition that the wind speed norm be lower than 35 m s^{-1} . The
 231 FMC of dead fuel varies between 0.04 and 0.3. In contrast to these “raw”
 232 inputs, the remaining ones are perturbation coefficients that are applied to
 233 reference values of some fuel parameters. Perturbation in heat of combustion
 234 and particle density are additive and applied to a common reference value
 235 used for all fuel types, whereas perturbations in fuel height, fuel load or
 236 surface-volume ratio are multiplicative coefficients and, for any of these three
 237 parameters, each one of the 13 fuel types receives a specific perturbation
 238 coefficient. This amounts to 46 variable inputs in the simulations, whose
 239 information is summarized in Table 1, including the range of each variable.
 240 The simulations are meant to be used for prevision of wildfire spread in
 241 Corsica before a fire starts, at any time, so the intervals of variation of
 242 the raw inputs were chosen to account for a wide variety of environmental
 243 conditions. Moreover, in this context, there is significant uncertainty in the
 244 simulations. The weather forecasts used to predict wind speed and FMC are
 245 possible sources of uncertainty; so are model simplifications and the choice

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).

246 of a given fuel parameterization. Therefore, the intervals of variation of both
247 raw inputs and fuel parameters also account for their uncertainty range.
248 Some intervals follow those of a previous study that focused on uncertainty
249 quantification (see notably Table 1 in [Allaire, Mallet, & Filippi, 2021](#)).

250 Finally, the quantity of interest in the present study is the area in hectares
251 of the burned surface obtained at the end of the simulation, namely after a
252 free fire spread of one hour, such a surface being represented in Figure 3.
253 It is possible with ForeFire to simulate any duration of fire and obtain the
254 state of the fire front at any moment between fire start and fire end; still the
255 one-hour area alone could be a relevant information for the firefighters as it
256 provides an estimation of the potential of fire growth if a fire that starts at
257 a given location is not contained fast enough, one hour being a typical time
258 for a fire to be detected and firefighters to arrive on-site.

259 **3 Emulation with deep learning**

260 In the context of fire growth prediction mentioned in Section 2, the absence
261 of knowledge regarding the location of fire start and the uncertainty in the
262 simulation are considerable difficulties that need to be addressed. An intu-
263 itive method consists in running a large number of simulations for ignition
264 points all across the map, where some inputs are determined from weather
265 forecasts. This procedure may or may not include perturbations in the in-
266 puts other than ignition point coordinates to account for uncertainty; but in

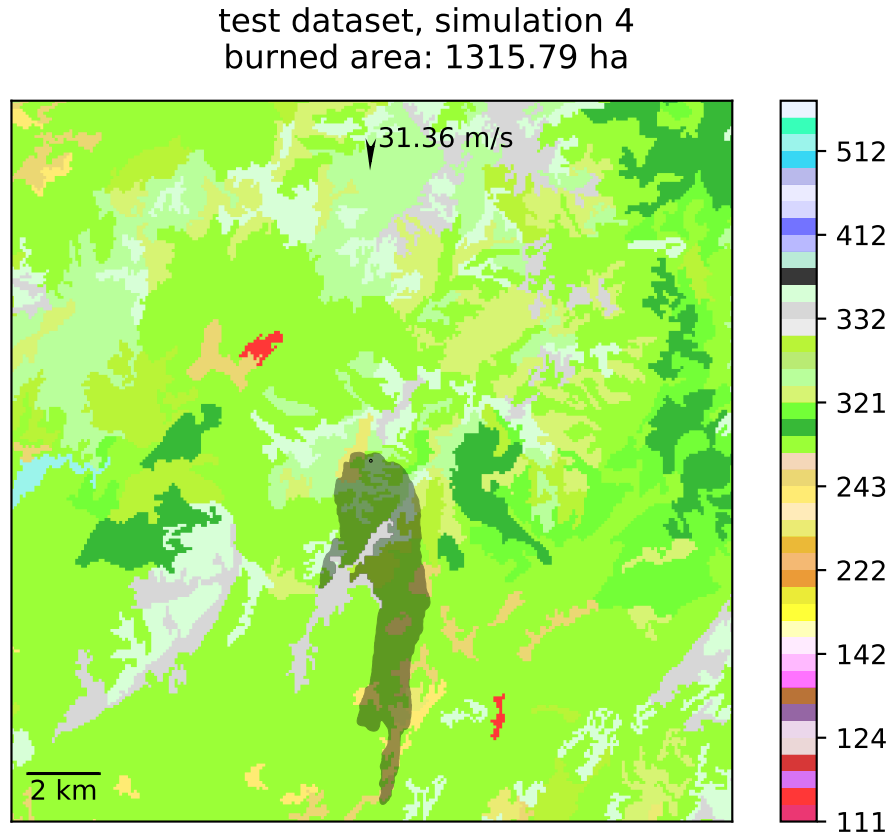


Figure 3: Example of a simulated burned surface after one hour returned by ForeFire.

The initial firefront 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

any case, the time required to run all the desired simulations in operational conditions is too high with usual numerical simulators such as ForeFire. This motivates the use of an emulator to compute the area of the output simulated burned surface in a reasonable amount of time, although with some error of approximation. It is desirable to obtain an emulator that approximates this quantity with high accuracy and has a significantly lower computational time than that of the simulator, but it can be quite challenging for an emulator to combine both properties.

3.1 Design of experiments

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

The DOE relies on a Latin Hypersquare Sample (LHS) in $[0, 1]^{46}$, which is a popular space-filling design. For all elements of the LHS, we apply an affine transformation from $[0, 1]^{46}$ to the hyperrectangle whose boundaries are defined by the ranges in Table 1. However, this procedure alone does not account for the restrictions to the definition domain implied by the con-

289 constraints on ignition point coordinates and wind speed norm. To include these
 290 constraints, we generate a LHS with more members than n_{train} , the desired
 291 number of training sample members, and keep only “valid” members, namely
 292 those that satisfy the constraints after the affine transformation, so that the
 293 resulting sample size is slightly lower than the target. The next step in the
 294 constitution of the DOE is to generate a Sobol’ sequence in $[0, 1]^{46}$. We com-
 295 plete the initial LHS (in $[0, 1]^{46}$) with members of the Sobol’ sequence based
 296 on a discrepancy criterion, following the idea proposed in (Iooss, Boussouf,
 297 Feuillard, & Marrel, 2010) to obtain an optimal complementary design. A
 298 notable difference in the present study is that the first elements selected by
 299 the algorithm are used to complete the training sample only if they are valid
 300 (they are ignored otherwise); then, when the target size n_{train} is reached, the
 301 next valid elements are used to form a test sample of size n_{test} . This proce-
 302 dure aims at selecting the points of the test sample so that they are located
 303 far from each other but also far from the points of the training sample, where
 304 the approximation error is expected to be higher.

