Innovative Featurization and Modeling for Solar Flare Prediction

by

V. R. Deshmukh

B.Tech., College of Engineering Pune, 2011

M.S., University of California Santa Barbara, 2013

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

2022

Committee Members:

Elizabeth Bradley, Chair

James Meiss

Thomas Berger

Claire Monteleoni

Qin Lv

Deshmukh, V. R. (Ph.D., Computer Science)

Innovative Featurization and Modeling for Solar Flare Prediction

Thesis directed by Prof. Elizabeth Bradley

Solar flares and the associated coronal mass ejections are spontaneous high energy bursts of plasma from the Sun. Since the interaction of this high energy plasma with the earth's atmosphere can significantly impact human activity (radiation hazards, economic loss from infrastructure damage, global navigation satellite system interference, etc.), accurately forecasting solar flares is of vital importance. These flares have been known to be associated with sunspots that have complex magnetic field structures called *active regions*. In recent years, the use of deep learning-based methods for solar flare prediction has gained significant traction. Much of this work is classification-based — every input magnetogram image is labeled as flaring or non-flaring based on whether a solar flare of a certain magnitude or higher occurred in the vicinity of the active region in the next k hours. Since solar eruptions are rare events, a highly imbalanced dataset is produced under such a labeling scheme, that poses problems for designing and evaluating machine learning models. Typically, images of photospheric magnetic field observations (called magnetograms) of active regions are either used as input data for convolutional neural network methods which automatically extract features, or are pre-processed to extract physics-based features that are used to train machine learning models such as support vector machines (SVM), multi-layer perceptrons (MLP), extremely randomized trees (ERT), etc. In this thesis, I explore novel directions in both approaches — feature engineering-based methods for training traditional machine learning models (e.g. MLP) and CNN-based deep learning prediction methods. First, I propose a new method to extract features from magnetograms using topological data analysis as a substitute for the standard physics-based features used predominantly by existing feature-based models. Through rigorous machine learning methodologies such as hyperparameter tuning, I show that these abstract shape-based features extracted do just as well as the carefully designed physics-based features widely used in this field.

Second, as part of the same study, I show that increasing the complexity of the machine learning model does not guarantee an increase in the prediction performance — a simple logistic regression model in fact performs slightly better than the complex long short-term memory (LSTM) model. I also study the model complexity further through dimensionality reduction using principal component analysis (PCA) and assess the effects upon model performance. Lastly, I propose a hybrid ML architecture that combines the forecasting power of a feature-based model (ERT) with that of an image-based prediction model (CNN), and show that this architecture reduces false positives in prediction — an important issue in this highly data-imbalanced forecasting problem.

Dedication

This is for you, *Aaji*. You left me halfway through this journey, but your blessings and love got me to the finish line.

Acknowledgements

Liz, I am glad of the day I randomly searched for computer scientists working on Chaos Theory, and found you! You are more than anything a graduate student can ask for in an advisor. Not only have you been an excellent academic advisor, providing me with countless research opportunities, but you have have been a strong pillar of support during some of the toughest times in my personal life. I thank you with all my heart.

My dear wife, Gandhali, for being a constant source of support throughout my Ph.D. Not only did you encourage me to come back to the wonderful world of academia, you helped me fight my imposter syndrome to reach the end. Our shared Ph.D. journey in beautiful Colorado has been the best time of my life. Thank you, Dr. Kogekar.

***Aai, Papa*, Shamika**, and all of my family who have always loved me unconditionally. Thank you for molding me into the person I am today.

James Meiss, **Thomas Berger** and **Natasha Flyer**, for some fantastic collaborations that helped shaped this thesis. Thank you for patiently indulging my amateur knowledge of dynamical systems, solar physics and applied mathematics.

Claire Monteleoni and **Qin Lv** for providing some great insights and critical feedback that helped improve my work.

Joshua Garland, for being my "academic elder brother", a wonderful collaborator and a constant source of inspiration.

My labmates **Golnar Gharooni Fard**, **Samantha Molnar**, **Shruthi Sukumar** and **Jessica Finnochiaro** for their invaluable friendship in these wonderful five years.

Contents

Chapter

Tables

Table

Figures

Figure

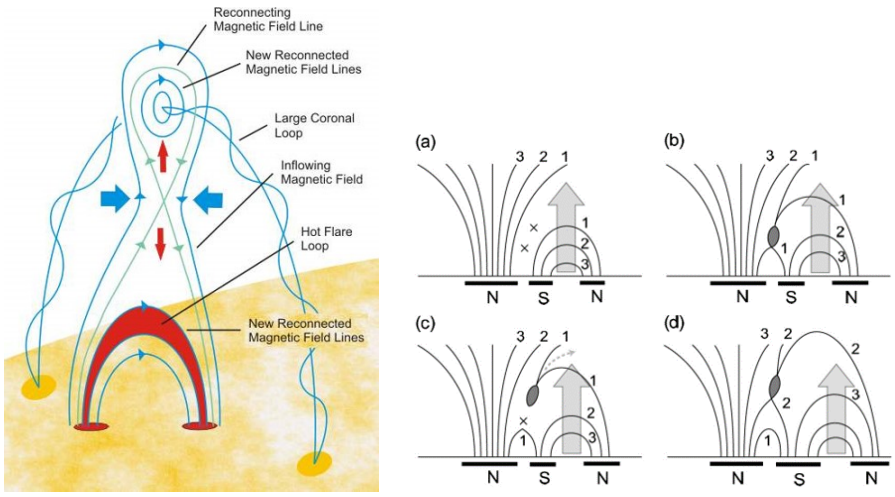
Chapter 1

Introduction

Solar flares are sudden flashes of brightening observed on the surface of the Sun — a result of sudden acceleration of charged solar particles to relativistic speeds. This acceleration can eventually lead to an ejection of high energy particles from the solar corona which travels outwards into space, called *coronal mass ejections (CME)*. Solar flares are formed as a result of the accelerated particles emitting radiation in a wide spectrum of electromagnetic waves such as radio, visible light, X-rays, and gamma rays. The matter ejected in a CME is released into the solar wind and travels out into the solar system attaining a speed ranging from 20 to 3200 km/s. Solar flares and CMEs can have a significant impact on a range of human activity. CMEs can lead to electrostatic discharge within spacecraft electronics – this may have been a cause for the communications blackout on the Galaxy 15 communications satellite [67]. The work by Odenwald et al. [77] estimates a cost of 70 billion dollars for lost revenue and satellite replacement under the impact of a Carrington event-calibre superstorm (one of the most extreme space weather events in history). A burst of protons through the human body can cause biological damage, endangering lives of astronauts in interplanetary travel [51]. CMEs can cause severe additional damage on interacting with the earth's atmosphere — magnetic storms triggered by CMEs can lead to changes in the total electron count in the ionosphere causing GPS systems to lose accuracy [40]. The geomagnetic storms caused in the earth's magnetosphere by CME shock waves induce currents in long transmission lines. Such induced currents can cause transformer coils to heat up and destruct, even leading to a chain reaction. For example, the great geomagnetic storm of March 13, 1989 closed down the entire

Hydro Quebec system [7]. A risk report by Lloyd's has estimated that a power outage from a Carrington-level event could impact a population of 20-40 million in North America alone, with outage durations ranging from 16 days to 1-2 years, and an economic cost of 0.6 to 2.6 trillion dollars [97].

An early warning system of such impending flares and their accompanying CMEs can significantly mitigate such critical hazards. An accurate forecast can enable satellite operators and power grid operators to shut down equipment in advance, astronauts to take shelter in radiation-safe zones, and navigators to move to backup systems in conditions of GPS failure. [96] estimate a savings of 450 million dollars in economic cost by forecasting geomagnetic storms. Considering the potential economic cost and danger to human life, it is important to accurately model and forecast the solar flare phenomenon.

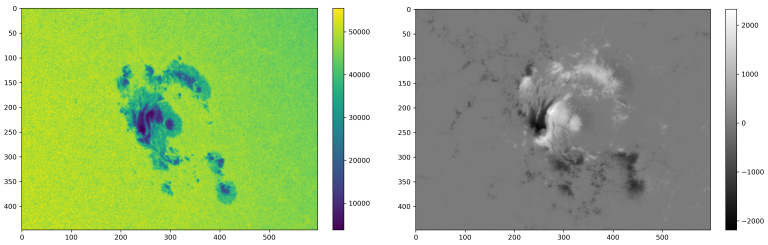


(a) (b)

Figure 1.1: (a) Standard Model for a solar flare showing magnetic reconnections of the arcing loops. Source: *Introduction to Solar Flares* – Gordon Holman, LSSP, NASA Goddard Space Flight Center, (b) Reconnection in an emergent magnetic flux (*Takasaki et al., The Astrophysical Journal, 2004*)

Solar flares (and possibly resulting CMEs) are known to be caused by *magnetic reconnections* – a sudden rearrangement of the photospheric magnetic field structure where proximate opposite fields lines approaching each other rearrange themselves to form new field lines moving away from

each other. Such magnetic reconnections are known to occur more frequently in areas with a complex magnetic field structure known as sunspot *active regions*. These active regions are formed when the Sun's magnetic field forms arc-like structures perpendicular to the surface. As seen in Fig. 1.1, these arcs have "positive footpoints" where the magnetic field lines are directed outwards from the surface, which arch over and go back into the "negative footpoints" on the surface. In such a magnetic field arrangement, magnetic reconnections can spontaneously occur when directionally opposite magnetic field lines approach each other, causing tight loops to be ejected in the vertical direction. Such reconnections may occur due to the magnetic twisting of an arc (Fig. 1.1(a)), or across different arcs in cases of newly emergent flux (Fig. 1.1(b)). Both of these scenarios represent a relatively complex magnetic field arrangement.



