Next Day Wildfire Spread Prediction with Nondimensionalization

DT Litster*
dt@mathematics.byu.edu
Brigham Young University

Nathaniel Driggs *
ndriggs@mathematics.byu.edu
Brigham Young University

Abstract

Predicting wildfire spread is crucial to timely response and allocation of limited resources. Using a novel approach based on units equivariance, we create dimensionless features from a new wildfire spread dataset. We find that naïve attempts at mixing dimensionless and dimensional variables fails to provide the improvements expected from fully nondimensionalized problems. These results imply that for physics applications where key variables remain unobserved, a more principled approach to nondimensionalization may be required. The code for this project can be found at https://github.com/ndriggs/dimensionless-wildfire.

1 Introduction

Wildfires cause billions of dollars of damage annually, destroying natural resources and homes while dumping carbon dioxide into the atmosphere. However, our knowledge of how fires behave is limited. Recent work has begun to include atmospheric details in the fire model, but recording accurate data during a fire can be difficult.

When modeling fire behavior, as with other physical problems, one expects the functions to be *units equivariant*. A function f is said to be units equivariant if changing the units causes an equal change in the output of the function. For example, the function $F(m, a) = ma$ is units equivariant; if one measures mass in grams instead of kilograms, the value of m is multiplied by 1000 and the output of the function F is multiplied by the same scaling factor.

In 2022, Soledad Villar et al [1] described a method for imposing units equivariance in physics related machine learning problems. Classically, one typically reformulates the problem by calculating nondimensional variables from the original variables. For example, taking the trivial task of "learning" Newton's first law, there is only one dimensionless variable $\pi_0 = \frac{F}{ma}$, and we would be trying to learn the function $f(\pi_0) = 1$. Villar et al demonstrated that by restricting the class of predictor functions to those that are units equivariant, significant gains can be made in training error and efficiency. However, nondimensionalization requires foreknowledge of every possible relevant variable and the appropriate dimensions for each of those variables.

By contrast, in ML applications one rarely has data for every relevant dimensional variable. The transition to dimensionless variables becomes difficult or even impossible. However, one might

*Denotes equal contribution

still hope to find modest gains in training error and efficiency by approximating units equivariance. In section 2, we discuss our data and our two approaches to approximate nondimensionalization.

In section 2, we discuss our data, including our two approaches to approximate nondimensionalization. In section 3, we describe our expiremental procedure. In section 4, we summarize our results, and in section 5 we discuss the implications of our experiment.

2 Data

We used the modified Next Day Wildfire Spread dataset (mNDWS) on Kaggle[1] which is based on original Next Day Wildfire Spread dataset (NDWS) [2] and has a forthcoming paper. This dataset came out on September 26th, 2024 and we are the first to our knowledge to use it. The inputs of this dataset, as well as descriptions and their units are shown in Table 1.

When processing our data, we had two distinct preprocessing pipelines: one for our baselines and one for our nondimensionalization. For our baseline experiments, we either scaled our data by the max and min in the training data, or normalized it by the mean and standard deviation of the training data to have mean 0 and unit variance.

To approximate units equivariance we took two approaches. The first was to ignore any variables which we could not make nondimensional, and to nondimensionalize the rest. The dimensional variables used in this case were elevation, CHILI, PDSI, NDVI, previous fire mask, energy release component, specific humidity, wind direction and wind speed, burning index, and min and max temperatures. The temperatures were included through the Boltzmann constant, which converts temperature to energy. Nine dimensionless variables were identified. We call this approach the "drop" approach.

The second approach to nondimensionalization was to nondimensionalize those variables that could be nondimensionalized and include the others with their original dimensions. This amounts to assuming that problematic variables such as precipitation are already dimensionless. Sixteen dimensionless variables were identified. We call this approach the "keep" approach.

Once the dimensions of each variable were identified, we used the Buckingham Pi theorem to identify dimensionless variables that we products of our original variables. This presented an interesting problem; some data channels have zero values. These variables would not be allowed to appear in denominators of our dimensionless variables, as such an operation would result in a NaN. Our data was simple enough that we could simply flip any such fractions, removing all possible divisions by zero.