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

308 **3.2 Neural network architecture**

309 Several techniques can be considered for emulation. Simple statistical meth-
 310 ods such as linear regression based on the inputs in Table 1 would most likely

311 lead to poor approximation because of the non-linearity of the model. Other
 312 methods such as those mentioned in Section 1, (i.e., Gaussian processes,
 313 polynomial chaos, high dimensional model reduction, radial basis functions)
 314 are interesting alternatives, however their computational requirements (re-
 315 garding time and/or memory space) can become prohibitory when there are
 316 both a high dimension ($d = 46$) and a large sample size ($\geq 10^5$).

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

333 Neural network models appear suitable for emulation of fire spread simu-

334 lations, not only because they usually perform well when trained on a large
 335 dataset, but also because they can handle several types of data. In partic-
 336 ular, CNNs proved to be quite successful in the classification of 2D inputs
 337 such as images (e.g. [Krizhevsky et al., 2012](#)), but also for regression (e.g. [Xie,
 338 Xing, Kong, Su, & Yang, 2015](#)), which is our target. Here, the simulations
 339 are also significantly influenced by the other (scalar) inputs, notably wind
 340 speed and FMC, so a network with a hybrid architecture to process both
 341 types of inputs (2D and scalar) seems well suited to our problem. The term
 342 “hybrid” may have different meanings when it comes to neural networks. It
 343 can refer to the succession of multiple ensembles of layers, with each ensem-
 344 ble appearing like a given type of neural network, as in ([Quang & Xie, 2016](#))
 345 where DNA sequences are first processed by a convolutional part then by a
 346 recurrent part; but in the present study it is understood as the use of specific
 347 types of layers for each type of input, as proposed in ([Yuan, Jiang, Li, &
 348 Huang, 2020](#)) where image, sequential, and scalar/categorical inputs are first
 349 processed separately by the network.

350 We propose an emulator based on a DNN with a hybrid architecture. A
 351 convolutional part processes the four 2D fields of elevation and fuel paramet-
 352 ers (prior to perturbation) h , σ_f and S_v in a square surrounding the ignition
 353 point with a side of approximately 20 km, which corresponds to an input of
 354 shape (256, 256, 4). Another part of the network processes the vector of
 355 size 46 of scalar simulation inputs mentioned in Table 1. The “absolute”
 356 coordinates (x, y) of the ignition point are replaced by (δ_x, δ_y) , which are

the coordinates of this point relatively to the center of the surrounding 2D fields. Also, both 2D and scalar inputs are scaled to $[-1, 1]$ through an affine transformation before being processed by the DNN.

The detailed architecture of the DNN is represented in both Figure 4 and Figure 5, where the first figure is more focused on the processing layers (i.e., convolutions, pooling, etc.), while the second represents the successive shapes of the data as they are processed by the network.

First, convolutions with a 2x2 window are applied to the 2D inputs, followed by a batch normalization layer, a Rectified Linear Unit (ReLU) activation and an average pooling layer with a 2x2 window. This succession of layers is repeated three more times, with a 3x3 window for the convolutions and more and more kernels. Convolutions are carried out without padding nor stride, and the first two average pooling layers result in the edge of the data being cropped, due to the odd input shape. Then, the output of these four blocks of layers is flattened and goes through a block consisting of a fully connected feed forward (aka dense) layer with 1024 output nodes, followed by batch normalization and ReLU activation. As for the scalar input, it goes through a similar block of layers. The output of these two blocks is concatenated and undergoes four similar blocks of layers. The intention behind the application of the dense blocks before concatenation is to concatenate vectors that have the same shape and potentially give similar importance to the 2D part and the scalar part in this mixed architecture. Finally, a dense layer followed by a ReLU activation and an increase of 0.45 ha (the minimum

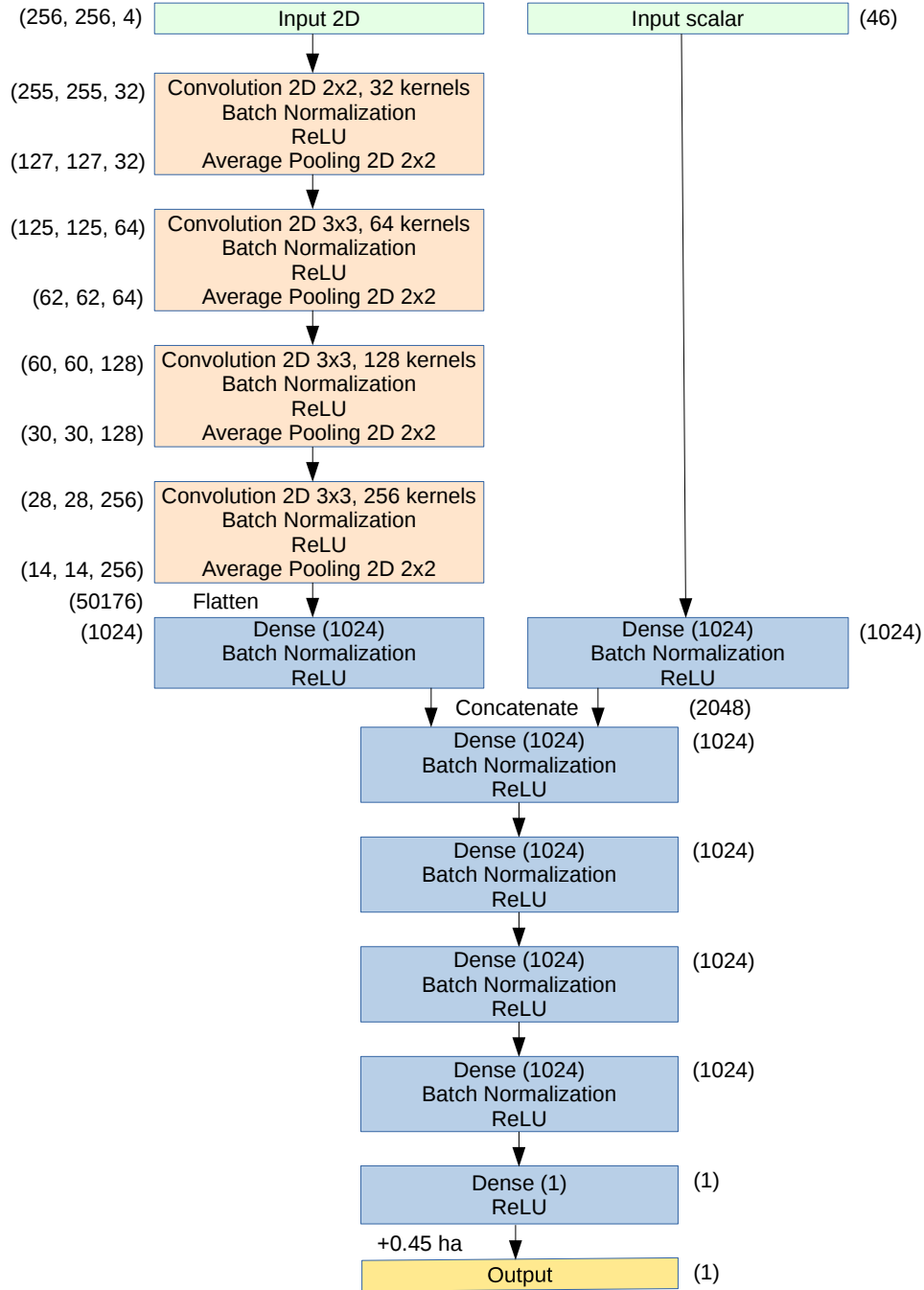


Figure 4: Neural network architecture. The numbers in brackets outside the boxes indicate the shape of the data as they are processed by the network.

