Assessing seismic-like events prediction in model knits with unsupervised machine learning

Adèle Douin, Samuel Poincloux, Jean-Philippe Bruneton, Frédéric Lechenault

Assessing seismic-like events prediction in model knits with unsupervised machine learning.

Adèle Douin[a], Samuel Poincloux[b], Jean-Philippe Bruneton[c], Frédéric Lechenault[a,*]

[a] *Laboratoire de Physique de l'Ecole Normale Supérieure, ENS, PSL Research University, CNRS, Sorbonne University, Université Paris Diderot, Sorbonne Paris Cité, 75005 Paris, France.,*

[b] *Department of Physics, The University of Tokyo, 7-3-1 Hongo, Tokyo 113-0033, Japan.,*

[c] *Laboratoire Interdisciplinaire des Energies de Demain (LIED), CNRS UMR 8236, Université Paris Cité, 75013 Paris, France.,*

Abstract

Knitted fabric exhibits avalanche-like events when deformed, which we are interested in predicting, by analogy with earthquakes. However, as in most analogous seismic models, the time intermittence and scale-invariance of these events severely jeopardize this endeavor. But more importantly, such predictions are hard to assess and not easily compared. Here we introduce a framework that allows not only to predict seismic-like, rebalanced time series, but to also evaluate and compare the relevance of competing predictions. It relies on a reinforcement learning environment that learns risk management in a model, seismically active city subjected to the crackling dynamics observed in the mechanical response of knitted fabric. Relying on extensive experimental data, we show that this mechanism allows to assess the finite predictibility of seismic-like activity, and to compare the performance of different approaches in the operational usage of such predictions.

Keywords: crackling noise, knitted fabric, seismicity, machine learning

*frederic.lechenault@phys.ens.fr

1. Introduction

A few years ago, the scientific community seemed outdistanced by the tremendous achievements of corporate machine learning research eg. in games [1–3] or in natural language processing [4, 5]. Yet the momentum impulsed by these early developments has now galvanized a growing number of fields into the use of deep learning in particular. Ranging through robotics and computer vision [6], exoplanet search [7], particle physics [8], cancer detection [9], protein folding [10], drug design [11], recent use has many times demonstrated super-human scientific abilities. As potentially every field can benefit one way or another from these revolutionary tools, it is natural to actively explore their reach within physics laboratories, which are prolific data sources.

The geophysics community has been an early adopter of these approaches [12], with many teams now working on earthquake prediction in the lab [13, 14] and in the field [15–21], and they have recently been invoked in the more general context of plasticity and crackling noise prediction [22, 23].

In practice though, even if a given predictor achieves a reasonable accuracy, it remains unclear how to take advantage of this edge in a practical situation: on the one hand, accuracy assigns every target with the same importance regardless of their relative relevance to a specific problem, and on the other, it doesn't allow direct comparison of similar problems with different targets, i.e. from one publication to another.

Here we propose an operational solution to overcome this challenge, based on the idea that the accuracy in the prediction of a specific quantity should be evaluated in light of how informative it is to the associated underlying question, in our case risk management. First, we introduce a range of scalar targets to be predicted from analogue seismic signals emitted during the experimental deformation of knitted fabric, and train a generic neural network at these predictions. Then we define a risk-based metric balancing the impact of political decisions in an imaginary city living on our system: evacuating the city results in a given social cost, but failing to evacuate might result in casualties. Finally, we train an agent to design a policy by minimizing this penalty over time through reinforcement learning, while having access only to the predictions of the different forecasters. This bias-free, autonomous decision making process allows us to quantitatively compare the various targets and associated parameters, and to give physical meaning to the notion of accuracy within this context. Finally, we discuss the resulting ranking in

terms of past and future time scales, assessing in particular the relevance of the initial forecasting endeavor.

2. Materials and methods

2.1. Experimental set up

In order to gather time series of seismic-like activity, we collect experimental data while mechanically cycling a nylon thread, loose Stockinette knit, which was recently shown [24] to display a rich, seismic-like response. We use a setup similar to that reported in [25]. We perform a tensile test, consisting in varying cyclically the elongation L of the fabric, mesuring the mechanical response during the stretching phase: the force signal and the stitches field displacement. This protocol yields a value of the force every time step of 0.04s, with a cycle duration of 2000s, leading to 50000 points of mechanical response per cycle, and 400 pictures. We collected dedicated data from 20 experiments of approximately 30 cycles each, which corresponds to roughly to 25 millions force points. The raw data for one cycle is shown in Fig. 1.

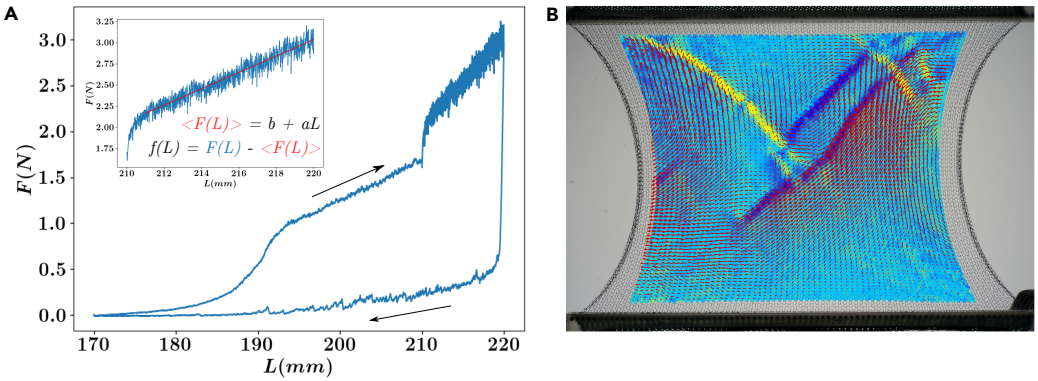

Figure 1: **A**: mechanical response of the fabric for one complete cycle of elongation followed by a compression between L_i and L_f. The knitted fabric becomes entirely 2D around $L = 195mm$. The slow velocity part between 210 and 220 mm is zoomed in the upper left corner, and the affine response is highlighted in red. **B** Picture of the full knitted fabric taken during an experiment, decorated by the non-affine part of the stitch displacement field (see [24] for more details). The color encodes the corresponding vorticity in arbitrary units.