3 Approach

3.1 Models

We experimented with two different models, both used in the literature. We used a CNN auto encoder, as used by the original NDWS paper [2]. We copied the TensorFlow model in the Kaggle notebook of the first author into PyTorch. We also used an atrous spatial pyramid pooling (ASPP) CNN which was shown in [3] to have improved performance on the NDWS dataset. We coded up the ASPP CNN to match the diagram and description in the paper.

[1]George Hulsey, September 2024 https://www.kaggle.com/datasets/georgehulsey/modified-next-day-wildfire-spread

variable	explanation and units dim?
elevation	m
chili (new)	Continuous Heat-Insolation Load Index 0 (very cool) - 255 (very warm)
impervious (new)	Percent of the pixel covered by developed impervious surface
water cover (new)	The frequency (percentage) with which water was present over some time period
population	persons per square kilometer
fuel classification latent dimensions (new)	the x, y, and z value of an embedding of the fuel classification of that pixel
NDVI	Normalized Difference Vegetation Index, dimensionless, range -1 to 1
pdsi	Palmer Drought Severity Index -15 to 15
pr	Precipitation amount, mm daily total
sph	Specific humidity, mass fraction
th	wind direction, degree
tmmn	min temp, K
tmmx	max temp, K
vs	Wind velocity at 10m, m/s
erc	energy release component BTU/ft^2, 1 BTU = 1,055 J
bi (new)	Burning index, NFDRS fire danger index (dimensionless)
fire masks	0 (no fire) 1 (fire) or -1 (unobserved due to cloud cover or other circumstances)

Table 1: Features and units of modified next day wildfire spread. (new) denotes features that were not included in the original Next Day Wildfire Spread dataset. Knowing which inputs have the possibility of being zero is important for knowing how we can and can't combine units (must avoid division by zero).

3.2 Learning Rate Schedules

We experimented with two different learning rates, both chosen from the literature. The paper that first introduced ASPP CNNs found that a polynomial learning rate policy $\alpha * (1 - \frac{iter}{max\_iter})^{power}$ performed well [4]. In our application we took *iter* to be the epoch, and *max_iter* or max epochs to be 100. We used $power = 0.9$ as they did in the paper.

To try and find a good learning rate schedule that might be better suited for the CNN auto encoder we turned to [5]. We noticed that the sinexp policy did the best for training their CNN3 on CIFAR-10. As defined in the paper, but with adapted notation, the sinexp policy is $(\alpha_{max} - \alpha_{min})\gamma^t |\sin(\frac{\pi t}{2l})| + \alpha_{min}$. We used the same parameter values as the paper, with $\alpha_{min} = 0.00005, \alpha_{max} = 0.006, \gamma = 0.99994$, and $l = 2000$.

3.3 Loss Function

We adapted the Tversky index for our loss function as used in [3], with one slight modification. We defined it as

$$\mathcal{L}_\theta = 1 - \frac{\text{TP}}{\text{TP} + \alpha\text{FN} + \beta\text{FP}}$$

with our modification from the paper being the addition of the β to better match the definition of Tversky index in other places. We used the values of $\alpha = 0.7, \beta = 0.3$ as in the paper. These values weight false positives less than false negatives. This is intentional since failing to predict fire spread could have worse consequences than being over cautious and predicting fire spread where there was none. We do not actually calculate the true and false positives and negatives which would require rounding, but rather use the output of the final sigmoid layer, $\hat{y}$ in the following way. Let y be the ground truth batch labels of 0 (no fire) and 1 (fire).

$$\text{TP} = \sum y \odot \hat{y} \qquad \text{FN} = \sum y \odot (1 - \hat{y}) \qquad \text{FP} = \sum (1 - y) \odot \hat{y}$$

Where $\odot$ is the tensor element-wise product and the summations sum over all elements.

3.4 Training

We trained and tested using the automated PyTorch Lightning Trainer. The training dataset has 4,848 samples, validation has 756 samples, and test has 1,460 samples. Since the test set has almost double the samples of validation, we switched their roles in our usage. We checkpointed the model that had the best F1 score on the test set during training, and tested its generalization on the validation set. The numbers reported in Table 2 are the out-of-sample scores on what the dataset author labeled as the validation set. Each experiment took less than 10 minutes to train.