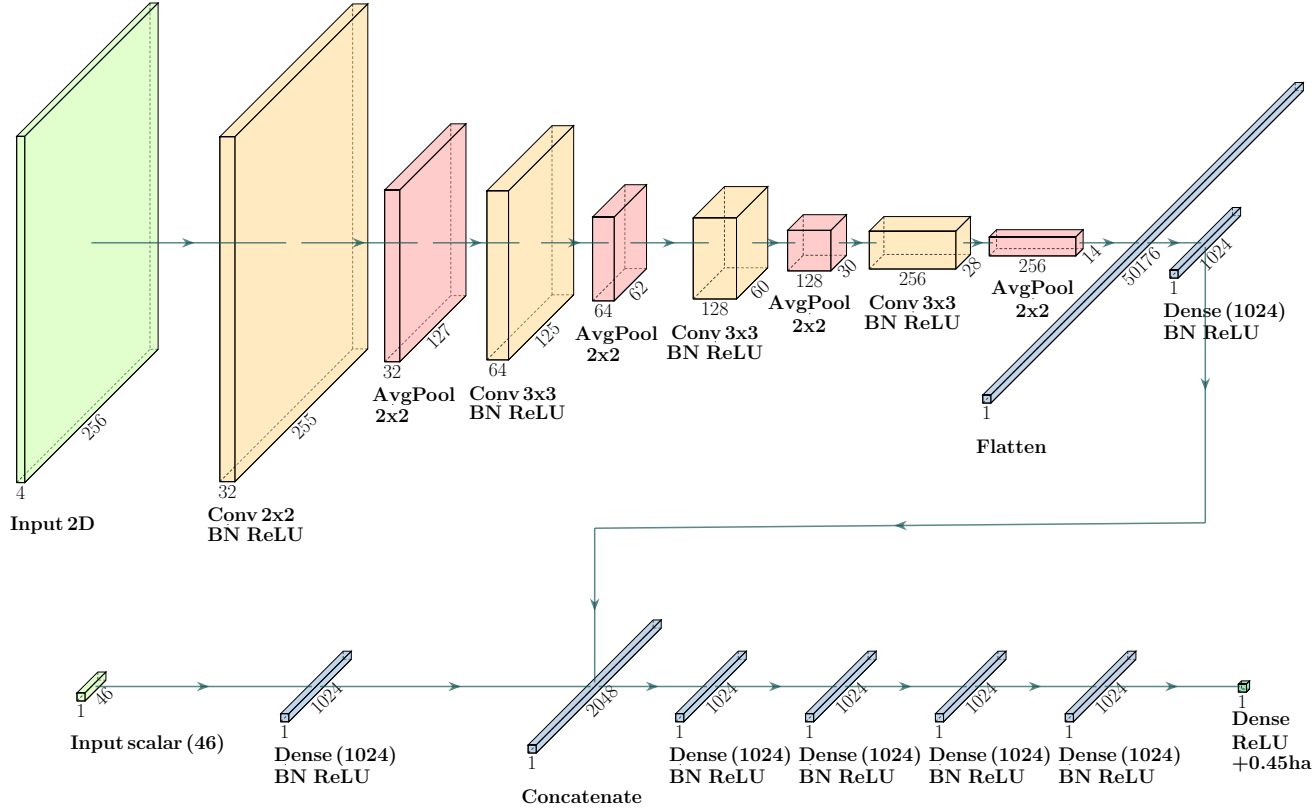


Figure 5: Representation of data processing in the neural network. The blocks indicate the shape of the data. The 2D input is derived from the four fields of elevation, and fuel parameters h , σ_f , and S_v . The 46 scalar inputs are derived from the simulation parameter inputs of Table 1. Conv: Convolution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

380 simulated burned surface area, corresponding to a fire that does not spread)
 381 are carried out, yielding the output of the network.

382 **3.3 Accuracy metrics and training strategy**

383 Among a dataset of size n , $\mathbf{u}^i$ denotes the i -th set of simulation inputs,
 384 $y(\mathbf{u}^i)$ the resulting output, and $\tilde{y}(\mathbf{u}^i)$ the corresponding value returned by
 385 the emulator. Several metrics can be used to evaluate the accuracy of $\tilde{y}$, the
 386 emulator of function y . In this study, we use the mean absolute error (MAE),
 387 the mean absolute percentage error (MAPE) and the standardized mean
 388 square error (SMSE, cf. [Rasmussen & Williams, 2006](#)), which are respectively
 389 defined as follows:

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

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

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

390 where $\bar{y} = \frac{1}{n} \sum_{i=1}^n y(\mathbf{u}^i)$ is the sample mean of the emulated function. The
 391 SMSE can be seen as a mean squared error normalized by the sample variance
 392 of y , and would be equal to 1 if the emulator was a constant function equal to
 393 the sample mean $\bar{y}$. The lower these scores, the more accurate the emulator.

394 The emulator can also be evaluated in terms of mean error, similarly to the
395 MAE but without the absolute value, that will be referred to as “bias” in the
396 following.

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

401 The procedure used to train the network’s parameters relies on a MAE
402 loss function with an Adadelata optimizer (Zeiler, 2012), without any regular-
403 ization due to layer parameters.

404 To enrich the train dataset, a form of data augmentation is carried out:
405 over one epoch, each member of the training dataset is used exactly once,
406 but possibly after a geometric transformation (rotations, axial symmetries,
407 or a combination of both). The geometric transformation is applied to the
408 2D field inputs as well as (W_x, W_y) , the wind speed vector, and (δ_x, δ_y) , the
409 relative coordinates of the ignition point. There is a 0.5 probability of having
410 no transformation, whereas the other transformations (seven different non-
411 identity applications) each have a 1/14 probability of being applied. We know
412 that in such a configuration, the simulated burned surface would be the same,
413 so this allows us to enrich the dataset (virtually, by a factor of eight) without
414 running additional ForeFire simulations, and might limit overfitting (Shorten
415 & Khoshgoftaar, 2019) since it allows for more possible configurations than
416 described in Section 2. Note that data augmentation is only used during

417 training. Also, with the synthetic datasets, there is no need to split the
418 training dataset to obtain a validation dataset, since the test dataset was
419 designed specifically to evaluate accuracy, as explained in Section 3.1. The
420 accuracy metrics of the network are simply computed for the test dataset at
421 the end of each epoch during training.