2.2. Data preprocessing

Upon stretching, the mechanical response of the fabric displays large fluctuations around an affine response that is removed by a linear detrending (Fig. 1). The amplitude of these fluctuations is however non-stationary over the cycles, signaling the wear of the fabric over time. In order to build a robust normalization of this signal, we first remove the temporal mean of the data and scale it by its standard deviation, these quantities being different from one cycle to the next. Furthermore, within a given cycle, we observe that the amplitude of the fluctuations increases linearly with the extension. We thus apply a second such normalization, where the statistics are collected over a sliding time window across all cycles, which achieves a signal f with a stationary distribution.

2.3. Event statistics : a seismic analogue system

Upon closer inspection, the normalized fluctuations f exhibit a complex, "stick-slip"-like behavior, with linear loading phases interrupted by abrupt drops, as can be seen in Fig. 2 **A**. These force drops, or "slip-events", were shown in [24] to be concomitant and correlated in amplitude with time-intermittent, spatially extended fault-like plastic events like those shown in Fig. 1 **B** (see Appendix B for morphological aspects of this analogy).

They are thus very reminiscent of the seismic "quakes" that threaten populations, and are as such at the heart of our prediction endeavor. We therefore define the quantity δf at each time step t by:

$$\delta f(t) = \begin{cases} \Delta f & \text{if } t \text{ is the beginning of a drop} \\ 0 & \text{else.} \end{cases} \quad (1)$$

where we characterize each drop by its amplitude Δf (see Fig. 2A). Following this construction, more than 93% of δf 's values are zeros. The distribution of the events Δf displays three different regimes: noisy structureless at low amplitude, a scale-invariant regime that extends over slightly more than 3 decades with an exponent of -1.3, and an exponential cut-off for very large and rare events. This distribution is characteristic of the avalanching dynamics found in seismic activity (see [26] and references within).

This long-tailed statistics suggests defining classes of event severity in a logarithmic scale. As an illustrative example, we will pick either N= 5 classes centered on the decimal magnitudes in the event number count distribution, or N=2 classes with two different thresholds a and b, isolating either the

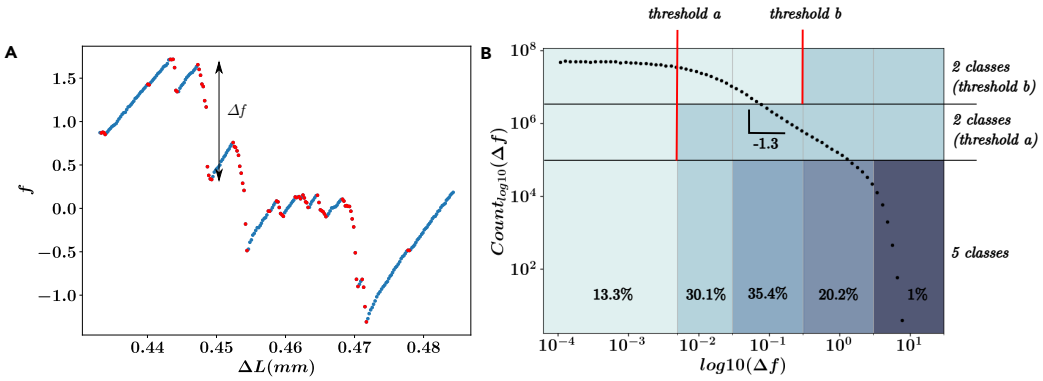

Figure 2: **A**: Zoom on the normalized fluctuation signal displaying "stick" phases in blue, and "slip" phases in red. A scalar event Δf is defined by the (positive) amplitude of a "slip" phase. **B**: Number count distribution function of force drops amplitudes in the normalized force fluctuations. It displays a power law regime with exponent -1.3. The division into five classes is arbitrary and based on the decimal magnitude of the force drops. The thresholds a and b used for two-class learning separate respectively the noise from the events, and "small" from "large" events. The figures correspond to the relative proportion of events in each of the five class.

noise or the large events, as shown in Fig. 2B. Class zero events correspond to noise, extended to a small event class 1. Then medium-sized events (class 2) and large events (class 3) potentially qualify as non-severe and severe respectively, while class 4 corresponds to very rare, catastrophic quakes.

3. Results

3.1. Predicting extreme events throught supervised machine learning

In this section we describe in detail the machine learning procedure we designed in order to predict various notions of danger imminence based on past measurements. We tackle time series forecasting using an artificial neural network whose architecture is detailed below.

Because most of the δf 's are zeros, the NN would only learn to predict zero without a proper rebalancing of the dataset. We deal with this issue by introducing a future horizon window of size τ (in time steps) and predicting the severity of future events within that window. We consider three strategies for aggregating the events in the near future:

- Target 1:
$$T1(t) = \max_{t' \in [t, t+\tau]} \delta f(t') \tag{2}$$
This target focuses only on the largest event in the near future discarding the temporal information,

- Target 2:
$$T2(t) = \Sigma_{t'=t}^{t'=t+\tau} \delta f(t') \tag{3}$$
aggregates the amplitudes of events in the near future,

- Target 3:
$$T3(t) = \Sigma_{t'=t}^{t'=t+\tau} e^{\frac{-(t'-t)}{(\tau/3)}} \delta f(t') \tag{4}$$
confers more weight to more imminent events.

These scalar targets are then mapped to discrete labels for classification, denoted Y(t) in the following, first with two classes separated by two different thresholds, one separating the noise from the events, denoted as a, and the second one separating significant against unsignificant events, denoted as b, and then with 5 classes matching the events statistics discussed earlier.