4 Results

The best value for all four metrics was achieved with a polynomial learning rate schedule and batch size of 32. With a relatively small dataset, it makes sense that a smaller batch size would be best. Seeing that cyclic learning rate policies are not widely used, it also makes sense that a more typical monotonically decreasing learning rate schedule won out. Given that the wildfire spread predicition figures from the ASPP CNN in [3] looked much more impressive than those of the CNN AE in [2],

data	model	lr schedule	batch size	test loss	test F1	test precision	test recall
normalized	ASPP CNN	polynomial	32	**0.7741**	0.1920	0.1201	0.6102
normalized	ASPP CNN	polynomial	64	0.9223	0.1014	0.0555	0.6520
normalized	ASPP CNN	sinexp	32	0.9726	0.0196	0.0100	0.7309
normalized	ASPP CNN	sinexp	64	0.9728	0.0205	0.0106	0.4601
normalized	CNN AE	polynomial	32	0.8227	**0.2167**	0.1649	0.3801
normalized	CNN AE	polynomial	64	0.9666	0.0198	0.0101	0.9000
normalized	CNN AE	sinexp	32	0.9731	0.0167	0.0085	0.7693
normalized	CNN AE	sinexp	64	0.9743	0.0	0.0	0.0
scaled	ASPP CNN	polynomial	32	0.8521	0.1611	**0.2432**	0.1381
scaled	ASPP CNN	polynomial	64	0.9061	0.1048	0.0572	0.7916
scaled	ASPP CNN	sinexp	32	0.9714	0.0263	0.0136	0.8213
scaled	ASPP CNN	sinexp	64	0.9733	0.0033	0.0025	0.0059
scaled	CNN AE	polynomial	32	0.9706	0.0178	0.0091	0.9935
scaled	CNN AE	polynomial	64	0.9239	0.0581	0.0304	0.8264
scaled	CNN AE	sinexp	32	0.9738	0.0029	0.0170	0.0017
scaled	CNN AE	sinexp	64	0.9740	0.0007	0.0232	0.0004
keep	ASPP CNN	polynomial	32	0.9999	0.0	0.0	0.0
keep	ASPP CNN	polynomial	64	0.9997	0.0	0.0	0.0
keep	ASPP CNN	sinexp	32	0.9729	0.0181	0.0092	0.8981
keep	ASPP CNN	sinexp	64	0.9724	0.0198	0.0101	0.8367
keep	CNN AE	polynomial	32	0.9976	0.0024	0.1865	0.0012
keep	CNN AE	polynomial	64	0.9924	0.0094	0.2479	0.0050
keep	CNN AE	sinexp	32	0.9747	0.0087	0.0072	0.02396
keep	CNN AE	sinexp	64	0.9731	0.0170	0.0086	0.9771
drop	ASPP CNN	polynomial	64	0.9999	0.0	0.0	0.0
drop	ASPP CNN	sinexp	32	0.9736	0.0170	0.0087	0.7453
drop	ASPP CNN	sinexp	64	0.9728	0.0186	0.0095	0.8442
drop	CNN AE	polynomial	32	0.9720	0.0213	0.0396	0.0519
drop	CNN AE	polynomial	64	0.9553	0.0315	0.0177	0.3629
drop	CNN AE	sinexp	32	0.9731	0.0168	0.0085	**1.0**
drop	CNN AE	sinexp	64	0.9736	0.0162	0.0090	0.1385

Table 2: Results for baseline and nondimensionalization experiments. Experiments with batch size 128 were also performed, but have been ommited since most of them did worse than their counterparts with smaller batch sizes.