422 3.4 Extraction of the actual emulator

423 The DNN presented in Section 3.2 relies on many convolutions that can
424 be computed much faster with high-performance graphics cards. However,
425 such computational resources may not be available in an operational context,
426 making the DNN unsuited for emulation due to its high computational time.

427 In order to circumvent this issue, the final layer of the convolutional part
428 of the network (of size 1024), before concatenation with the scalar part, is
429 pre-computed. Indeed, due to the spatial resolution of the elevation and land
430 cover fields of approximately 80 m, there is a finite amount of possibilities for
431 the 2D input and the subsequent layers up to the end of the convolutional
432 part, which will take the same values as long as the ignition point is located
433 in a given cell of side ~ 80 m. In the present case, there are $\sim 1.2 \times 10^6$
434 possibilities for Corsica.

435 The actual emulator consists in the remaining part of the DNN and its
436 inputs are the pre-computed final layer of the convolutional part as well as
437 the scalar vector of size 46. This part of the network only involves some
438 dense blocks and a concatenation of the two parts of the network, that can

439 be computed much faster—even on a machine without specific acceleration.

440 **3.5 Implementation**

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

445 Training and accuracy evaluation of the DNN up to the retrieval of the
446 actual emulator are carried out on a GPU accelerated compute node. The
447 computational time of the actual emulator is evaluated on a machine with
448 32 CPU.

449 The size of the datasets are $n_{\text{train}} = 5 \times 10^6$ and $n_{\text{test}} = 10^4$. Training is
450 carried out for 100 epochs with batches of size 400, and the hyperparameters
451 of the Adadelata optimizer are a decay rate of 0.95, a conditioning constant ϵ
452 of 10^{-7} , and a learning rate of 0.3, which is an extra factor in the right-hand
453 term of Equation (14) in (Zeiler, 2012).

454 **4 Results and discussion**

455 The computational time of a simulation (with ForeFire) of wildland fire
456 spread took an average of approximately 25 s. This time highly depends
457 on the input of the simulation and can range from about 0.1 second to more
458 than an hour. Overall, the larger the simulated burned surface, the more

459 computations are carried out during the simulation. Given the simulation
 460 settings presented in Section 2, the obtained burned surface areas range from
 461 0.45 ha to 24 804.4 ha among the training dataset. Some statistics of this out-
 put in the training dataset are presented in Table 2. The high variance of

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.

462
 463 the simulation output is consistent with that of computational time. The
 464 minimum output corresponds to the area of the initial burned surface and
 465 is obtained in a few simulations (approximately half a thousandth) where
 466 the FMC is very close to the moisture of extinction (0.3) in the ROS model,
 467 leading to a fire that almost does not spread. Similar statistics are obtained
 468 with the test dataset, except for the maximum output (14 403.7 ha). The test
 469 dataset, having a much lower size than that of the training dataset, is less
 470 representative of tail of the output distribution, hence the lower maximum.

471 Most simulations result in a burned surface of less than 1000 ha, which is
 472 realistic for a fire that spreads freely during one hour. Still, a non-negligible
 473 amount of simulations result in burned surfaces that are most certainly bigger
 474 than what would be observed in reality; and this amount would probably be
 475 higher were it not for non-burnable zones that significantly contribute to limit

476 fire spread in some cases. This is mostly due to the fact that the simulations
477 rely on simplifying assumptions where wind speed and FMC are constant
478 in time and the DOE allows these inputs to vary in very large intervals.
479 Therefore, it is not surprising to obtain a very large burned surface in a
480 simulation where the wind speed is extremely high, the FMC extremely low,
481 and no unburnable zone is reached during a whole hour of spread. Although
482 somewhat unrealistic, the extremely high values of simulated burned surfaces
483 were not removed from the dataset. This might make the emulation more
484 difficult but the ability to discriminate between a wide range of situations,
485 even extreme ones, is relevant in wildland fire spread.

486 The evolution of the MAE over training of the DNN for 100 epochs is
487 reported in Figure 6. At a given epoch, the predicted values for both test
488 and training datasets result from the model obtained at the epoch's end. Due
489 to high computational time, the MAE was only computed for the training
490 dataset (without applying data augmentation) at the first epoch and every
491 five epoch starting from the fifth. On the one hand, the MAE for the test
492 dataset decreases overall until it reaches 81.5 ha after about 78 epochs after
493 which it oscillates around that value. On the other hand, the MAE for the
494 training dataset decreases overall, faster than the MAE of the test dataset,
495 so while both scores are almost identical at the start the gap between the
496 two increases with the number of epochs.

497 It appears that the increasing overfitting of the network does not induce
498 lower accuracy over the test dataset. Also, it is unlikely that carrying out

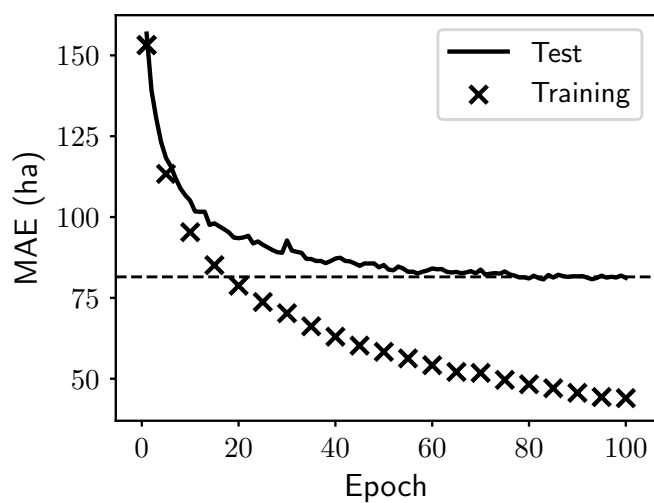


Figure 6: MAE over training. The solid curve represents the MAE for the test dataset, while the crosses represent the MAE computed for the training dataset at the end of the first epoch and after every five epochs starting from the fifth. The horizontal dotted line corresponds to $\text{MAE}=81.5$ ha.

more training epochs would result in a significant decrease of the error metrics for the test dataset. Consequently, the model with the best SMSE over the test set, which was obtained at the end of the 94-th epoch, was selected to define the emulator. The emulator with the best MAE was not selected because its MAE was only slightly lower (80.7 ha instead of 81.2 ha), while the other scores were all better for the model with the best SMSE.

The error metrics of the emulator are reported in Table 3 and Table 4, respectively relating to the test dataset and the training dataset. The

Model \ Metric	MAE	MAPE	SMSE	Bias
Mean of training	461.9 ha	2266.0%	100.0%	2.2 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 .