Target	Class 0	Class 1	Class 2	Class 3	Class 4
δf	94%	2.2%	2.3%	1.42%	0.08%
T1	33.5%	13%	26%	26%	1.5%
T2	33.2%	11%	25%	29%	1.8%
T3	49%	15.8%	23%	12%	0.2%

Table 1: Example of class rebalancing: percentage of the data - raw δf and the 3 targets evaluated for $\tau = 20$ - falling in the 5 classes defined in the text.

We have tested different choices of $\tau = 20, 40, 60$ time steps, which roughly coincide with the average time delay between events from class 1 to 3. The class rebalancing achieved by all targets for $\tau = 20$ is displayed in Table 1. Note that our approach is somewhat hybrid (via the use of τ) with respect to other choices of the literature, where, eg. the prediction task focused on either time to failure regardless of amplitude [13] or amplitude of next event regardless of when it happens [20].

The input of the NN is built by stacking together three 1D arrays:

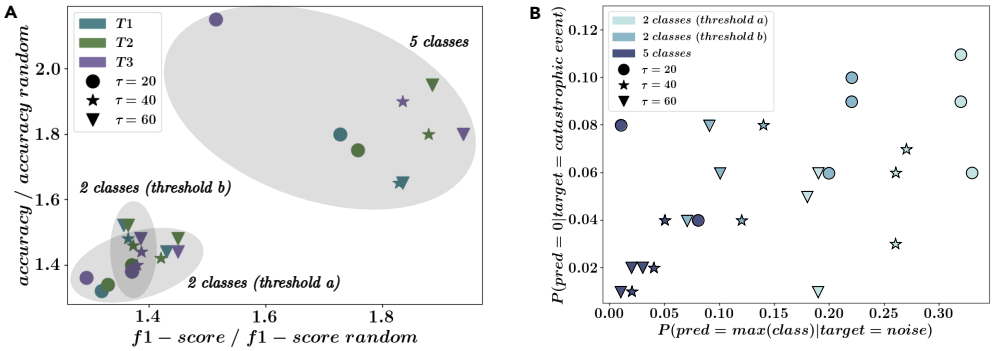

Figure 3: **A**: Accuracy and F1-score for all models with respect to a random prediction. **B**: Learning performance in the false positive (x-axis) and false-negative (y-axis) space between noise and extreme events only for all models.

- $(f(t-n-1), \ldots, f(t-1))$,
- $(\delta f(t-n-1), \ldots, \delta f(t-1))$,
- $(Y(t-n-\tau-1), \ldots, Y(t-\tau-1))$.

Here, n is a past horizon that we set to 256 time steps.

We investigated different NN architectures and found that convolutional-based NNs achieved the best results. We will thus only report the results obtained with a ResNet18 [27] (see Appendix G for details). The whole dataset of 27M sequences was split into 22M of them for training, 2.5M for the validation, and 2.5M for testing. The training sequences are then subsampled in order to enforce equiprobability of the associated labels. The test set is however not subsampled since we want to probe our model's abilities on realistic data.

We would like to be able to compare our results with those found in the literature [13–21], but the choice in machine learning strategy makes this comparison difficult, even counter intuitive. Indeed, the two-class models preferred in the literature present the best accuracy, whereas we find it not very helpful to predict extreme events only. Moreover, for N=5, the accuracy value alone fails to qualify that ability, due in particular to the proportion of events in each class being highly imbalanced in the test set.

We thus also report the F1-score, which is the harmonic mean of recall and precision, two additional and complementary metrics [20] (see Appendix

C for definitions), which more precisely qualifies the ratio of true positives and true negatives for each class. Both these indicators, normalized by the value they would assume with a random prediction, are displayed in Fig. 3A: the NN is indeed able to learn and generalize from the training set, and increasing the number of classes or the future horizon τ typically leads to better performance. The target $T2$, which is simply the sum of the slip amplitudes in the future horizon, also seems to be mildly favored.

This result is confirmed by the direct study of false positive and false negative rates restricted to the two edge classes, noise versus extreme events only, as seen in Fig. 3B: increasing the number of event classes greatly reduces both the risk of predicting low class events when a major event is in fact going to happen, and the risk of overestimating the danger.

Furthermore, although these false positive and negative rates may seem very low (around 1% for the best model in the bottom left corner of Fig. 3B), this is still far from being perfect because extreme events have themselves an even lower occurrence rate (around ten times smaller).

As stated in the introduction, these predictions are not handy to read or use, and a quantitative assessment of their differential quality is still lacking at this point. One way around these difficulties consists in defining exactly what we mean by a useful prediction: within the georisk context, a prediction is useful if it allows policy-makers to make better informed decisions, in order in particular to save more lives while sparing the local economic activity as much as possible. In the next section we thus describe a second layer of (reinforcement) learning meant to design a risk-management policy based on the predictions of this first layer. We may on the one hand be able to compare the different models in terms of a single risk-based metric. On the other, upon inspection of the past predictions of the model, we may be able to improve the decisions with respect to some naive policy by taking advantage of the *dynamics* of the past predictions of event severity.

3.2. Assessment of the results: Game Theoretical Framework

In this section, we introduce KnitCity, a virtual city that is subjected to "quakes" based on the crackling dynamics extracted from the mechanical response of the knitted fabric. KnitCity laboratory keeps track of all past seismic data, and has access to all past predictions of one of the previously designed model.

At each time step, the mayor must decide whether to evacuate the city or not based on these predictions. Such a decision is a trade-off between the risk

of exposing citizens to catastrophic quakes, resulting in a severe human loss, and the risk of evacuating the city for nothing, to which we may associate a "social cost", related to the corresponding economic loss for example. Such a social cost will be denoted λ for each day spent out of the city.