(a) (b)

Figure 1.2: (a) White-light intensity observations or continuum images (measured in $\frac{W}{m^2}$) and (b) photospheric line-of-sight images of an active region. The dark spots in the continuum image indicates that the sunspot is cooler than the surrounding region. The photospheric magnetic field (measured in Gauss) reveals the structure of the active region in more detail – the line-of-sight field indicates that the magnetic field is more vertically aligned (upward or downward).

Fig. 1.2 shows the top-view structure of an active region as observed in white light and from a line-of-sight (LOS) photospheric magnetic field perspective recorded by the Helioseismic and Magnetic Imager (HMI) onboard the Solar Dynamic Observatory (SDO) mission. The white light observations show the darker cooler structures in the active region against the background of higher intensity. The vertical field line arrangement in an active region restricts the movement of hot plasma from the interior of the Sun to the surface, causing these regions to be cooler and darker than their neighboring regions (hence the term, *sunspots*). Of the detailed 3D magnetic field

structures shown in Fig. 1.1, the HMI can record the photospheric field (at the footpoints), giving us 2D magnetic field images, called *magnetograms*. The structure on these plots reveals interesting structures in the active region — some cooler spots in white have a high line-of-sight (along the HMI instrument) magnetic field coming out of plane (positive polarity) while other cooler, darker spots have a high LOS magnetic field going into the plane (negative polarity).

The 2D images of the photospheric magnetic field observations provide important clues as to the complexity of the 3D magnetic field arching over into the chromosphere and the corona. The more complex and intertwined the photospheric field structure is, the more complicated the full 3D magnetic field, and higher the chance of a magnetic reconnection and a solar eruption.

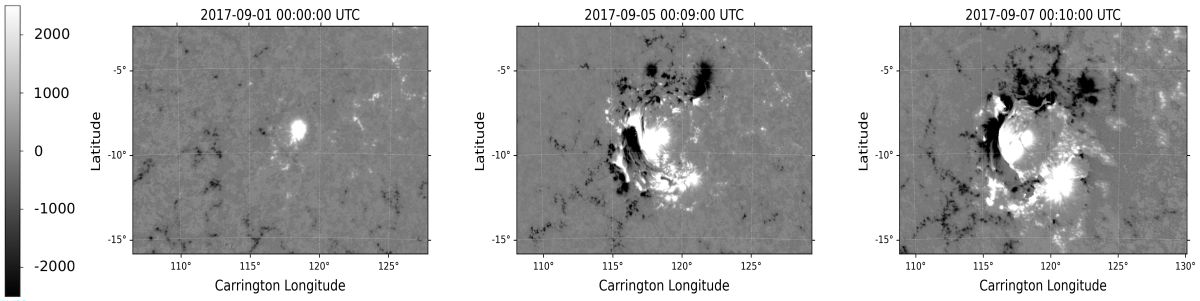


Figure 1.3: A series of HMI magnetograms of sunspot #AR 12673, which produced multiple large eruptions as it crossed the disk of the Sun in September 2017: (a) at 0000 UT on 9/1, (b) at 0900 UT on 9/5, roughly 24 hours before this sunspot produced an X-class solar flare, and (c) at 1000 UT on 9/7, around the time of an M-class flare.

Fig. 1.3 shows a series of line-of-sight magnetograms of an active region before and during an eruptive period. In panel (a), the active region is newly emerged and is concentrated into two relatively compact, positive (white) and black (negative) centers—a "bipolar" configuration. Such "simple" configurations store little free energy and are rarely associated with eruptions. However, as more magnetic flux emerges and the active region evolves under the influence of the plasma flows in the photosphere, it is stretched, rotated, and sheared into the complex shape shown in panel (b). While in this complex configuration, the active region produced a strong flare that had major impacts on Earth-based radio reception. The further development shown in panel (c), later in this

sunspot's series of flaring events, is characterized by intense "polarity mixing," with positive and negative field in close proximity in highly sheared and stretched shapes. From these figures, it is clear that the shape of the photospheric magnetic field is an important indicator of a potential solar eruption. This is a major point of leverage for the advances described later in this thesis.

Considering the correlation between the behavior of the magnetic field and the solar flare occurrence, it is no surprise that the existing operational space weather forecasters classify sunspots based on their photospheric magnetic field structure, and determine their flaring probability based on the historical flaring rate of that class. Over the past couple of decades, there has been a significant research effort towards developing statistical and machine learning models for this task. Recently, there has also been a burst of work on more complex deep-learning methods to further automate this task. These data-driven methods predominantly use the photospheric magnetic field snapshots to train the models. They associate with each magnetogram a probability of a flare occurrence in an upcoming time window, or alternatively, a categorical forecast for a flare occurrence in the upcoming time window. These models (discussed in Chapter 2) either extract properties based on the physics of the magnetogram (total flux, current helicity, etc.) and use them as *features* to train standard machine learning models (SVM, MLP, etc.), or train on the magnetogram images directly using more recent deep learning models such as convolutional neural networks.

While these methods have achieved moderate success, certain limitations prevent them from doing significantly better than expert forecasters. One important challenge is the rare-event nature of the problem, since major solar eruptions occur rather infrequently. Posed as a two-class machine learning classification problem (flare versus no-flare), the resulting dataset is highly imbalanced: for each flaring sample, there are approximately 79 non-flaring samples. Such a high dataset imbalance leads to challenges in model development and evaluation.

In this thesis, my objective is to expand on the ML-based flare forecasting research by applying new feature engineering techniques and deep learning models for improving the solar flare forecasting accuracy. To be concise, I will address the following questions:

- Can the shape of the active region magnetic field be leveraged for improving ML-based solar eruption forecasting?
- Does the complexity of the modeling technique have an impact on the prediction performance for this highly-imbalanced dataset problem?
- Can the high dataset imbalance problem be addressed using hybrid architecture design and tuning on better metrics?

The thesis is outlined as follows. Chapter 2 describes the current effort in data-driven flare forecasting approaches and their limitations, setting the stage for the novel improvements described in this thesis. Chapter 3 addresses the first problem posed in the list above — quantifying the shape of the magnetogram and utilizing it for flare forecasting. In this chapter, I will introduce algorithms from topological data analysis to extract shape-based active region features for training ML models. Chapter 4 is focused on answering the second question: comparing the performance of various traditional machine learning and deep learning models for a 24-hour flare prediction task. This chapter outlines the design, optimization and evaluation processes for four different ML/DL models, trained using the standard physics-based features, the novel topological features, and combinations thereof. As an additional comparison point, these feature-based models are also compared with a CNN model that is trained directly on images. Chapter 5 explores novel architectures that combine the forecasting power of a CNN-based model (trained on magnetogram images) with an extremely randomized trees (ERT) model (trained on numerical features) to reduce the false positives in a 12-hour flare prediction task. Chapter 6 concludes the thesis, and offers some thoughts about future work in this area.

Chapter 2

A Survey of Data-Driven Flare Prediction Methods

In operational space weather forecasting offices, human forecasters currently use the McIntosh [71] or Hale [42, 53] classification systems to give each active region an alphabetical designation reflecting its structure. Each category of active region has a statistical 24-hour eruption probability derived from historical records [24]. This leads, after additional adjustments for factors such as rate of flux emergence, to the 24-hour eruption probability for a particular active region. Recent statistical methods extend and formalize this approach by using historical flaring rates, together with a Poisson process hypothesis, to develop more-complicated models. For example, [36] uses the McIntosh classification to determine the probabilities of C-, M-, or X-class flares and [100] uses a power-law distribution of the flare magnitudes to determine an empirical eruption probability. However, these statistical methods have not been used in operational forecasting, primarily because they do not show greater predictive capability than the historical forecasts that use look-up tables [57, 58].

In the past decade, the large increase in magnetogram data afforded by space missions and advances in data access have shifted the forefront of flare-prediction research from empirical modeling methods to "data analytic" approaches such as machine learning. [15] summarizes the state of the art in machine learning approaches to space weather applications. In ML-based prediction applications, characteristic "features" of the photospheric magnetic field, sometimes combined with features seen in simultaneous Extreme UltraViolet (EUV) images of the solar corona, are used in a statistical sense to "train" a computational model to predict the probability of an eruption within a

given time period (usually 24 hours). One example is a support vector machine (SVM) architecture to perform a binary classification of magnetograms as flaring or non-flaring [9, 11, 74, 103, 105]. [74] applied decision trees and clustering to the same task. A variety of other ML algorithms, such as Bayesian networks [104], radial basis model networks [23], logistic regression [105], LASSO regression [14, 46], and random forests or Extremely Randomized Trees (ERTs) [14, 74] have also been implemented for solar flare prediction with some degree of success. In the new age of deep learning, neural networks have resurfaced as a very powerful and popular tool for going beyond simple direct binary classification on the input space by instead learning complex nonlinear relationships among their inputs. Multilayer Perceptrons (MLPs), the simplest form of neural networks, have also been used to great advantage for solar flare prediction [29, 30, 75]. Long short-term memory (LSTM) models — neural networks that learn important patterns from a temporal sequence of data — have been used in some approaches [20, 92]. [8] use Fuzzy C-Means—an unsupervised machine learning method—in combination with some of the aforementioned supervised methods for solar flare prediction. Approaches such as [41] and [50], while not machine-learning approaches themselves, provide statistical tools for evaluating the engineered features in terms of their potential advantage in machine learning models.