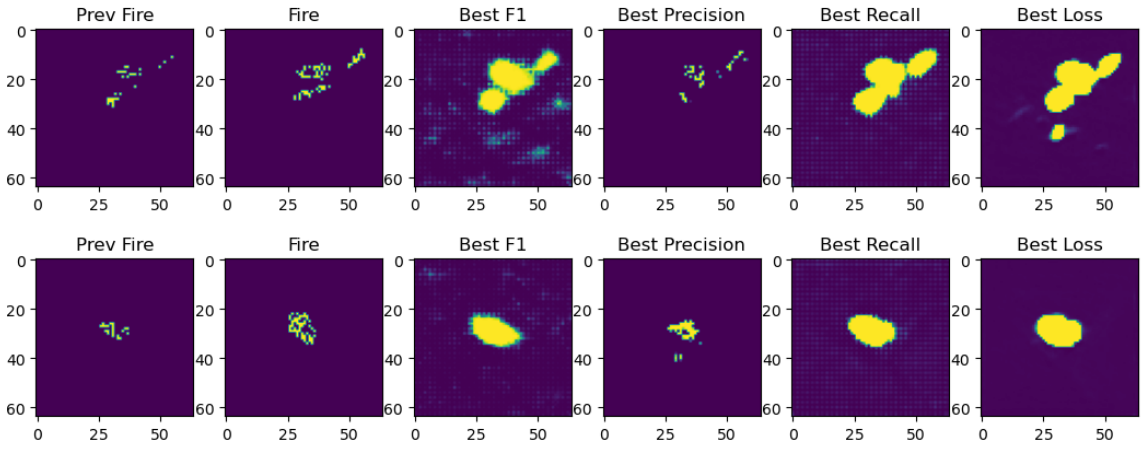

Figure 1: Previous fire mask, next day fire mask, and the predicted next day fire of each of the baseline models that scored best in each of the metrics in Table 2 on two different instances from what we used as the test set. Clearly we picked the wrong loss function and are discounting false positives too much, but unfortunately we didn't have time to iterate. We should have used the exact same loss function used in [3] and left out the β.

we were surprised to see the CNN AE achieve the highest F1 score. Seeing these results we realize that our experimental budget would have perhaps been better spent experimenting with different learning rate policies rather than batch sizes.

Also surprising was that attempts at mixing nondimensionalization with traditional methods made the models worse. The F1 scores and loss were significantly worse than the traditional methods. The sinexp learning rate schedule helped both the drop and keep methods of nondimensionalization with recall, allowing one model perfect recall. It is possible that the restricted class of units equivariant functions allowed for better recall at the cost of precision.

5 Conclusion

Naïve methods for partial nondimensionalization of the wildfire dataset proved unfruitful. It is possible that with only limited data, the class of functions which are units equivariant to some subset of the data is too restrictive. If this were the case, future work could include learning a representation for unobserved variables that would allow for total nondimensionalization. One could attempt to learn what nondimensional variables went unobserved that help your dimensional variables play together, and could even learn to estimate the value of such latent variables.

Our results should be considered preliminary, however; the results could change if one prioritized precision over recall, for example. It is possible that using different topologies or a larger dataset might mitigate some of the losses incurred from nondimensionalization, and may even lead to gains in generalization error without the need for more sophisticated nondimensionalization techniques.

Acknowledgements

Special thanks to Rick Johnson in the Office of Research Computing for helping me to realize my frustrating bug was due to my virtual environment and helping me to create a new one.

References

[1] S. Villar, W. Yao, D. W. Hogg, B. Blum-Smith, and B. Dumitrascu, "Dimensionless machine learning: Imposing exact units equivariance," 2022.

[2] F. Huot, R. L. Hu, N. Goyal, T. Sankar, M. Ihme, and Y.-F. Chen, "Next day wildfire spread: A machine learning dataset to predict wildfire spreading from remote-sensing data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, p. 1–13, 2022.

[3] M. Marjani, M. Mahdianpari, S. Ali Ahmadi, E. Hemmati, F. Mohammadimanesh, and M. Saadi Mesgari, "Application of explainable artificial intelligence in predicting wildfire spread: An aspp-enabled cnn approach," *IEEE Geoscience and Remote Sensing Letters*, vol. 21, pp. 1–5, 2024.

[4] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," 2017.

[5] Y. Wu, L. Liu, J. Bae, K.-H. Chow, A. Iyengar, C. Pu, W. Wei, L. Yu, and Q. Zhang, "Demystifying learning rate policies for high accuracy training of deep neural networks," 2019.