On the other hand, we model the human cost with the following weights corresponding respectively to events from class 0 to 4:

$$\text{damage} = \mu\,[0, 0, 0, 1, 10] \tag{5}$$

where μ is a free parameter. Such a modeling choice reflects the fact that for actual earthquakes, there is a threshold "magnitude" below which human cost vanishes. Also, increasing by an order of magnitude the damages caused by class 4 events with respect to class 3 events reflects the fact that amplitude drops, which embody the events severity, increase exponentially with class labels. Since the ratio μ/λ is the only parameter that drives the trade-off between casualties and time out, λ can be arbitrarily set to 1. There are then two limiting behaviors: if μ goes to infinity, the optimal choice is to always stay out of the city, and to always stay in if μ goes to zero, though μ cannot be estimated *a priori* since it is based on social considerations.

Using Eq. (5) and the event statistics, we may estimate typical values of μ by requiring that a random policy that evacuates the city p percent of the time yields a social cost of the same order of magnitude as the human cost. Values of p around 0.2 for example (see results section below) led to the order of magnitude: $\mu = \mathcal{O}\,(20)$. The influence of μ on the following results is discussed in Appendix E.

A decision policy π can be evaluated by a reward given by the sum over an episode:

$$R = \sum_{t \in episode} s(a(t)) + h(a(t), \delta f(t)) \tag{6}$$

where s (0 or -1) is the social cost determined by action a at time t, and h is the human cost that depends on both the action and the actual event δf:

$$s(a) = \begin{cases} -\lambda & \text{if action: leave} \\ 0 & \text{if action: stay.} \end{cases} \tag{7}$$

$$h(a, \delta f) = \begin{cases} 0 & \text{if action: leave} \\ -\text{damage}(\delta f) & \text{if action: stay.} \end{cases} \tag{8}$$

In the following, we shall also use two more quantities in order to represent policy performances. We define

$$\eta = 1 - \frac{\text{actual casualties following the policy}}{\text{maximum casualties}} \tag{9}$$

the life saving rate averaged over episodes and

$$\kappa = 1 - \frac{\text{number of steps where we evacuate}}{\text{number of steps in the episode}} \tag{10}$$

the population retention rate averaged over episodes: the policy consisting of always staying in the city has coordinates $(0, 1)$ in the (η, κ) plane, while the policy consisting in always evacuating the city has coordinates $(1, 0)$ (see Fig. 4A). An agent that randomly evacuates the city at some rate p should on average expect casualties proportional to $1 - p$ and thus the set of all possible random policies must lie on average on the $y = 1 - x$ straight line.

The optimal policy that consists in evacuating the city only when the damages are larger than the social cost can be evaluated by simply looking into the future: we find $\eta = 1$ and κ also very close to 1 since dangerous events are very scarce. If we moreover add that the evacuation rate is fixed and continuously varying from 0 to 1, then we may define "an optimal frontier" as being the green straight lines in Fig. 4A.

The average reward by time step can also be mapped as $r = r(\kappa, \eta) = -A_1 + \kappa + A_2\eta$ via the previous equations, for two constants A_1 and A_2 that can be computed given the signal statistics and μ. Models can thus be ranked with respect to the average reward per time step; constant rewards are decreasing straight lines in that space, see e.g. Fig. 4A. The relationships between r, R, η, κ, A1 and A2 is discussed in Appendix D.

Using the NN's predictions discussed above, we first manually define a naive policy based only the last available prediction, where we evacuate at every alert beyond the threshold λ, ie. every time an event of class 3 or 4 is predicted:

$$\pi_{\text{naive}} = \begin{cases} \text{leave} & \text{if damage(prediction)} > \lambda \\ \text{stay} & \text{else.} \end{cases} \tag{11}$$

Results of such a naive policy are shown in Fig.4A. The natural question is then whether we can go beyond the naive policy by leveraging the sequence of past NN's predictions through the training of a reinforcement learning agent.

We therefore trained an unsupervised model consisting in a Categorical Deep-Q-Learner [28] belonging to the wide class of Reinforcement Learning (RL) Agents that have achieved remarkable performance in many areas that need the time component and memory to be taken into account : from games to control, auto-pilots, etc. Such a network learns the probability distribution function of the expected rewards for a given action and a current state, and acts by sampling it. Details about the chosen hyperparameters are explained in Appendix H. We trained multiple RL agents that had access to a varying number of past predictions. We set $\mu = 20$ and train the RL agent for the 27 models obtained in the previous section, with either 1, 4 or 16 past predictions made available to the RL agent. We will denote RL(1), RL(4) and RL(16) these models. In terms of the average reward per step, the results in Fig. 4 clearly shows that RL agents improve upon the naive policy. Notice also that some of the models converge to the "always in" policy (especially RL(1) models).

Zooming in on the best performing models with a reward per time step $|r|$ below 0.4, we can return a list of best models. Irrespective of the number of past data fed to the RL agent, the best model is systematically the one that uses $N = 5$ classes with target T3 and a future horizon of 20 or 40.

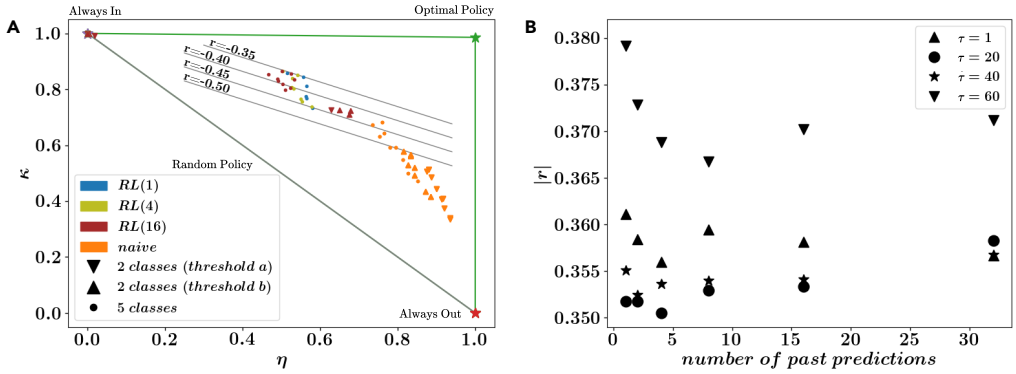

Figure 4: **A**: Ranking of learned policies with an RL agent compared to the naive policy for the 27 different models trained in the previous section. Also shown are the lines of constant reward per time-step. **B**: Ranking of learned policies for the model defined with $N = 5$ classes and target $T3$. The largest impact on the results comes from the size of the future horizon, with τ around 20 being clearly favored. The number of past predictions fed to the RL agent is less sensitive, although we find a best model for $n = 4$ past predictions, and a slight decrease of learning performance if we feed too many of them. Here the reward per step is plotted in absolute value, and best models have the lowest $|r|$ values.