A key challenge is to choose the best features—the individual measurable properties or characteristics of the phenomena that can be extracted from the raw data and then used by the machine-learning method. To date, the majority of features used in ML-based flare prediction methods have been predominantly physics-based.[1] While physics-based features are important, an ML model exclusively trained on them might miss crucial information available through features extracted by other means—for example, quantification of the shape of the active region. The first part of my thesis explores the use of methods that aim to quantify the shape complexity of a 2D magnetogram using tools from computational topology and computational geometry. The motivation for this is not only because shapes on a magnetogram are what human forecasters use in their classifications,

[1] Some recent exceptions to this are features extracted using convolutional filters by [46], features learnt from an autoencoder by [20], and spatial features based on Ripley's index [83]—a functional summary of the density of a point cloud at various scales —by [93].

but also because these shapes have fundamental, meaningful connections to the physical causes of an eruption.

Such a wide variety of machine learning models applied to this problem also raises an important question: which is the best one? While there has been a performance comparison done across a subset of these models [35, 74], a systematic comparison in terms of both model performance *and complexity* is missing. This will be the second major focus of my thesis, where I investigate the solar flare prediction performance of models which vary in complexity by orders of magnitude (from logistic regression to LSTMs). A major component of this study involves an automated hyperparameter tuning approach for a fair comparison across models, similar to that done in [22], but with a more sophisticated sampling mechanism. An alternative approach to modifying the model complexity without significantly changing its architecture is by altering the feature-set representation. In this part of my thesis, I also investigate dimensionality reduction of different feature-sets using principal component analysis (PCA) and using the PCA-reduced features to train the different ML models.

In recent years, the data-driven flare forecasting community has turned its attention to deep learning methods that automatically extract important features from raw image data that are relevant for flare-based classification. Examples that fall into this category are [1, 44, 61, 78, 107, 108], all of which use convolutional neural network (CNN) models. The work by [20] uses an autoencoder model to perform an unsupervised feature extraction from image data — features that are then used to train a recurrent neural network (RNN) architecture. An important advantage of these types of models is that they can completely avoid the need for feature engineering of raw image data. This is useful, especially when domain knowledge is not available. However, the features from these architectures are extracted from a sequence of convolution operations on neighboring pixels and do not necessarily represent a comprehensive set of information that is sufficient for completely describing the raw data. An important problem that remains unexplored in the field of solar flare prediction, to the best of my knowledge, is combining the forecasting power of the CNN-extracted features with engineered features to improve flare prediction accuracy. This is the final major

component of my thesis, where I develop a hybrid deep learning model (a standard CNN model and an ERT model) to leverage the power of both approaches.

There is also a class of flare forecasting methods that attempt to completely model the full 3D fields using the measured photospheric field as a given boundary condition. The 2D magnetogram images provide only the boundary conditions of such physics: the magnetic reconnection that triggers eruptions takes place well above the region sampled by a magnetogram, in the upper atmosphere of the sun. The simplest way to do this is to extrapolate the surface field into the corona assuming zero current, using the potential field solution to the Laplace equation [4, 5, 99]. However, potential fields cannot store energy—they are by definition the lowest energy state of the field and thus cannot model the build-up of energy leading to a CME. More-sophisticated strategies such as Non-Linear Force-Free Field (NLFFF) extrapolations [3, 26, 80, 101], have produced accurate reproductions of active region coronal loop structures and, in some special cases, non-potential energy storage sites above photospheric polarity reversal regions. These NLFFF models are presently the most promising avenue of physics-based coronal magnetic field and eruption modeling [88], but they have a large number of free parameters that require extensive manual tuning. These degrees of freedom reflect the non-uniqueness of the boundary value problem: the photospheric boundary conditions are not sufficient to uniquely determine the field [28, 72, 87]. Hence, the operational forecasting utility of these methods appears limited at present. An alternate approach models the connectivity between opposite polarity magnetic patches in the photosphere to identify significant structures such as magnetic "nulls" and "separatrix surfaces." For example, the Magnetic Charge Topology (MCT) metric is used to characterize eruption potential by [5]. The work by [66] reviews the application of "field-line topology" to infer connections and Tarr *et al.* [94, 95] apply these methods to analyze the eruption potential of active regions. While the structures extracted by these sophisticated methods are meaningful in the context of solar eruptions, their computational complexity presently limits their operational application[2] .

[2] Note that Barnes and Leka have gone on to apply their MCT metric to an operations-ready flare prediction algorithm called Discriminate Analysis Flare Forecasting System (DAFFS) [59].

Chapter 3

Featurizing Active Region Magnetograms

The previous two chapters should by now have impressed the reader of the importance of magnetic field data of active regions in providing useful information for solar flare prediction, and the ability of machine-learning methods to leverage that information. For that to happen, however, the data needs to be preprocessed first. One of the important considerations that determines the ML modeling strategy is the input representation of data. Traditional machine learning approaches such as logistic regression, support vector machines, extremely randomized trees, etc. typically work well with a more compact set of numerical features, rather than high-dimensional raw images. Developing these models for solar flare prediction thus requires extracting numerical attributes from raw data (in this case, raw magnetogram images). This methodology, called *feature engineering*, is usually a preferred approach for problems where domain knowledge has been sufficiently developed, which is true for the solar magnetic eruption problem. Extensive studies for understanding the magnetic eruption process has allowed solar physicists to identify important physical features derived from the magnetic field images that are useful indicators for solar flares, and thus solar eruptions. Some examples of such features are the total magnetic flux, the magnetic flux around the neutral lines (boundaries shared by highly positive and highly negative flux regions), the vertical current, current helicity, etc. A complete list of these features for each observed active region is available in a public database (described later in Section 3.2).

It is no surprise that a large majority of the ML-based flare prediction literature, discussed in Chapter 2, use these features. The advantage of feature-based models is their relatively lower

complexity as compared to the newer deep learning models. However, the features may not represent the most comprehensive and representative set that could be possibly extracted from the raw data. The other approach is to use raw image data directly as input to an ML model. This approach, adopted by many of the latest deep learning methods such as convolutional neural networks or self-attention models, involves automatically learning salient features from raw data as part of the training process. Such methods are very useful in the absence of domain knowledge. Further, they can be useful in supplementing the information available from engineered features. However, a drawback of these methods is their high complexity when compared with traditional ML models.

In this thesis, I introduce a novel feature extraction process based on the abstract shape of the magnetic field layout of an active region. The chief motivation for developing these abstract topology-based features is domain-independent featurization with the intention of training ML models of lower complexity, an important objective of this thesis. In this chapter, I will first describe the magnetic field data that I use in this thesis, discussing the challenges that it poses for machine learning. I also describe the labeling methodology, i.e. how each magnetic field image is labeled as flaring or non-flaring. I then introduce the standard physics-based feature set that has been popularly adopted by ML-based solar flare prediction literature. Finally, I introduce new shape-based features by applying topological data analysis to magnetogram data. The following chapters will use both of these feature sets to train and evaluate various ML models.

3.1 Data

Over the past few decades, various earth-based and orbital telescopes have been observing the magnetic field of the Sun for the purposes of scientific study, as well as forecasting of solar eruptions. Of these, one that has been widely adopted for ML-based solar flare prediction is the Solar Dynamics Observatory (SDO) [79]. Launched in 2010, the SDO mission uses the Helioseismic and Magnetic Imager (HMI) instrument [84] to observe full-disk vector magnetic field images of the Sun's photosphere at a cadence of 12 minutes. These images, available on the Stanford Joint Space Operations Center (JSOC), capture the magnetic field evolution of the active regions (ARs) as they

travel across the solar disk. Since these ARs only occupy a small part of the solar disk, it is beneficial to cut out rectangles around them and use these cut-outs for flare forecasting. The Solar HMI Active Region Patches (SHARPs) database, maintained by the JSOC, stores these patches, indexed by the active region number and the time-stamp of observation. Multiple representations of the vector field data are available in the database; in this thesis, I will be using the `hmi.sharp_cea_720s` variant, which stores the Lambert Cylindrical Equal-Area projection of the magnetic field — B_r, B_θ and B_ϕ. Of the three field components in each SHARPs image, I use only the radial component, B_r. This is standard practice in ML-based flare forecasting work, since the radial surface field is used as the boundary condition for computing global coronal magnetic fields that are not yet routinely measurable [16]. Example images of the radial component are shown in Fig. 1.3 of Chapter 1.