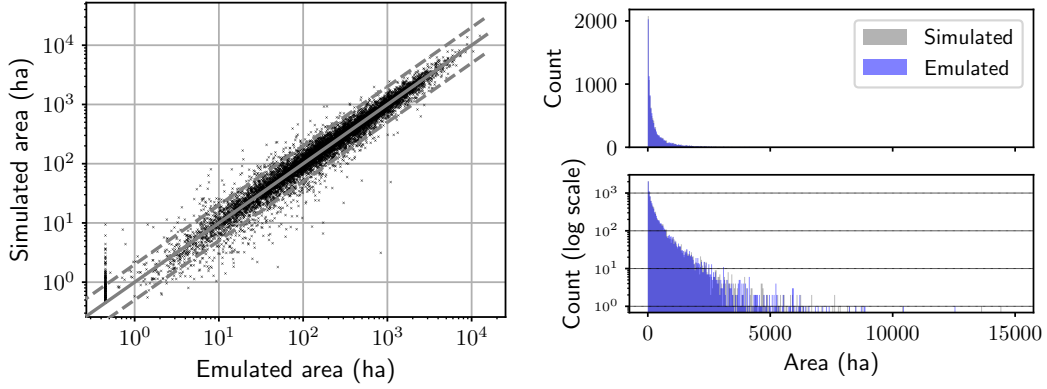
Model \ Metric	MAE	MAPE	SMSE	Bias
Mean of training	461.5 ha	2139%	100.0%	0 ha
DNN after 100 epochs	44.0 ha	23.8%	1.2%	−7.6 ha
Emulator (from DNN after 94 epochs)	45.1 ha	23.2%	1.2%	−0.9 ha

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

metrics obtained with a simplistic model that consists in always predicting the mean simulated burned surface of the training dataset (455.7 ha) are reported for comparison, as well as these of the DNN with the parameters

510 obtained at the end of training. Although a MAE of 81.2 ha might seem high,
511 it is much lower compared to that of the simplistic model (461.9 ha). The
512 SMSE of 6.0% means that 94.0% of the variance in the test dataset output
513 is explained by the emulator, which is very good given the range of variation
514 in simulation inputs. The relative error is also satisfactory with a MAPE of
515 32.8% on the test dataset, especially when compared to that of the simplistic
516 model (2266.0%). As for computational time on a 32-CPU machine, the
517 outputs for the test dataset are obtained in about half a second with the
518 emulator against 56 s with the whole DNN, which corresponds to a speed-
519 up by a factor of about 100. Also, the corresponding ForeFire simulations
520 would have been obtained in about two hours with parallel computations
521 on the 32-CPU machine, meaning that the emulator allows a speed-up by
522 about 15,000 times. For a dataset where the simulated burned surface tends
523 to be higher, the average computational time with ForeFire could be higher,
524 which is not the case for the emulator for which computational time does not
525 depend on the output fire size, meaning that the resulting speed-up factor
526 would be higher.

527 For more insight regarding the approximation, the emulator output for
528 each member of the test dataset is plotted against the actual values of sim-
529 ulated burned area in Figure 7. The vast majority of the emulated values
530 are close to their simulated counterparts and 9,332 out of 10,000 are at most
531 either twice higher or half lower. In 157 cases, the emulator returns the
532 minimum value of 0.45 ha, while the actual simulated value may go up to



(a) Individual burned areas in log scale. (b) Histograms of burned areas.

Figure 7: Comparison between the burned area simulated by ForeFire and the corresponding emulator output over the test dataset of size 10^4 .

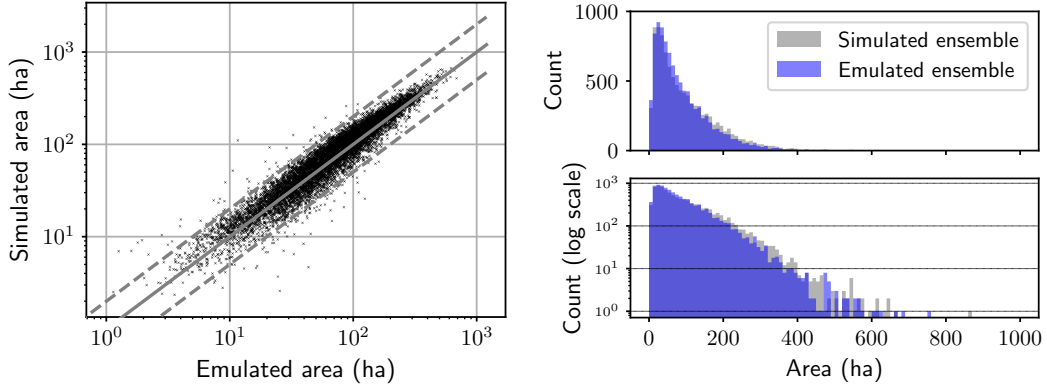
(a) The solid oblique gray line corresponds to a perfect match and the dotted lines correspond to an error by a factor of 0.5 and 2.

(b) Light gray: simulated area; blue: emulated area. 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.

533 10 ha; this corresponds to the apparent “black vertical bar” at the lower left
 534 of the graph in Figure 7a. There are 29 simulations for which the emulated
 535 burned area is at least five times lower (11 of them being equal to 0.45 ha)
 536 and 43 simulations for which the emulated value is at least five times higher.
 537 In the latter cases, most of the simulated burned surfaces are small (≤ 10 ha
 538 in 32 simulations out of 43), which usually contributes to a higher relative
 539 error; but not all of them. In some of these cases of overprediction by the
 540 emulator, there is a relatively small area close to the ignition point in the
 541 main direction of fire spread that seems to considerably slow down the fire.
 542 The emulator probably has difficulty when it comes to accounting for some
 543 particular configurations of the underlying fuel and altitude fields, especially
 544 small non-burnable areas, given that the convolutional part of the DNN re-
 545 duces the size of inputs by a factor of 256 when processing it for the emulator
 546 (from 262,144 to 1024). Overall, the individual errors lead to similar distri-
 547 butions of burned area as the emulator has a small bias of -6.5 ha and, as
 548 shown in Figure 7b, the histogram of emulated burned areas is slightly less
 549 dispersed (standard deviation of 752.9 ha against 782.5 ha).

550 The emulator is also evaluated with an ensemble of ForeFire simulations
 551 that correspond to a real Corsican fire that occurred near Calenzana during
 552 summer 2017 and burned about 120 ha. Most of the spread for this fire
 553 took place during the first hour after ignition. For this case, some reference
 554 inputs are defined from weather predictions and a presumed ignition point
 555 is identified, as explained in (Allaire, Filippi, & Mallet, 2020). Then, an

556 ensemble of perturbed simulations is generated, where the inputs presented
 557 in Table 1 follow a calibrated distribution that was obtained in a previous
 558 study (Allaire et al., 2021) with $\beta = 1/2$. It should be noted that the
 559 resulting ensemble of burned surface areas in the present study is not the
 560 same as in (Allaire et al., 2021) because supplementary inputs were variable
 561 in the previous study (such as perturbations in the times of fire start and fire
 562 end, which could make the simulated fire duration different from one hour).
 563 The 10,000 simulated burned surface areas of the ensemble are compared
 564 to their emulated counterparts in Figure 8. Similarly to the test dataset,
 565 most emulated values fall into the range of half to twice the simulated value,
 566 leading to a MAPE of 22.7%. A MAE of 18.7 ha is obtained and individual
 567 errors result in a distribution of the emulator output that is less dispersed
 568 than that of the simulated output, as shown in Figure 8b, with a bias of
 569 -9.6 ha and a standard deviation of 77.7 ha against 86.1 ha. The overall
 570 agreement between simulation and emulation is good for this simulated fire
 571 case, and the simulations were computed in 20 minutes while the emulator
 572 predictions only took a bit more than a second. The speed-up factor is about
 573 1000 this time, which is lower than the several thousands obtained for the
 574 test dataset; this is explained by the lower simulation time for this fire case
 575 (20 min instead of about two hours for the test dataset). This performance
 576 is quite promising for application to ensemble forecasting, but care should
 577 be taken as propagation of uncertainty leads to different output distributions
 578 according to the model (either ForeFire or its emulator) used.