Exploring further our results, we now fix the model (target 3 and $N = 5$), and vary τ in the range $(1, 20, 40, 60)$ and for $(1, 2, 4, 8, 16, 32)$ past predictions made available to the RL agent. The results, displayed in Fig. 4B, show that RL(4) and $\tau = 20$ are the best hyperparameters to use.

4. Discussion

In an effort to gain insights into the immediate future crackling activity in the dynamometric signal of a knitted fabric, we have established a general framework, applicable *de facto* to any time series prediction for which a clear motivation can be rationalized, in which we can on the one hand compare through a common rating the predictive power of the predictor, and on the other explicitly construct a step by step action plan answering the initial motivation by efficiently exploiting the predictions. This development allows in particular to rank the relevance of the various targets with past and future horizons used in neural network predictors for the risk management in a model seismic environment. Our strategy, which consists in balancing instantaneous deleterious events with a time integrated quantity spontaneously produces interesting policies, can be extended to real-world situations when the cost function can be estimated, i.e. geological or climatic hazard. Encouragingly, our analysis clearly shows that these predictors yield some valuable information about the future.

Of course many challenges remain to be tackled within our approach. Extensive optimization of hyperparameters would certainly allow substantial refinement both in the quality of the predictor and that of the decision-making agent: our work only provides a proof of concept of the type of insights that can be obtained within the model environment of KnitCity, and was not fully fine-tuned. Building on our findings, it would be worth further investigating the role of the number of output classes, even going to the regression limit.

Furthermore, our dataset allows for spatially resolved predictions [24] which would of course also be beneficial within the geoseismic context as satellite imaging enables such analysis: this approach is high on our list of future priorities. However, establishing a quantitative correspondence between the space and time scales in our system and those in the geophysical context would be required if we were to export the machinery introduced above to real-world situations, and this preoccupation will be at the heart of our future work on spatially resolved prediction.

Acknowledgements

We thank Sebastien Moulinet for his assistance in controlling the experiment, and Florent Krzakala for numerical equipment. We acknowledge the PaRisArtificial Intelligence Research InstitutE (PRAIRIE) for financial support.

Appendix A. Knitted Fabric and Experimental Protocol

The knits were crafted using a Toyota KS858 single bed knitting machine with nylon mono-filament (Stroft® GTM) of diameter $d = 150\,\mu$m and length 25.4 m. All samples are 83×83 stitches with an average lateral and longitudinal stitch size of, respectively, 3.9 and 2.8 mm. The yarn Young's modulus $E \approx 5.1\,$GPa was measured in a standard tensile test.

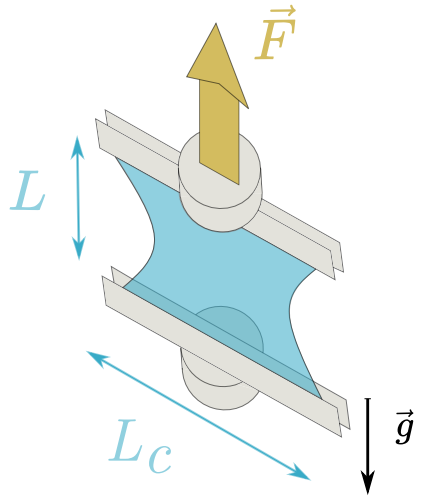

Figure A.5: Color. Experimental setup: a clamped knitted fabric is set into an unixial traction device. We record both the tensile force and images of the fabric upon stretching.

The knit is clamped on its upper and lower rows by means of screws holding each stitch individually, imposing a constant spacing between them along the corresponding rows. Starting from an initial configuration with height $L_i = 170\,$mm and width $L_c = 360\,$mm, we perform the tensile test using an Instron® (model 5965) dynamometer mounted with a 50 N load cell. The elongation L of the fabric between the jaws is varied cyclically between L_i and $L_f = 220\,$mm. The force signal is recorded at high frequency

(25 Hz) during the stretching phase on a shorter elongation range, between $L_m = 210\,\text{mm}$ and L_f, and pictures of the knit are taken every $L_f = 5\,\text{s}$. To approach the quasi-static deformation limit in the interval $[L_m, L_f]$, we impose a constant loading speed and set it at a small value $v = 5\,\mu\text{m/s}$. To reduce the duration of the experiment, we fix $v = 5\,\text{mm/s}$ outside this measurement window.