In this thesis, I choose SHARPs records from May 2010 to December 2017, using a one-hour cadence to reduce redundancy between consecutive images. This data set contains a total of $505,872$ records. Of these, some of the images lie close to the limb of the Sun, containing NaN values for some of the pixels, and therefore unsuitable for training ML models. Thus, I filter out all images containing NaN values from the dataset, resulting in a final dataset of $459,042$ images. For training supervised ML models on this data, each record has to be categorized as flaring or non-flaring. The flaring data is recorded by the NOAA Geostationary Operational Environment Satellite (GOES) X-ray Spectrometer, which measures the solar X-ray flux intensity for wavelengths in the range 1-8 Å. The observations are available in the NOAA XRS flare catalog[1] which contains records of the location, time, and magnitude of all solar flares since 1975. This catalog categorizes flares into five classes based on their peak X-ray flux intensity, from the weaker A, B and C, to the stronger M- and X-class flares. The flux intensity, in $\frac{W}{m^2}$, for each of these classes is presented in Table 3.1. For the period of May 2010 to December 2017, the GOES database reports 509 M-class flares and 36 X-class flares. Focusing on the stronger flares, because of their potentially catastrophic consequences, I label each SHARPs as flaring if an M1.0+ flare occurred (one with an intensity above $10^{-5}\frac{W}{m^2}$) within 24 hours after the observation time; otherwise, it is labeled as non-flaring. This yields a

[1] `www.ngdc.noaa.gov/stp/space-weather/solar-data/solar-features/solar-flares/x-rays/goes/xrs/`

Flare class	X-ray flux in $\frac{W}{m^2}$
A	$< 10^{-7}$
B	$10^{-7} - 10^{-6}$
C	$10^{-6} - 10^{-5}$
M	$10^{-5} - 10^{-4}$
X	$> 10^{-4}$

Table 3.1: Peak output flux measured from the GOES X-ray band (1-8 Å).

data set containing 3872 active regions, with 453,273 SHARPs labeled as non-flaring and 5769 as flaring.

This kind of imbalance is a major issue from a machine-learning point of view, as I will discuss in the later chapters. I now discuss the two approaches for extracting features for each of the magnetogram image of this dataset.

3.2 Physics-based features

Each record in the SHARPs data set includes a set of metadata containing values for a variety of numerical attributes that represent properties of the corresponding active region. These attributes, developed and refined by the solar physics community over the past 11 years, are given in Table 3.2. They are predominantly derived from the raw magnetic flux observations and are believed to be useful indicators of solar flares. Their values are calculated automatically and stored by the JSOC group at Stanford[2] . The vast majority of ML-based flare forecasting work has, as mentioned above, used these 20 quantities as the feature set.

3.3 Shape-based features

The evolution of the magnetic fields in the Sun during the lead-up to a solar flare manifests as an increase in the complexity of the structures on the magnetogram. This observation, which is visually obvious from the shapes of the regions in Fig. 1.3, plays a critical role in the qualitative classifications used in operational space-weather forecasts. The McIntosh classification system used

[2] http://jsoc.stanford.edu/

Acronym	Description	Units
LAT_FWT	Latitude of the flux-weighted center of active pixels	$degrees$
LON_FWT	Longitude of the flux-weighted center of active pixels	$degrees$
AREA_ACR	Line-of-sight field active pixel area	MH
USFLUX	Total unsigned flux	Mx
MEANGAM	Mean inclination angle, gamma	$degrees$
MEANGBT	Mean value of the total field gradient	G/Mm
MEANGBZ	Mean value of the vertical field gradient	G/Mm
MEANGBH	Mean value of the horizontal field gradient	G/Mm
MEANJZD	Mean vertical current density	mA/m^2
TOTUSJZ	Total unsigned vertical current	A
MEANALP	Total twist parameter, alpha	$1/Mm$
MEANJZH	Mean current helicity	G^2/m
TOTUSJH	Total unsigned current helicity	G^2/m
ABSNJZH	Absolute value of the net current helicity	G^2/m
SAVNCPP	Sum of the absolute value of the net currents per polarity	A
MEANPOT	Mean photospheric excess magnetic energy density	$ergs/cm^3$
TOTPOT	Total photospheric magnetic energy density	$ergs/cm^3$
MEANSHR	Mean shear angle (measured using B_{total})	$degrees$
SHRGT45	Percentage of pixels with a mean shear angle greater than 45 degrees	$percent$
R_VALUE	Sum of flux near polarity inversion line	G

Table 3.2: The SHARPs feature set. Values for these 20 features, and the associated error estimates, are available for each magnetogram in the SDO HMI database. Abbreviations: A and mA are Amperes and milli-Amperes, respectively; Mm is megameters, G is Gauss, MH is micro-hemispheres and Mx is Maxwells.

at the NOAA Space Weather Prediction Center, for instance, is based on characteristics like the presence of umbras and penumbras. These spatial details are, however, largely absent from the definitions of the SHARPs features of Table 3.2, most of which are aggregate quantities such as means or totals. Thus it seems natural to explore whether features that characterize shape would be useful in ML-based flare forecasting methods—not only because this is what human forecasters use in their classifications, but also because AR shape in the photosphere has fundamental, meaningful connections to the coronal magnetic field physics leading up to an eruption.

Topology is the fundamental mathematics of shape: it distinguishes sets that cannot be deformed into one another by continuous transformations. Part of this shape classification (homology) corresponds to the number of components, 2D holes, 3D voids, etc., of a set. These numbers are the "Betti" numbers, $\beta_0, \beta_1, \beta_2, \ldots$, where β_k is the number of k-dimensional "holes." Topological data analysis (TDA), or computational topology, operationalizes this framework for situations where one has only finitely many samples of an object. One way to create a shape from finite data is to "fill in the gaps" between the samples by treating two points as connected if they lie within some distance ϵ of one other. Building an approximating object at an ϵ-scale, TDA computes the Betti numbers, then varies ϵ and repeats the process. The dependence of β_k on ϵ provides an approximate topological signature that captures the "shape" of an object at multiple resolutions. A richer representation tracks the ϵ value at which each component or hole is formed, and at which it is destroyed or merges with another. This results in a set of $(birth, death)$ values of ϵ for each component or hole. This methodology has proved to be quite powerful; it has been successfully applied to a range of different problems ranging from coverage of sensor networks [25], to structures in natural images [38], neural spike train data [90], and even the large-scale structure of the universe [102].

My strategy for employing TDA in solar-flare forecasting, which I first proposed in [29], is somewhat different from the approach described in the previous paragraph. Conjecturing that the shapes of the level sets of a magnetogram are of central importance, I use the magnetic field intensity B_r as the variable parameter, rather than a distance ϵ. I threshold the SHARPs image,

keeping only the pixels where the magnetic field intensity falls at or below some value (i.e., sub-level thresholding) then compute the topology of the resulting object. By varying the threshold and tracking the birth and death of each feature, I obtain a signature that captures the morphological richness of a magnetogram in a manner that factors in the field strength as well as its spatial structure. A brief synopsis of this TDA-based feature extraction process follows, more details are given in [29].

I start by building what is technically known as a cubical complex [48] from the pixels in the SHARPs image whose B_r values fall below some threshold. Pixels are connected in such a complex if they share an edge or a vertex. Since magnetograms are 2D images, only connected components and non-contractable loops make sense—there is no higher-dimensional structure. Counting these gives β_0 and β_1 for the given threshold field, and this computation is then repeated for a range of B_r thresholds.

Figure 3.1 demonstrates this procedure for a simple example. Panel (a) represents an image, with each pixel color-coded according to intensity. Given an intensity value, a sub-level thresholded image corresponds to those pixels with a magnitude at or below the threshold. The gray regions in panels (b)-(f) represent such images for thresholds $B_r \in [0, 4]$. Pixels that share an edge or a vertex in a thresholded image become a component, which contributes to β_0. Empty regions (black) in the interior of a thresholded image that are surrounded a loop of connected gray pixels become holes, incrementing β_1. In panel (b), where $B_r = 0$, the gray, thresholded image contains two components separated by the empty, black region where the intensity is larger than zero; thus $\beta_1 = 2$. There is a single non-contractable loop in the image (the red curve) that encloses the empty region, so $\beta_1 = 1$. Increasing the threshold to $B_r = 1$, panel (c), causes the image to enlarge, shrinking the hole and splitting it into two; thus $\beta_1 = 2$ (the green box corresponds to a loop in the image since gray pixels are connected at vertices). The two components from panel (b) merge in panel (c), and for the remainder of the thresholding process, the number of components remains one, so $\beta_0 = 1$. Upon raising the threshold (panel d), the dominant hole splits into seven while the single pixel hole remains intact, bringing the number of holes to eight ($\beta_1 = 8$). At $B_r = 3$ in panel (e), two holes