(a) Individual burned areas in log scale. (b) Histograms of burned areas.

Figure 8: Comparison between the ensemble of burned areas simulated by ForeFire for the fire case of Calenzana and their emulated counterparts.

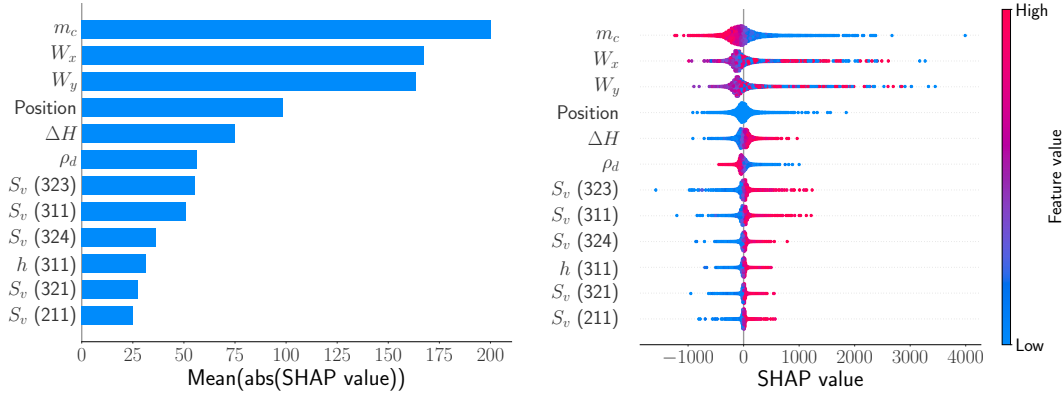
(a) The solid gray line corresponds to a perfect match and the dotted lines correspond to an error by a factor of 0.5 and 2.

(b) Light gray: simulated area; blue: emulated area. 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.

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

590 Given the complexity of the emulator, one may approach it as a black-box
 591 and estimate the overall influence of its inputs with Shapley additive expla-
 592 nations (SHAP, cf. [Lundberg & Lee, 2017](#)), a feature attribution method.
 593 The features we focus on are the inputs of the emulator, namely the 1024
 594 “position” scalars linked to the 2D fields surrounding the ignition point stem-
 595 ming from the convolutions and the remaining 46 scalar inputs. Approximate
 596 SHAP values are computed for each member of the test dataset by means of
 597 expected gradients; this procedure relies on the assumption that the model to
 598 explain is linear and that the features are independent. While these assump-
 599 tions are not verified with the emulator, this method allows for computation
 600 of approximate SHAP values in a reasonable amount of time. Although these
 601 values should be taken with care when analyzed individually they can still

602 provide some insight on the overall input influence over a whole dataset. For
 603 each member of the test dataset, the expected gradient is estimated based on
 604 a subset of size 50,000 sampled randomly from the training dataset. Given
 605 that the 1024 position scalars are difficult to interpret and expected to have
 606 little individual influence on the output due to their correlation, we consider
 607 the sum of their SHAP values, which is identified via a fictitious variable
 608 named “Position”. The approximate SHAP values obtained for 12 of the 47
 609 resulting variables are summarized in Figure 9. The values obtained for each
 610 of the 10,000 test members represented in Figure 9b indicate a good overall
 611 agreement with the main characteristics of the fire behavior model. High
 612 FMC (m_c) tends to decrease the output while low FMC tends to increase it.
 613 High positive SHAP values for the coordinates of wind speed (W_x and W_y)
 614 are obtained for extreme values of these inputs, i.e. close to either -35 m s^{-1}
 615 or 35 m s^{-1} (in blue and red, respectively) while the negative values are ob-
 616 tained for intermediate values (close to 0 m s^{-1}). SHAP values associated to
 617 the perturbation of ΔH are also consistent with our expectations. Regard-
 618 ing the rankings of the inputs when looking at the absolute SHAP values
 619 averaged over the test dataset in Figure 9a, the three most influential inputs
 620 are the FMC and the coordinates of wind speed. Position is ranked fourth,
 621 perturbations on fuel parameters that affect all fuel types (ΔH and ρ_d) are
 622 ranked fifth and sixth, and the remaining ranks are attributed to the other
 623 perturbations of fuel parameters, as well as δ_x and δ_y (ranked last). Interest-
 624 ingly, when the positional inputs are not summed, their individual influence



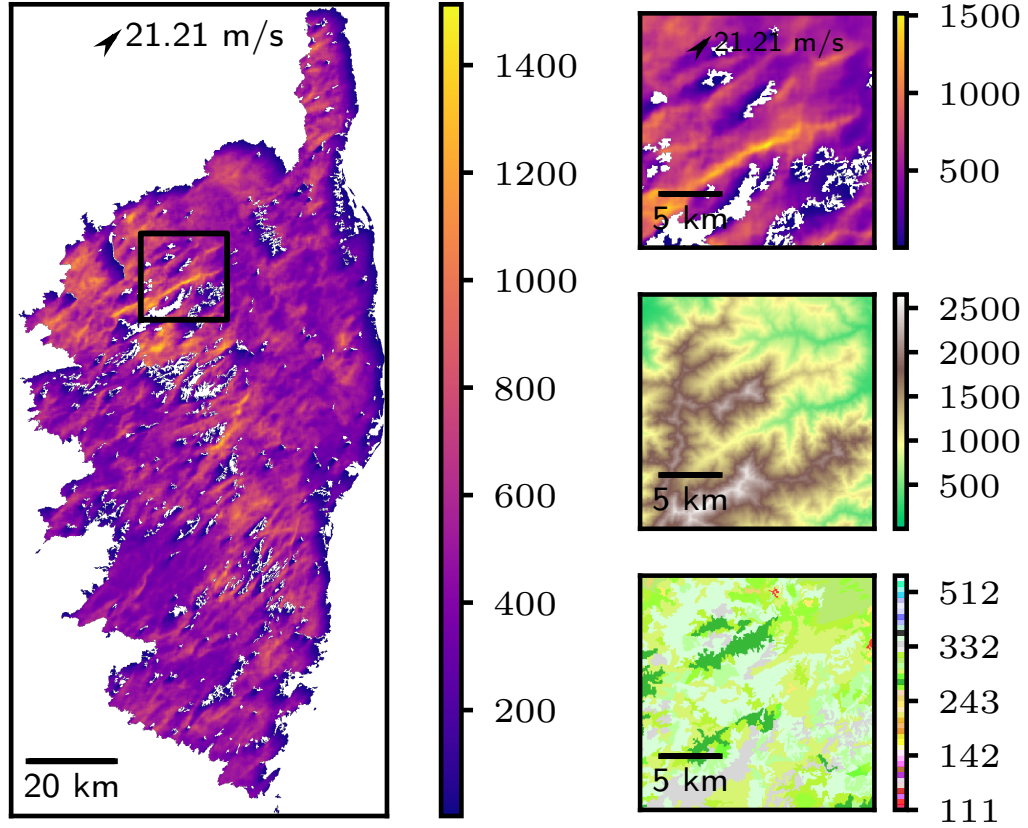
(a) Mean of the absolute value over the test dataset.