Appendix B. Event Morphology

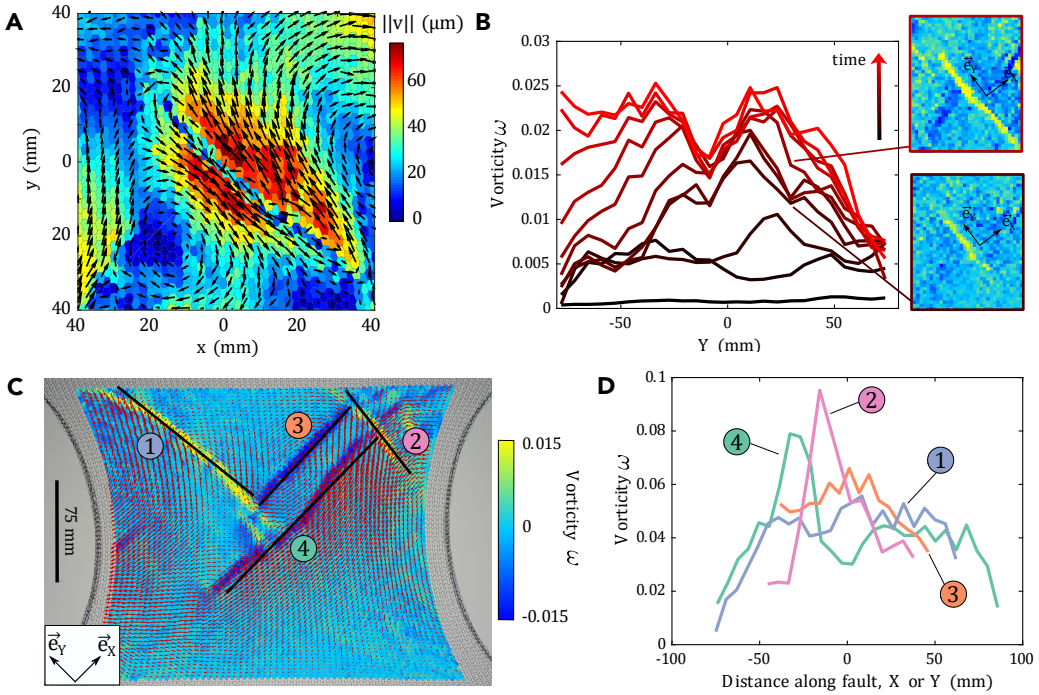

Figure B.6: Color. **A**: Non-affine part of the stitches displacement field surrounding an isolated event. Each stitch is colored by the norm of its displacement (arbitrary scale). **B**: The emergence and propagation of a fault line observed with a high-speed camera. **C**: Picture of the a full knitted fabric with stitches colored by their vorticity values. **D**: Corresponding vorticity profile of each event (1 to 4) highlighted in **C**.

To strengthen our argument that the slip events display a deep similarity with earthquakes and fault slips, we describe in more details the spatial morphology of the shear lines inducing the slip events. In Fig.B.6 **A**, we can clearly distinguish an extended shear on both sides of the fault line, strongly

reminiscent of GPS displacement data obtained across active geophysical faults, such as that shown in [29].

In Fig.B.6 **B**, we measure the displacement of the stitches with a fast camera (Photron APX RS 3000) at different time steps from a reference frame, and compute the corresponding vorticity field for each time step. We compute the position Y of each stitch along the fault direction e_Y. The profile of the fault is then obtained by summing for each position Y the vorticity of the corresponding stitches. Each profile line in the plot is separated by 50 ms, which corresponds to an elongation of the fabric of 37.5 μm as it is loaded at a constant speed of 0.75 mm for this specific experiment. Two snapshots, where each pixel represents a stitch and is colored by its vorticity value, are also shown. We observe that a fault is growing progressively from a point (first snapshot), but then seems to trigger another event as the growth develops from another position. This observation is compatible with propagating faults and avalanching events.

In Fig.B.6 **C**, 4 events, colored by their vorticity values, can be detected in the knitted fabric, and are labeled from 1 to 4. The corresponding vorticity profile of each event is shown in Fig.B.6 **D**. While the less extended events seem to present a peaked profile, extended ones like number 4 show multimodal profiles. This multi-modality is very reminiscent of that measured in geophysical faults, which are usually constituted of several interacting faults, producing distinct displacement maxima, as observed for example in [30].

Appendix C. Supervised Learning Metrics

We recall the definition of standard classification metrics for reference.

$$\text{accuracy} = \frac{\text{number of true predictions}}{\text{total number of elements}} \tag{C.1}$$

$$\text{f1-score} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}} \tag{C.2}$$

$$\text{mean precision} = \frac{1}{\text{number of classes}} \sum_i precision(i) \tag{C.3}$$

$$\text{mean recall} = \frac{1}{\text{number of classes}} \sum_i recall(i) \tag{C.4}$$

$$precision(i) = \frac{\text{number of true predicitons in class i}}{\text{number of elements predicted in class i}} \tag{C.5}$$

$$recall(i) = \frac{\text{number of true predicitons in class i}}{\text{number of elements in class i}} \tag{C.6}$$

Appendix D. Relations between parameters in the Game Theoritical Framework

Our models are all evaluated on the same set of episodes, over which the κ and η metrics are averaged. We can then express the maximum casualties, over this set, as a function of the probabilities of occurence of the events, such as $damage(\delta f) \neq 0 : \ p_{d3} = \mathbb{P}(\delta f \in class3)$ and $p_{d4} = \mathbb{P}(\delta f \in class4)$.

$$\text{maximum casualties} = T * \mu * (d_3 p_{d3} + d_4 p_{d4}) \tag{D.1}$$

with T being the number of time step in an episode, $d_3 = damage(\delta f \in class3)$ and $d_4 = damage(\delta f \in class4)$. Thus, we can write the average reward by episode $< R >$ as a function of η and κ:

$$\begin{aligned} R &= \sum_{t \in episode} s(a(t)) + h(a(t), \delta f(t)) \\ \Rightarrow < R > &= T * \{-\lambda * (1-\kappa) - \mu * (d_3 p_{d3} + d_4 p_{d4})(1-\eta)\} \end{aligned} \tag{D.2}$$

To eliminate the dependency on the number of steps per episode, we define the average reward by step, over all the episodes of the set, by:

$$\begin{aligned} r &= \frac{< R >}{T} \\ &= -1 - \mu * (d_3 * p_{d3} + d_4 * p_{d4}) + \kappa + \mu * (d_3 * p_{d3} + d_4 * p_{d4}) * \eta \end{aligned} \tag{D.3}$$

which is indeed of the form $r = -A1 + \kappa + A2\eta$ where $A_1 = -1 - \mu * (d_3 * p_{d3} + d_4 * p_{d4})$ and $A_2 = \mu * (d_3 * p_{d3} + d_4 * p_{d4})$ being the constants we refer to in the main text. In particular, the relation between r and $< R >$ follows from the definition of η and κ.