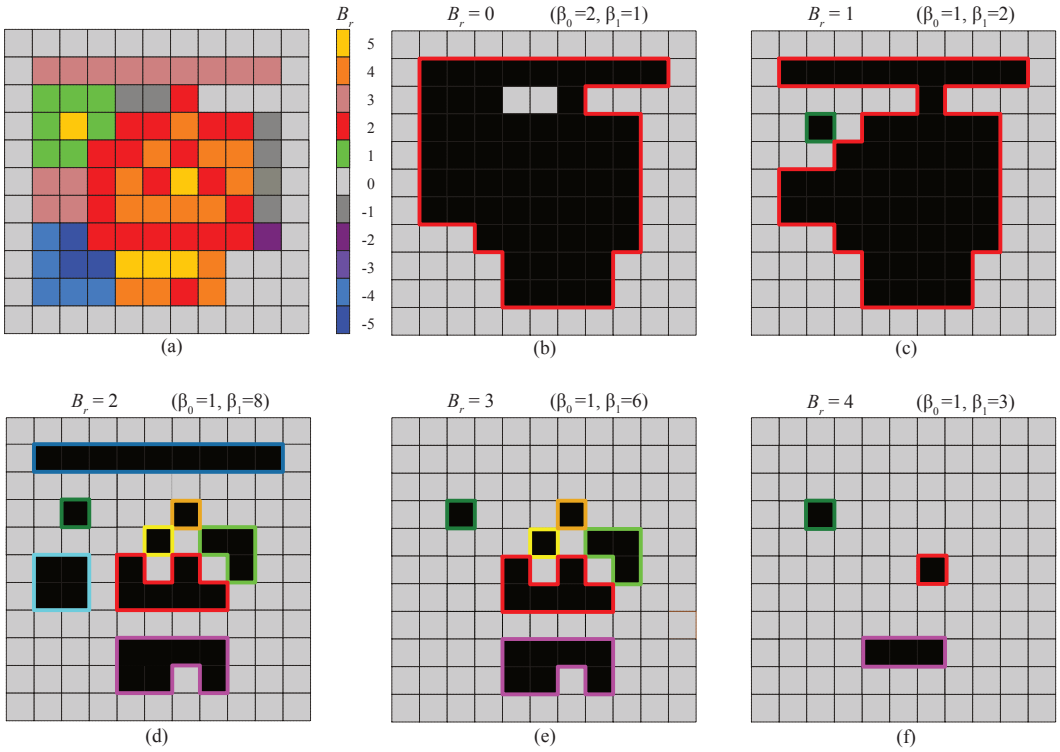


Figure 3.1: Topological data analysis for the example "image" in panel (a). The gray pixels in panels (b)-(f) represent the cubical complex of the image for five sub-level thresholding values, $B_r = 0, 1, 2, 3$, and 4. For each complex, the (β_0, β_1) values are given. The colored loops represent the holes in the thresholded images.

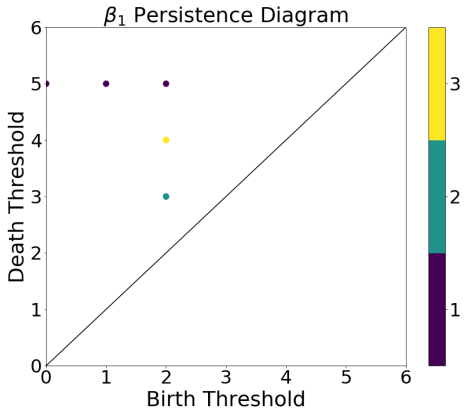


Figure 3.2: The β_1 persistence diagram for Fig. 3.1 The color scale represents the multiplicity, i.e., how many points share the same $(birth, death)$ values.

disappear or *die*; thus $\beta_1 = 6$. Finally, for $B_r = 4$ in panel (f), the number of holes is reduced to three as three of the holes are filled in by new image pixels. If the threshold were raised to five (not shown), all pixels would be filled so that the image would have $\beta_0 = 1$ and $\beta_1 = 0$.

A number of different representations have been developed by the TDA community to capture information about the scale and complexity of the different structures [38]. One of the most common, the persistence diagram (PD) [34], is a plot of the birth and death filtration values for each feature (e.g., the threshold value of B_r). Figure 3.2 shows a β_1 PD for the example of Fig. 3.1. Each point in this PD corresponds to the $(birth, death)$ thresholds for a hole in the filtered image. We use color to indicate the number of holes with same lifespan. In our example, there are three holes that live until $B_r = 5$. The longest-living hole, indicated by the red border in Fig. 3.1, is born at $B_r = 0$, and thus corresponds to the point (0,5) on the PD. The single pixel hole (green loop) is born at $B_r = 1$, giving the point $(1, 5)$ on the PD, and the hole with the magenta border is born at $B_r = 2$ so it appears on the PD at $(2, 5)$. There are also five relatively short-lived holes; three of them (with yellow, orange and light-green borders) are represented by the yellow point $(2, 4)$ on the PD. The remaining two are the shortest-lived (cyan and blue) and correspond to $(2, 3)$. Points that lie far from the diagonal on a PD are said to be *persistent*, as their birth and death values are widely separated. Points near the diagonal, which have short lifespans, are often formed due

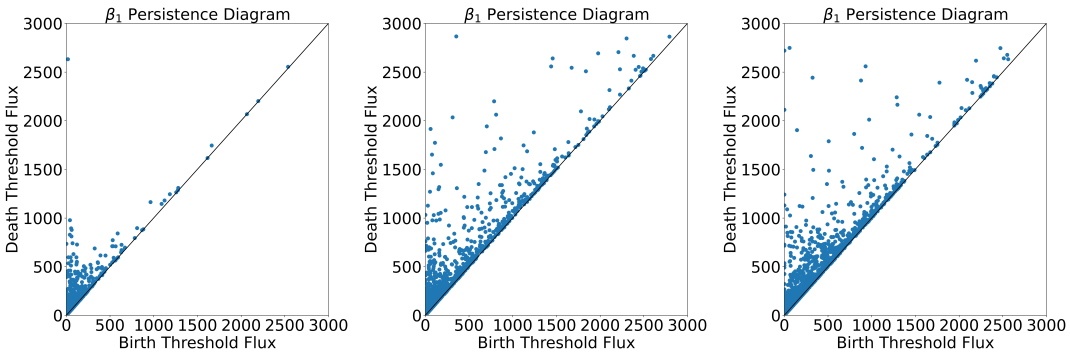


Figure 3.3: β_1 persistence diagrams generated from the positive magnetic flux density values of the magnetograms of Fig. 1.3. A clear change in the topology of the field structure is observed well in advance of the major flare eruption that occurred in this AR at 0910 UT on 6 September 2017.

to noise in the data [38].

One can similarly construct a β_0 persistence diagram to capture the birth and death of the connected components in the analysis. For the example here, this is not too interesting, since apart from the brief appearance of a second component at $B_r = 0$, there is only one component for all thresholds. This holds true for the β_0 PDs of actual magnetograms as well—they do not add much information to the analysis. For this reason, I restrict further analysis to β_1 PDs.

Persistence diagrams (PDs) of real-world images can be far more complicated. Figure 3.3 shows β_1 PDs for the AR from Fig. 1.3. As in Fig. 3.1, these persistence diagrams show only the positive sub-level thresholds for B_r. The three images in Fig. 3.3 clearly bring out the evolving structure of this AR. The relatively simple structure of the fields in Fig. 1.3(a) creates one long-lived hole in the PD of Fig. 3.3(a) at approximately (0, 2600), corresponding to the central white spot, along with a large number of short-lived holes near the diagonal, which correspond to short-lived structures likely due to noise. The persistence diagram in panel (b), 24 hours before an X-class flare, has a large number of long-lived holes, reflecting the complexity of the structure of the active region at this time. After the flare, the PD in the rightmost panel is still complex, but the number of off-diagonal holes has reduced—the number of holes with a lifespan of 500 Gauss or more, for example, drops from 70 in panel (b) to 61 in panel (c).

We can also construct separate PDs for the negative flux regions by choosing negative threshold values, though in this case it is appropriate to use "super-level" instead of sub-level thresholding. Thus for this case I first include—for the threshold $B_r = 0$—all pixels with $B_r \geq 0$, and then filter for increasingly negative values of B_r. For example, in Fig. 3.1(a), this process will leave a hole surrounding the blue pixels for a super-level threshold of $B_r = -3$. For the AR of Fig. 1.3, the resulting PDs (not shown) exhibit a similar evolution pattern to those in Fig. 3.3.

These results suggest that PDs can be useful indicators of impending solar flares. To operationalize this in the context of machine learning, however, there is an additional challenge. ML models generally require data that have a fixed dimension, but the PD contains an arbitrary number of $(birth, death)$ tuples. Various approaches to this "vectorization" problem have been proposed [2, 13, 17, 18, 19, 54, 55, 82]. Here I use a simple technique: I choose 10 equally spaced thresholds with $B_r > 0$ for sub-level thresholding and 10 thresholds with $B_r < 0$ for super-level thresholding, setting the maximum $|B_r| = 5000G$; this gives a range that covers the magnetic flux observed in most active regions.[3] The threshold spacing was chosen to ensure a good balance between redundancy (i.e., closely spaced, highly similar images with identical β counts) and adequate representation (avoiding a spacing so coarse as to miss important information). This was validated by experimentation using configurations of various spacing values, and determining the one which gave the most optimal prediction score using standard machine learning models described in the next chapter. As future work, a more rigorous approach could be to study the variation of the β counts over a very fine initial grid of resolutions. One could then use it to choose a final adaptive grid that is coarser where the β counts vary slowly as a function of threshold, and is finer where they vary rapidly.

This process produces two 10×10 discrete PDs, one for positive and one for negative B_r. Instead of this full information, I construct a 10-dimensional vector from each PD by simply counting the number of "live" holes at each threshold. These 20 values make up the feature set to be

[3] Specifically the thresholds are $\{263G, \cdots, 4473G, 5000G\}$ for sub-levels and $\{-263G, \cdots, -4473G, -5000G\}$ for super-levels.

used for training ML models in Chapter 4.

Chapter 4

Modeling Strategies: Is More Complexity Better?

In Chapter 3, I discussed two different feature sets that are derived from raw solar magnetogram images. The first feature set, also the more popularly used one, consists of physics-based attributes (called SHARPs features) that have been carefully handcrafted by domain experts, while the second feature set is the abstract topological properties proposed in Section 3.3. In ML-based solar flare prediction literature, a host of machine learning models have been developed, most of which use the SHARPs features. These range from simpler statistical models like linear discriminant analysis (LDA) to more complex models such as neural nets (see Chapter 2 for a complete survey of the ML models used in different papers). While some papers in this literature have provided a performance comparison of different models, a systematic comparison in terms of both model complexity and performance has not been done before. Additionally, for a fair model performance comparison, each model needs to be optimized in terms of its hyperparameters, a methodology that is not necessarily followed in existing studies. In this chapter, I aim to tackle both of these issues by proposing a systematic comparison of ML models in terms of both performance and complexity using an automatic hyperparameter tuning framework. I specifically address two important research questions:

- How does increasing model complexity affect 24-hour forecast accuracy for solar flares?
- Are there useful ways to move beyond the set of physics-based attributes in the SHARPs metadata?