(b) Individual values for each member of the test dataset.

Figure 9: 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.

625 is quite low: the 54th scalar of the vector of size 1024 is the highest ranked
 626 at rank 32 only. Although we only have an approximation of SHAP values,
 627 these results are qualitatively the ones we would expect from fire spread sim-
 628 ulations and indicate that the emulator has an overall relationship between
 629 inputs and output that is fairly consistent with typical behavior of wildland
 630 fire spread.

631 The “physical” behavior of the emulator is also analyzed through the lens
 632 of fire danger mapping in Figure 10, that represents the response surface of
 633 the emulator where the ignition point varies in Corsica on grid of the al-
 634 titude field, whereas the other inputs are fixed to $m_c = 0.13$, $(W_x, W_y) =$
 635 $(15, 15) \text{ ms}^{-1}$, and no perturbation on fuel parameters. This mapping in-
 636 volves $\sim 1.2 \times 10^6$ emulator computations, which are carried out in about
 637 40s only. Values lower than 200 ha can be observed toward the south-west of
 638 non-burnable areas (mostly water bodies, rocky mountain tops over 1800 m
 639 with no vegetation, and urban areas), while most of the other ignition points
 640 are associated to values higher than 300 ha; which is consistent with the
 641 input wind speed vector pointing to the north-east. Also, there is a fairly
 642 high spatial variation of the emulated burned area that goes up to about
 643 1500 ha. The smaller region shown in Figure 10b presents some of the high-
 644 est values. Compared to the underlying 2D fields of altitude and fuel used
 645 in the simulations does not reveal clear influence of either one of these fields
 646 on the emulated output (except for the ignition points to the south-west of
 647 non-burnable locations). An animated version of Figure 10a with varying



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

Figure 10: 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.

wind is available as Supplementary material. Considering that the approximation errors of the emulator are relatively low, it appears that, overall, the map generated using the emulator highlights locations where ignition would induce larger burned areas.

5 Conclusions

The basis for the present study was simulations of wildland fire spread with the numerical solver ForeFire using the underlying ROS model of Rothermel. These simulations represented free fire spread during one hour from a small initial burned surface located at all possible areas in Corsica island. The terrain was represented by 2D fields of fuel and altitude at approximately 80-m resolution in the simulations. Some environmental input parameters, namely FMC, wind speed, and perturbation of fuel parameters, were also allowed to vary in a wide range. ForeFire simulations can be computed in a reasonable amount of time, yet too high for applications that require a large number of simulations on a daily basis. This motivated the use of an emulator in order to faster compute an approximation of the output simulated burned area (in hectares).

The proposed approach consisted in training a DNN used for regression. The network has a hybrid architecture to deal with 2D fields of environmental parameters and with scalar inputs. On the one hand, the 2D fields are restricted to a square of 20 km side centered around the ignition point to filter

669 out information that is, for the most part, not used during the simulation,
 670 and go through convolutional blocks due to their similarity to images. On
 671 the other hand, the remaining scalar inputs go through dense blocks, are
 672 concatenated with last layer of the convolutional part, followed by more
 673 dense blocks. Training was carried out with a large dataset of size 5×10^6
 674 obtained from a LHS sample, which could be augmented during training, and
 675 a complementary test sample of size 10^4 was obtained from a low-discrepancy
 676 sequence.

677 Although training resulted in some overfitting, this did not seem to have
 678 any adverse impact on the emulator prediction in the test dataset. The last
 679 layer of the convolutional part of the DNN for all fuel cells ($\sim 1.2 \times 10^6$)
 680 of the map of Corsica for which ignition is possible in the simulation is pre-
 681 computed. This allows to reduce computational time since the resulting
 682 positional information can be used together with the scalar inputs to run
 683 computations with only the remaining part of the DNN, which was chosen
 684 as emulator of burned surface area. The emulator showed satisfactory perfor-
 685 mance. In the test dataset, it explains 94.0% of the variance of the output,
 686 it has a MAPE of 32.8%. Also, compared to the ForeFire simulations for
 687 fire danger mapping, the emulator computations are carried out thousands
 688 of times faster on a 32-CPU machine. Finally, the overall influence of the
 689 inputs on emulator output seems consistent with typical behavior of wildland
 690 fire spread.

691 Preliminary results suggest that the emulator is suited to ensemble pre-

dictions and fire danger mapping, notably due to the considerable speed-up factor. For instance, 1.2 million ForeFire simulations requiring 25 s on average would be computed in more than 10 days on a 32-CPU machine, while this took about 40 s with the emulator, that is to say more than 20,000 times faster. A major research perspective consists in evaluating the emulator for use in these applications but now in a more extensive manner, namely with actual weather forecasts that cover the whole island, generating danger maps for every hour (at least) during an entire fire season, and considering several real fire cases. 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 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 longer than 1 hour, provided that it is trained with realistic weather time series, the definition of which is not obvious. As for network architecture,

715 upsampling layers could be considered, hoping that they would re-constitute
716 a good raster approximation of the burned surface which could be used di-
717 rectly as output (as in [Hodges & Lattimer, 2019](#)) or as the layer previous
718 to the final output node estimating the number of hectares burned. Also,
719 multi-dimensional recurrent neural networks ([Graves, Fernández, & Schmid-](#)
720 [huber, 2007](#)) could be considered as substitute for the convolutional part
721 of the DNN. Regardless of the complexity of the emulator, the main prop-
722 erties to pursue remain the same: low approximation error and decreased
723 computational time.