By definition, we have

$$\kappa = 1 - \frac{\text{T*p}}{\text{T}} = 1 - p \tag{D.4}$$

To relate η and p, we must first define the probability of staying while an event of class 3 occurs: $\mathbb{P}(action = stay \cap \delta f \in class3)$, and the probability of staying while an event of class 4 occurs : $\mathbb{P}(action = stay \cap \delta f \in class4)$. Then we can write casualties following the policy as

$$\begin{aligned} \text{casualties} = T * \mu * (d_3 * \mathbb{P}(action = stay \cap \delta f \in class3) \\ + d_4 * \mathbb{P}(action = stay \cap \delta f \in class4)) \end{aligned} \tag{D.5}$$

where $\mathbb{P}(action = stay \cap \delta f \in class3)$ and $\mathbb{P}(action = stay \cap \delta f \in class4)$ have a non trivial relation to p. However, in the particular case when p is chosen randomly, $P(action = stay) = 1 - p$ and $P(\delta f \in class3)$ or $P(\delta f \in class4)$ become independent, allowing us to simplify the expressions (D.5) and (9), leading to $\eta = p$ and $\kappa = 1 - \eta$.

Appendix E. Effects of μ

We report on the effect of varying the parameter μ, summarized in Fig. E.7. Points here are trained models grouped by values of μ. This allows to see a smooth evolution from always-in (low μ) to always-out (high μ) policies as expected. Again, this figure emphasizes that there is no unique way to define an optimal policy when predictions are not perfect; instead we get a family of trained decisional agents that are μ dependent. However the point is that the framework we introduced now provides with both quantitative and qualitative tools to explore and rank decision policies, and most importantly it allows to compare predictions of possibly completely different nature.

Appendix F. Data and materials availability

All raw and processed data are available under CC-By licence at https://osf.io/nf847/ in the KnitQuakesForecast Project (DOI 10.17605/OSF.IO/NF847). The source code of the data analysis can be found at https://github.com/adeledouin/ KnitAnalyse_open/, and is plug and play with the above data set.

Appendix G. Machine learning neural net

The Neural Network used in this work is a ResNet18 – inspired by the one coded in

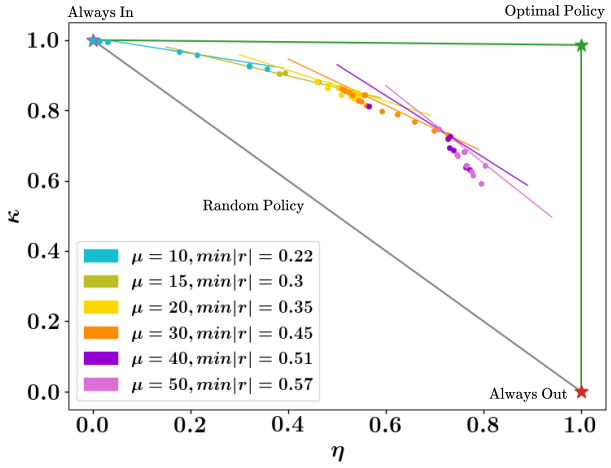

Figure E.7: Color. Typical positions of optimal learned policies in the η, κ space when μ is varied; for models with T_3, $N = 5$ classes and τ in the tuple $(20, 40, 60)$, and different number of past predictions. Colored lines are iso-r associated to each value of μ.

https://github.com/pytorch/vision/ with blocks size = $[64, 128, 256, 512]$, deepths = $[2, 2, 2, 2]$, and ResNetBasickBlocs - but with 1D convolution instead of 3D. The code source can be found at https://github.com/adeledouin/KnitQuakesForecast_open/. We used Adam as an optimizer with an initial learning rate of 0.01, a weight decay of 1.10^{-4} and a learning rate decay of 0.5 every 50 epochs for 300-epoch runs. As criterion we used the cross entropy loss of Pytorch – with the weight of each classes.

Appendix H. Reinforcement learning environment and model

We developed a Gym environment, and used a Categorical Deep-Q-Learner from the prfl package that can be found at https://github.com/pfnet/pfrl/. We use a distributional Q-function with discrete action of 51 atoms, with 2 hidden layers of 64 neurons. As optimizer we use Adam with an 1.10^{-3} rate, and a discounting factor γ of 0.95. We subsampled the test set of the previous Section 3.1.2 in a set of 75 episodes, or "games" of 5000 time steps each (such that on average, two major events of class 4 are present in each game), and validate on unseen data grouped in 50 episodes of 5000 steps. All models are then tested on 180 new unseen episodes of the same number of steps. The code source can be found at

https://github.com/adeledouin/KnitQuakesForecast_open/.

References

[1] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, et al., Mastering chess and shogi by self-play with a general reinforcement learning algorithm, arXiv preprint arXiv:1712.01815 (2017).

[2] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing atari with deep reinforcement learning, arXiv preprint arXiv:1312.5602 (2013).

[3] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al., Grandmaster level in starcraft ii using multi-agent reinforcement learning, Nature 575 (7782) (2019) 350–354.

[4] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).

[5] J. Li, W. Monroe, A. Ritter, M. Galley, J. Gao, D. Jurafsky, Deep reinforcement learning for dialogue generation, arXiv preprint arXiv:1606.01541 (2016).

[6] J. Ruiz-del Solar, P. Loncomilla, Applications of deep learning in robot vision, in: Deep Learning in Computer Vision, CRC Press, 2020, pp. 211–232.

[7] C. J. Shallue, A. Vanderburg, Identifying exoplanets with deep learning: A five-planet resonant chain around kepler-80 and an eighth planet around kepler-90, The Astronomical Journal 155 (2) (2018) 94.

[8] A. Radovic, M. Williams, D. Rousseau, M. Kagan, D. Bonacorsi, A. Himmel, A. Aurisano, K. Terao, T. Wongjirad, Machine learning at the energy and intensity frontiers of particle physics, Nature 560 (7716) (2018) 41–48.