The outline of this chapter is as follows. First I describe in brief the cohort of ML models used in my comparison study. Next, I dive into the details of the hyperparameter tuning framework for optimizing these models. I then discuss the results from the comparison experiments, including the effects upon model performance for different forecasting windows. I also discuss the performance comparison between the baseline SHARPs and the novel TDA-based features over the four different models. Taking the complexity/performance analysis further, I investigate how lowering the complexity via feature set reduction affects the model performance.

4.1 Machine Learning Models

In this study, I use four different models: logistic regression, multilayer perceptrons, long short-term memories, and extremely randomized trees. For each of these models, I tune the relevant hyperparameters as explained in Section 4.2, then compare the results on the flare-forecasting problem across different feature sets and models trained on the dataset described in Section 3.1. In the following paragraphs, I give brief descriptions of these methods; for more details, please see [39, 73] or any other basic machine-learning reference.

Logistic regression, perhaps the simplest of all models in the ML literature, uses a sigmoid function as the basis function to fit the model (h) to the data (x):

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

Here, θ is a vector of n weights that are applied to each element of the n-dimensional input vector. For $x \in \mathbb{R}^n$, the model output $h_\theta(x)$ takes the range $[0, 1]$. In the context of the flare-prediction problem, $h_\theta(x)$ represents the flaring probability, which we then convert to a categorical "flare/no-flare" output using a threshold of 0.5. That is, an input x with a model output value of $h_\theta(x) < 0.5$ is classified as non-flaring, else is classified as flaring. Training this model is a matter of determining values for θ; we accomplish this using the LBFGS algorithm of [64]. The only hyperparameter involved in this process is the class weight that is used in the gradient descent: the higher the weight for one class, the more the model is penalized for getting the classification for that class

wrong.

A **multilayer perceptron** or MLP is a type of feedforward artificial neural network containing multiple layers of nodes or *neurons*, with the outputs of each layer propagated forward to the next layer. These canonical ML models contain an input layer, some number of hidden layers, and an output layer. The output of each neuron is a nonlinear function of its inputs with a single free parameter (typically a multiplicative constant) that is "learned" during training using some gradient descent optimization on a suitable loss function: in this case, the weighted binary cross-entropy loss function, as described in [29]. We employ the Adagrad optimizer [33] to update the weights during gradient descent and an architecture with five dense layers with 36, 24, 16, eight and two nodes, respectively, a configuration optimized via a manual trial-and-error approach. There are two important hyperparameters here: the class weight, which plays a similar role as in the logistic regression model, and the L2 regularization constant in the loss function, which ensures that none of the neuron weights get too large, which would lead to overfitting.

Active regions evolve over time in ways that are meaningful from the standpoint of solar physics. Logistic regression and MLPs cannot leverage the information that is implicit in a sequence of magnetograms, as their forecasts are based on individual snapshots. For that reason, a thorough evaluation of ML-based solar flare forecasting should include methods that factor in the history of the observations. To that end, I use a **long short-term memory** (LSTM), which is designed to work with temporal sequences of data. LSTMs operationalize this via the addition of a feedback loop in the hidden layer that takes the state calculated in time step t_{n-1} and feeds it to the same network when processing the sample at time step t_n. The LSTM used in this study contains the same number of hidden layers as the MLP discussed above, but with such a feedback loop incorporated in one of the hidden layers. This strategy for propagating information forward in time is powerful, but it can complicate the training of these models. Simple gradient descent, for instance, can be problematic if there are long-term temporal dependencies in the data, since the gradient can decay as it is propagated through the timesteps. To address this, LSTM nodes often incorporate mechanisms called "forget gates" that limit the number of steps through which

information is propagated forward in time. This limit is an important choice, and one that is generally tuned by hand for a given model and data set. We take that approach here, finding that a sequence length of ten works well in our application. Like MLPs, members of this class of models have two hyperparameters that significantly impact their performance: the weights in the binary cross-entropy loss function and the L2 regularization constant.

Decision trees are a wholly different class of ML models. As the name suggests, they have a tree-like structure, where each branch point represents a decision based on some attribute of the data and the leaf nodes correspond to the salient classes for the problem at hand (flaring and non-flaring, in our case). The values of the different attributes of each input datum dictate the path taken through the tree during the classification process, eventually routing the outcome to one of these classes. A major advantage of this strategy is that its results are an indication of how well each individual feature is able to divide the data set: a major step towards explainability, a critical challenge in modern AI. The main disadvantage of decision trees is their tendency for overfitting. One can mitigate this by building an ensemble of decision trees using a different randomized feature subset selection at each branch point—a so-called "Random Forest"—and use the mean or mode of their predictions to classify a sample. The **extremely randomized tree (ERT)** [37] that I use in this thesis is a variant of this approach. Training these models involves three hyperparameters: the class weight; the minimum impurity decrease, which controls when nodes will split; and the number of the trees in the ensemble, which plays a role in overfitting.

This set of choices covers the full gamut of ML methods, in terms of architecture (e.g., tree-based or not) and complexity (number of free parameters), making it a sensible evaluation set for the research question. There is another issue, though. Since the performance of an ML model depends on both the data and the training process, and because that training process is governed by the model's hyperparameters, a fair comparison of two different ML models requires tuning their hyperparameters individually, as described next.

4.2 Model Evaluation and Tuning

Before I dive into the details of hyperparameter tuning, a quick word on how the above ML models are evaluated on the available data. The dataset, consisting of $453,273$ magnetogram images over 3872 active regions, has already been described in the previous chapter. Training, optimizing and evaluating ML models requires splitting this data into training, validation, and testing sets respectively. A high-level summary of this procedure follows; see Section 3.1 for the full details. I assign all images of a given AR to a single set, rather than choosing the images for each set randomly. This ensures that the data for the same AR at different points in its evolution does not appear in both the training and testing sets, thus preventing a possible artificial score improvement from testing the model on data related to that used for training. Of the 3872 active regions, I use 70% for training (for fitting the model to the data) and 30% for the testing (to evaluate the trained and tuned models on previously unseen data). Notice that I do not outright reserve any samples for the validation set. I use k-fold cross-validation to choose the validation samples from the training set; I will discuss this in detail in the next section. I produce ten different training-testing splits through randomized selections, each generated with a different seed. These serve as ten trial runs of each experiment.

To optimize the hyperparameters for an ML model, one evaluates its performance on a subset of the data, called the validation set, for different hyperparameter combinations. The best-performing combination is then used to train the model on the training set before it is evaluated on the corresponding test set. Using distinct subsets of the data for these three purposes ensures that the processes are completely independent, and thus not artificially boosted. Instead of using a single validation set, a better strategy is to perform what is called a k-fold cross-validation. In this approach, the training set is divided randomly into k subsets, or "folds," of roughly equal size. For each hyperparameter combination being evaluated, one of the k folds acts as the validation set and the remaining $k-1$ folds are merged together into the training set. After being trained on this $k-1$-fold training set, the model is then tested on the 1-fold validation set. This process is

repeated using each of the k folds, individually, and the success of the hyperparameter combination is judged by the mean of the model performance metric across these runs.

This process is illustrated in Figure 4.1. The first step is the standard random splitting of the data set. I repeat this ten times using a 70%-30% ratio to generate 10 trial runs, then generate k folds on each of those training sets. In machine-learning practice, k is generally chosen as 3, 5, or 10. In my study, $k = 5$ worked well for obtaining sufficiently long training and validation sets. A second important decision in this tuning process is the sampling method for the hyperparameter value combinations. Various strategies have been proposed for this in the ML literature: grid search, random sampling, Bayesian sampling, etc. Here, I use the Bayesian optimization method of [69], a Gaussian-process based approach that uses a mixture of exploration and exploitation to optimize hyperparameter combination samples. I employ the Python implementation `BayesOptSearch` and use the `ray.tune` Python library [62] to deploy the sampling and evaluation process on the graphics processing unit. As shown in Figure 4.1, the `BayesOptSearch` method iteratively samples 40 hyperparameter combinations and chooses the combination that gives the best performance on the validation set, as measured by some metric that compares the forecast to the ground truth (the choice of this metric has some important implications; I use the True Skill Statistic, as described further in the following section.) For `BayesOptSearch`, I use the upper confidence bound as the acquisition function with the exploration parameter κ set to five in order to strike a good balance between exploitation and exploration in the sampling process; see [91] for more details. That sampling process begins with 10 uniformly chosen samples over the hyperparameter space, continuing in steps of 10 until the algorithm has explored 40 different hyperparameter combinations.[1] The model is then trained on the full training set using the best-performing hyperparameter values before being evaluated on the 30% test set.