724 Acknowledgments

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

727

728 This work was carried out using HPC resources from GENCI-IDRIS (Grant
729 2020-AD011011630).

References

- 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>
- Feranec, J., Soukup, T., Hazeu, G., & Jaffrain, G. (2016). *European landscape dynamics: Corine land cover data*. Boca Raton, USA: CRC Press.
- Filippi, J.-B., Morandini, F., Balbi, J. H., & Hill, D. R. (2010). Discrete event front-tracking simulation of a physical fire-spread model. *SIMULATION*, 86(10), 629–646. <https://doi.org/10.1177/0037549709343117>
- Finney, M. A. (1998). *Farsite: Fire area simulator-model development and evaluation*. Res. Pap. RMRS-RP-4, Revised 2004, Ogden, UT: U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station. 47 p.
- Finney, M. A., McHugh, C. W., Grenfell, I. C., Riley, K. L., & Short, K. C. (2011). A simulation of probabilistic wildfire risk components for the continental united states. *Stochastic Environmental Research and Risk Assessment*, 25(7), 973–1000. <https://doi.org/10.1007/s00477-011-0462-z>

- 752 Ghisu, T., Arca, B., Pellizzaro, G., & Duce, P. (2015). An optimal cel-
 753 lular automata algorithm for simulating wildfire spread. *Environ-*
 754 *mental Modelling & Software*, 71, 1–14. [https://doi.org/10.1016/](https://doi.org/10.1016/j.envsoft.2015.05.001)
 755 [j.envsoft.2015.05.001](https://doi.org/10.1016/j.envsoft.2015.05.001)
- 756 Graves, A., Fernández, S., & Schmidhuber, J. (2007). *Multi-dimensional*
 757 *recurrent neural networks*. arXiv preprint, arXiv:0705.2011.
- 758 Hodges, J. L., & Lattimer, B. Y. (2019). Wildland fire spread modeling
 759 using convolutional neural networks. *Fire Technology*, 55(6), 2115–
 760 2142. <https://doi.org/10.1007/s10694-019-00846-4>
- 761 Iooss, B., Boussouf, L., Feuillard, V., & Marrel, A. (2010). Numerical studies
 762 of the metamodel fitting and validation processes. *International Jour-*
 763 *nal On Advances in Systems and Measurements*, 3, 11–21. Retrieved
 764 from <https://hal.archives-ouvertes.fr/hal-00444666>
- 765 Jain, P., Coogan, S. C., Subramanian, S. G., Crowley, M., Taylor, S. W., &
 766 Flannigan, M. D. (2020). A review of machine learning applications
 767 in wildfire science and management. *Environmental Reviews*. [https://](https://doi.org/10.1139/er-2020-0019)
 768 doi.org/10.1139/er-2020-0019
- 769 Johnston, P., Kelso, J., & Milne, G. J. (2008). Efficient simulation of wildfire
 770 spread on an irregular grid. *International Journal of Wildland Fire*, 17,
 771 614–627. <https://doi.org/10.1071/WF06147>
- 772 Katurji, M., Nikolic, J., Zhong, S., Pratt, S., Yu, L., & Heilman, W. E.
 773 (2015). Application of a statistical emulator to fire emission modeling.
 774 *Environmental Modelling & Software*, 73, 254–259. <https://doi.org/>

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 modeling. *Fire Safety Journal*, 91, 768–775. (Fire Safety Science: Proceedings of the 12th International Symposium) <https://doi.org/10.1016/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>

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. In I. Guyon et al. (Eds.), *Advances in neural information processing systems 30* (pp. 4765–4774). Curran Associates, Inc.

Mallet, V., Keyes, D., & Fendell, F. (2009). Modeling wildland fire propa-

798 gation with level set methods. *Computers & Mathematics with Appli-*
799 *cations*, 57(7), 1089–1101. <https://doi.org/10.1016/j.camwa.2008.10>
800 .089

801 Parisien, M., Kafka, V., Hirsch, K., Todd, J., Lavoie, S., & Maczek, P. (2005).
802 *Mapping wildfire susceptibility with the BURN-P3 simulation model*.
803 Natural Resources Canada, Information Report NOR-X-405, Canadian
804 Forest Service, Northern Forestry Centre, Edmonton, Alberta.

805 Quang, D., & Xie, X. (2016). DanQ: a hybrid convolutional and recurrent
806 deep neural network for quantifying the function of DNA sequences.
807 *Nucleic Acids Research*, 44(11), e107–e107. [https://doi.org/10.1093/](https://doi.org/10.1093/nar/gkw226)
808 [nar/gkw226](https://doi.org/10.1093/nar/gkw226)

809 Radke, D., Hessler, A., & Ellsworth, D. (2019). FireCast: Leveraging deep
810 learning to predict wildfire spread. In *Proceedings of the twenty-eighth*
811 *international joint conference on artificial intelligence, IJCAI-19* (pp.
812 4575–4581). International Joint Conferences on Artificial Intelligence
813 Organization. <https://doi.org/10.24963/ijcai.2019/636>

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

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

821 -14-2951-2014

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

826 Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmen-
827 tation for deep learning. *Journal of Big Data*, 6(1). [https://doi.org/](https://doi.org/10.1186/s40537-019-0197-0)
828 10.1186/s40537-019-0197-0

829 Sullivan, A. L. (2009a). Wildland surface fire spread modelling, 1990 -
830 2007. 1: Physical and quasi-physical models. *International Journal of*
831 *Wildland Fire*, 18(4), 349–368. <https://doi.org/10.1071/wf06143>

832 Sullivan, A. L. (2009b). Wildland surface fire spread modelling, 1990 -
833 2007. 3: Simulation and mathematical analogue models. *International*
834 *Journal of Wildland Fire*, 18(4), 387–403. [https://doi.org/10.1071/](https://doi.org/10.1071/wf06144)
835 wf06144

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

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

844 .2019.02.002

845 Tymstra, C., Bryce, R., Wotton, B., Taylor, S., & Armitage, O. (2010).
846 *Development and structure of prometheus: the canadian wildland fire*
847 *growth simulation model*. Natural Resources Canada, Information Re-
848 port NOR-X-417, Canadian Forest Service, Northern Forestry Centre,
849 Edmonton, Alberta. 88 p.

850 Xie, Y., Xing, F., Kong, X., Su, H., & Yang, L. (2015). Beyond classification:
851 Structured regression for robust cell detection using convolutional neu-
852 ral network. In N. Navab, J. Hornegger, W. M. Wells, & A. F. Frangi
853 (Eds.), *Medical image computing and computer-assisted intervention –*
854 *MICCAI 2015* (pp. 358–365). Cham: Springer International Publish-
855 ing.

856 Yuan, Z., Jiang, Y., Li, J., & Huang, H. (2020). *Hybrid-dnns: Hybrid deep*
857 *neural networks for mixed inputs*. arXiv preprint arXiv:2005.08419.

858 Zeiler, M. D. (2012). *Adadelata: An adaptive learning rate method*. arXiv
859 preprint arXiv:1212.5701.

860 Zhou, T., Ding, L., Ji, J., Yu, L., & Wang, Z. (2020). Combined estimation of
861 fire perimeters and fuel adjustment factors in FARSITE for forecasting
862 wildland fire propagation. *Fire Safety Journal*, 116, 103167. [https://](https://doi.org/10.1016/j.firesaf.2020.103167)
863 doi.org/10.1016/j.firesaf.2020.103167