[9] D. Ribli, A. Horváth, Z. Unger, P. Pollner, I. Csabai, Detecting and classifying lesions in mammograms with deep learning, Scientific reports 8 (1) (2018) 1–7.

[10] A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, A. W. Nelson, A. Bridgland, et al., Improved protein structure prediction using potentials from deep learning, Nature 577 (7792) (2020) 706–710.

[11] Y. Jing, Y. Bian, Z. Hu, L. Wang, X.-Q. S. Xie, Deep learning for drug design: an artificial intelligence paradigm for drug discovery in the big data era, The AAPS journal 20 (3) (2018) 1–10.

[12] A. Panakkat, H. Adeli, Neural network models for earthquake magnitude prediction using multiple seismicity indicators, International Journal of Neural Systems 17 (01) (2007) 13–33. doi:10.1142/S0129065707000890. URL https://www.worldscientific.com/doi/abs/10.1142/S0129065707000890

[13] B. Rouet-Leduc, C. Hulbert, N. Lubbers, K. Barros, C. J. Humphreys, P. A. Johnson, Machine Learning Predicts Laboratory Earthquakes, Geophysical Research Letters 44 (18) (2017) 9276–9282. doi:10.1002/2017gl074677.

[14] P. A. Johnson, B. Rouet-Leduc, L. J. Pyrak-Nolte, G. C. Beroza, C. J. Marone, C. Hulbert, A. Howard, P. Singer, D. Gordeev, D. Karaflos, C. J. Levinson, P. Pfeiffer, K. M. Puk, W. Reade, Laboratory earthquake forecasting: A machine learning competition, PNAS 118 (5) (2021) e2011362118. doi:10.1073/pnas.2011362118.

[15] K. Asim, F. Martínez-Álvarez, A. Basit, T. Iqbal, Earthquake magnitude prediction in Hindukush region using machine learning techniques, Natural Hazards 85 (2017) 471–486. doi:10.1007/s11069-016-2579-3.

[16] K. Mohankumar, K. Sangeetha, A study on earthquake prediction using neural network algorithms, International Journal of Computer Sciences and Engineering 6 (10) (2018) 200–204.

[17] F. Corbi, L. Sandri, J. Bedford, F. Funiciello, S. Brizzi, M. Rosenau, S. Lallemand, Machine Learning Can Predict the Timing and Size of Analog Earthquakes, Geophysical Research Letters 46 (3) (2019)

1303–1311. doi:10.1029/2018GL081251.
URL https://onlinelibrary.wiley.com/doi/abs/10.1029/2018GL081251

[18] J. Huang, X. Wang, Y. Zhao, C. Xin, H. Xiang, Large earthquake magnitude prediction in taiwan based on deep learning neural network, Neural Network World 28 (2) (2018) 149–160.

[19] B. Rouet-Leduc, C. Hulbert, I. McBrearty, P. A. Johnson, Probing slow earthquakes with deep learning, Geophysical Research Letters 47 (4), arXiv: 1906.08033 (Feb. 2020). doi:10.1029/2019GL085870.
URL http://arxiv.org/abs/1906.08033

[20] R. Mallouhy, C. A. Jaoude, C. Guyeux, A. Makhoul, Major earthquake event prediction using various machine learning algorithms, in: 2019 ICT-DM, 2019, pp. 1–7, iSSN: 2643-6868. doi:10.1109/ICT-DM47966.2019.9032983.

[21] M. Yousefzadeh, S. A. Hosseini, M. Farnaghi, Spatiotemporally explicit earthquake prediction using deep neural network, Soil Dynamics and Earthquake Engineering 144 (2021) 106663. doi:https://doi.org/10.1016/j.soildyn.2021.106663.
URL https://www.sciencedirect.com/science/article/pii/S0267726121000853

[22] H. Salmenjoki, M. J. Alava, L. Laurson, Machine learning plastic deformation of crystals, Nature communications 9 (1) (2018) 1–7.

[23] S. Papanikolaou, Learning local, quenched disorder in plasticity and other crackling noise phenomena, npj Computational Materials 4 (1) (2018) 1–7.

[24] S. Poincloux, M. Adda-Bedia, F. Lechenault, Crackling Dynamics in the Mechanical Response of Knitted Fabrics, Physical Review Letters 121 (5) (2018) 058002. doi:10.1103/PhysRevLett.121.058002.
URL https://link.aps.org/doi/10.1103/PhysRevLett.121.058002

[25] S. Poincloux, M. Adda-Bedia, F. Lechenault, Geometry and Elasticity of a Knitted Fabric, Physical Review X 8 (2) (2018) 021075. doi:10.1103/PhysRevX.8.021075.
URL https://link.aps.org/doi/10.1103/PhysRevX.8.021075

[26] O. Ramos, E. Altshuler, K. J. Maloy, Avalanche Prediction in a Self-Organized Pile of Beads, Physical Review Letters 102 (7) (2009) 078701. doi:10.1103/PhysRevLett.102.078701.
URL https://link.aps.org/doi/10.1103/PhysRevLett.102.078701

[27] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[28] M. G. Bellemare, W. Dabney, R. Munos, A Distributional Perspective on Reinforcement Learning, arXiv:1707.06887 [cs, stat]ArXiv: 1707.06887 (Jul. 2017).
URL http://arxiv.org/abs/1707.06887

[29] Y. Fialko, M. Simons, D. Agnew, The complete (3-D) surface displacement field in the epicentral area of the 1999 MW7.1 Hector Mine Earthquake, California, from space geodetic observations, Geophysical Research Letters - GEOPHYS RES LETT 28 (2001) 3063–3066. doi:10.1029/2001GL013174.

[30] E. J. M. Willemse, D. D. Pollard, A. Aydin, Three-dimensional analyses of slip distributions on normal fault arrays with consequences for fault scaling, Journal of Structural Geology 18 (1996) 295–309, aDS Bibcode: 1996JSG....18..295W. doi:10.1016/S0191-8141(96)80051-4.