Not all hyperparameters require this kind of complex, computationally intensive treatment; some of them can be quite effectively tuned using a manual trial-and-error approach. I employ a two-phase model-tuning strategy, first performing a hand optimization of hyperparameters like the

[1] i.e., using `initial_random_steps` in `ray.tune`, with `max_concurrent_trials`=10

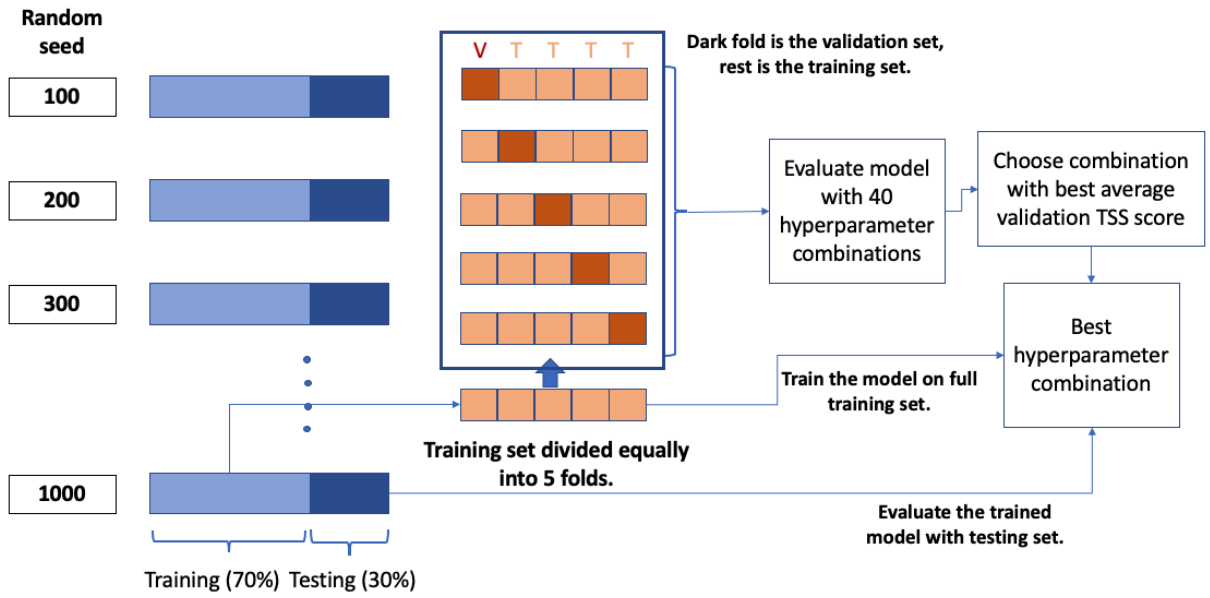


Figure 4.1: Hyperparameter tuning workflow.

number of model layers, the nonlinear activation function, the optimization function, etc.—choices that are made from among a finite number of options, and that generally impact all metrics in a similar way. This is followed by the automated approach described above for tuning hyperparameters such as the loss function weights or the regularization penalty. Automatic tuning of these parameters is important for two reasons. Firstly, they take on continuous values, so hand-tuning them to a precise number becomes cumbersome. Secondly, their effects are not independent: changing one of them often improves one metric at the cost of another—a situation where an automated, systematic exploration of the search space can be especially appropriate. It is entirely possible, of course, to use the automated approach for *all* the hyperparameters, but that significantly increases the computation time.

4.3 Model Comparison Results

In this section, I compare the relative prediction performance of the machine-learning based flare-forecasting models described in Section 4.1 for feature sets covered in Chapter 3. After extracting values for those features from the labeled data set, I carry out the hyperparameter tuning procedure outlined in the previous section on each model/feature-set combination, using the k-fold cross validation approach on the 10 datasets, then train the model with optimized hyperparameters on the corresponding training set. To evaluate the results, I run the model on the corresponding test set and compare its 24-hour forecasts to the ground truth using four standard prediction metrics: accuracy; the true skill statistic or TSS (also known as the H&KSS), the Heidke skill score (HSS_2), frequency bias (Bias), and F_1, which is the harmonic mean of precision and recall. These metrics, whose detailed formulae can be found in [6, 9, 24], are derived from the entries of the contingency table—i.e., the numbers of true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN), and are summarized in Table 4.1.

In the context of this problem, a flaring magnetogram is considered as a positive while a non-flaring magnetogram is considered a negative. Since our data set includes 5769 of the former and 447504 of the latter, accuracy is not a very useful metric here; a simple model that classified every

Metric	Formula
Accuracy	$\frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}$
True Skill Statistic (TSS)	$\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} - \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}}$
Heidke Skill Score (HSS_2)	$\frac{2(\mathrm{TP}\times\mathrm{TN}-\mathrm{FP}\times\mathrm{FN})}{(\mathrm{TP}+\mathrm{FP})(\mathrm{FP}+\mathrm{TN})+(\mathrm{TP}+\mathrm{FN})(\mathrm{FN}+\mathrm{TN})}$
F_1	$\frac{\mathrm{TP}}{\mathrm{TP}+\frac{1}{2}(\mathrm{FP}+\mathrm{FN})}$
Bias	$\frac{\mathrm{TP}+\mathrm{FP}}{\mathrm{TP}+\mathrm{FN}}$

Table 4.1: Metrics used for evaluating the binary forecasting models.

input as non-flaring would have a high accuracy of 98.7%. The different skill scores strike various balances between correctly forecasting the positive and negative samples. TSS ranges from $[-1, 1]$ and HSS_2 from $[-\frac{2PN}{P^2+N^2}, 1]$, where P and N are the total positive and negative samples in the dataset being evaluated. In both cases, these scores are 1 when there are no false positives or false negatives, while a score of 0 means the model is doing only as well as a random forecast: that is, an "always no-flare forecast." Bias has the range $[0, \infty]$, where $Bias < 1$ indicates underforecasting (many false negatives), and $Bias > 1$ implies overforecasting (many false positives). F_1 has the range $[0, 1]$ with 1 indicating a perfect forecast score. These metrics, all of which are used broadly in the flare-prediction literature, represent a good span of ways to quantify the performance of ML models.

As discussed in Section 4.1, hyperparameters for each model must be individually tuned in order to provide a fair comparison of their performance. The choice of metric plays a subtle role here, since hyperparameters can have different effects on the various metrics. Tuning performance based on values of one of them, then, can impact performance as measured by the others. The choice of which metric to use for a particular problem is often left as a decision for the forecaster based on their priorities: some might wish to have a model that prioritizes the TSS score (e.g. [30]), while others might prefer a model that has lower false positive rate [31] or is more reliable, as defined in [76]. In this application, a number of choices are possible. Optimizing based on accuracy

would be a particularly bad choice here, as it would lead to models defaulting to the "always no-flare forecast" configuration. I use TSS in my work, choosing hyperparameters that maximize its values via the k-fold cross-validation described above. TSS is a common choice in the flare-forecasting literature, as well as the broader deep-learning literature. It does come with limitations, however; optimizing the TSS score can lead to high false positives due to the high dataset imbalance, thereby impacting some of the other metrics like precision, F_1, and Bias. This effect manifests in the results described below.

These experiments were carried out on an NVIDIA Titan RTX (24 GB, 33 MHz) GPU for the deep-learning models (MLP, LSTM, and CNN), and on an Intel i9-9280X (3.30 GHz) CPU for the simpler models (logistic regression and ERT). [2] Run times ranged from 2.5 seconds to train and 0.02 seconds to test each logistic regression model on the SHARPs feature set to 210 seconds and 9 seconds for the LSTM model using the combined feature set. The run time of the hyperparameter tuning procedure also depended on the model and the data, ranging from 40 seconds for each logistic regression model with the SHARPs feature set to just under two hours for each LSTM model with the combined feature set.

Table 4.2 compares the performance of the various models across three feature sets: the traditional physics-based features that appear in the SHARPs metadata, the shape-based attributes extracted from each magnetogram image using topological data analysis via the procedures described in Section 3.3, and a third set that combines the two. Note that the topological features perform just as well as the SHARPs features. Moreover, the combined feature set does not provide any improvement over either of its two constituents, confirming that neither feature set provides a significant advantage over the other. This is an answer to the second research question: abstract spatial properties of active region magnetograms appear to give ML methods just as much traction on flare-forecasting problems as the set of physics-based SHARPs attributes, thus providing a way to move beyond these standardized SHARPs metadata features.

[2] Different machine-learning models lend themselves to different types of hardware, depending on how well they parallelize. The machine used to carry out these experiments affects only the run time, not the results.

Model	Feature Set	# of Parameters	Accuracy	TSS	HSS_2	F_1	Bias
Logistic Regression	SHARPs	21	0.87 ± 0.01	0.79 ± 0.01	0.13 ± 0.02	0.15 ± 0.02	11.44 ± 1.75
	Topological	21	0.87 ± 0.01	0.78 ± 0.02	0.12 ± 0.02	0.14 ± 0.02	11.88 ± 1.40
	Combined	41	0.87 ± 0.02	0.79 ± 0.02	0.13 ± 0.02	0.15 ± 0.02	11.76 ± 1.98
ERT	SHARPs	332	0.84 ± 0.01	0.79 ± 0.01	0.11 ± 0.01	0.13 ± 0.01	13.96 ± 0.81
	Topological	483	0.85 ± 0.02	0.76 ± 0.04	0.11 ± 0.02	0.13 ± 0.02	13.34 ± 1.93
	Combined	340	0.86 ± 0.02	0.77 ± 0.03	0.12 ± 0.02	0.14 ± 0.02	12.27 ± 2.12
MLP	SHARPs	2198	0.85 ± 0.02	0.76 ± 0.02	0.11 ± 0.02	0.13 ± 0.02	13.13 ± 2.42
	Topological	2198	0.85 ± 0.02	0.76 ± 0.02	0.11 ± 0.01	0.13 ± 0.01	13.56 ± 1.53
	Combined	2918	0.86 ± 0.03	0.76 ± 0.03	0.11 ± 0.02	0.13 ± 0.02	13.10 ± 1.78
LSTM	SHARPs	6662	0.87 ± 0.02	0.75 ± 0.02	0.12 ± 0.02	0.14 ± 0.02	11.90 ± 1.93
	Topological	6662	0.85 ± 0.02	0.75 ± 0.03	0.11 ± 0.02	0.13 ± 0.02	13.28 ± 2.11
	Combined	7382	0.86 ± 0.01	0.75 ± 0.02	0.12 ± 0.01	0.14 ± 0.01	12.00 ± 1.63
Optimal Scores			1.0	1.0	1.0	1.0	1.0

Table 4.2: 24-hour forecast performance of different machine-learning models using three different feature sets. The third column shows the number of free parameters needed to classify a single data sample. In case of the ERT, this is equal to the average depth of the tree (the path taken by a data sample from the root to a leaf node in the tree).

A between-models comparison addresses the first research question posed at the beginning: whether model complexity is an advantage in the context of this problem. The numbers in the table suggest that the answer is no. Indeed, the general trend in the TSS scores shows that increased complexity results in slight performance *reduction*. That is, while the MLP and LSTM models have orders of magnitude more parameters, the simpler logistic regression and ERT models in fact perform marginally better, as judged by the TSS scores. (For the other metrics, there is no significant change across the four models.) The lack of "value added" for the more-complex models may simply be due to data limitations: in general, the higher the complexity of a machine-learning model, the more data is needed for training. If the training set is too small, the model will overfit that data, causing it to fail to generalize well to the testing set. Simpler models, by their very nature, avoid this trap. Note that while ≈ 460000 samples might seem large, many of the images in the solar flare data set are similar, and many of those are non-flaring. In other words, the total amount of *information*—i.e., the number of sufficiently diverse and useful samples—is small in this data set. Determining whether data limitations are in play is an important open problem in current ML research. One way to approach it is to compare the values of the weighted binary cross-entropy loss function across the models; another is to observe the patterns in the convergence over the training process. Both are problematic for the more-complex models in our study. The ERT model does not generate a loss, nor does it have an iterative training process. For models that do have an iterative training procedure (MLP and LSTM), I carried out the second test, finding that the validation loss and training loss both reached asymptotes during the training process, suggesting (but of course not proving) that overfitting is not at issue. The first approach is not useful in the case of the LSTM because one-to-one loss comparisons are problematic in such models due to the variation in the number of samples in the training and testing sets that occurs when the original data are converted to temporal sequences by the feedback loop in the model. In the future, as more data is recorded by SDO/HMI, we will be able to learn—by re-running these experiments on longer, richer data sets—whether the lack of value-added for model complexity is an artifact of data limitations or something more fundamental.

To assess the effects of the prediction horizon, I carried out a set of experiments using the best-performing model/feature combination from Table 4.2—logistic regression with the combined feature set—and generated forecasts for 3, 6, and 12 hours in addition to the previous 24 hour case. The results are shown in Table 4.3. As one would expect, the shorter the forecast window, the higher the accuracy, but the story in the other columns of the table is more complicated: TSS is roughly similar for all forecasting windows, while HSS_2, F_1, and Bias actually *worsen* as the forecast window shrinks. This counterintuitive result is almost certainly due to an increase in imbalance that is created by shortening the window: the number of negative samples in the data set increases at the expense of the positive samples (for each flare, fewer magnetograms will be labeled as "flaring within m hours" if m is smaller). This is, again, a side effect of tuning on the TSS score, as discussed above: optimizing that particular score leads to a high false positive rate for a severly imbalanced dataset.

The LSTM is not only the most complicated model in this study—by a factor of three, as judged by the number of free parameters—but also the only one that uses the AR *history*. In view of this, its lack of performance is particularly striking. The feedback loop in this architecture makes it difficult to deconvolve the effects of the temporal history and the number of parameters, so we cannot say for sure whether or not the former, alone, confers any advantage, but all of the additional free parameters certainly do not appear to help.

Forecasting Window	Accuracy	TSS	HSS_2	F_1	Bias
24 hours	0.87 ± 0.01	0.79 ± 0.01	0.13 ± 0.02	0.15 ± 0.02	11.44 ± 1.75
12 hours	0.89 ± 0.02	0.82 ± 0.01	0.10 ± 0.03	0.11 ± 0.03	17.13 ± 4.10
6 hours	0.91 ± 0.01	0.82 ± 0.02	0.07 ± 0.02	0.07 ± 0.02	25.17 ± 4.83
3 hours	0.93 ± 0.01	0.79 ± 0.05	0.05 ± 0.01	0.05 ± 0.01	33.18 ± 4.60

Table 4.3: Logistic regression forecasts for different horizons.

4.4 Feature-Set Reduction

As part of the training process, ML models learn which attributes of the input data, in which combinations, are meaningful. Their efficiency in doing so depends on the amount and complexity of the data, and also—importantly—on the way it is represented. In general, a larger feature set leads to increased model complexity, which in turn can result in overfitting and increased computational time [39], as well as requiring more training data. The nature of the individual features also matters. Features that are not salient, or that are redundant, slow down the learning process. For these reasons, it is important to minimize the number of features and maximize their relevance.

A good way to approach this problem is to apply dimensionality-reduction techniques to the feature space in order to find the most relevant subspaces. A variety of methods have been proposed for this, including principal component analysis, linear discriminant analysis, t-distributed stochastic neighbor embedding [98], uniform manifold approximation and projection [70], etc. I use principal component analysis (PCA) because of its simplicity and effectiveness. It determines an alternative basis set to represent the data by iteratively constructing an orthogonal basis such that the variance of the data along the first dimension is maximal, the second dimension is in the direction of maximum variance that is orthogonal to the first dimension, and so on. One then typically keeps the first l, say, principal vectors that together account for some chosen fraction of the total variance. This approach, applied to a given feature set with n dimensions, effectively reduces the dimensionality from n to l. Note that any or all of the original n values may appear in the coordinates of each of these l basis vectors; I will discuss this more below.

Applying this approach to the ten randomly shuffled training sets discussed previously—first using the SHARPs feature set of Sec. 3.2, then the shape-based feature set of Sec. 3.3, and finally their combination—gives Fig. 4.2. For the SHARPs feature set, the first principal component captured 38% of the variance; adding a second component increases the total to 60%. The topological feature set is much more anisotropic: its first principal component captures almost 85% of the variance of the data set, a level that requires four principal components in the SHARPs feature set.

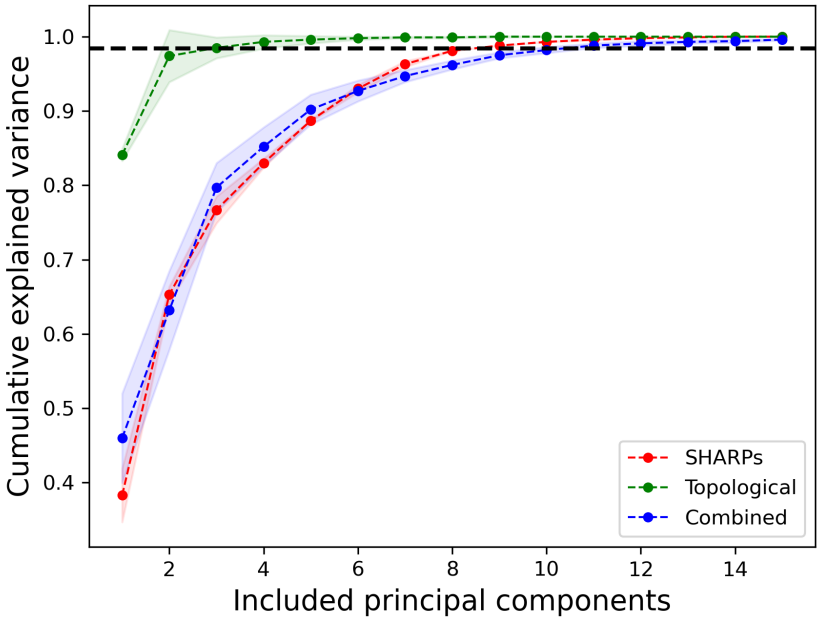
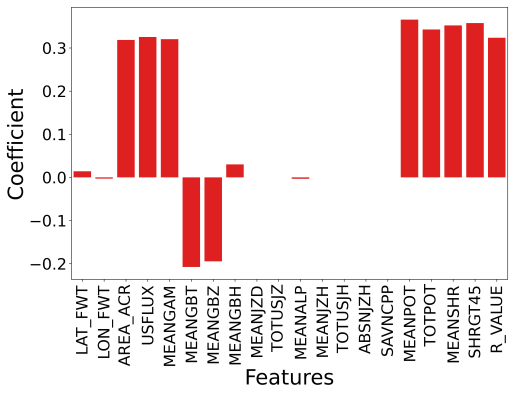


Figure 4.2: Cumulative explained variance plots of the principal components for the three feature sets, determined from the ten training sets. The darker curves represent the medians of the explained variance, while the shaded regions around them indicate standard deviations. The dashed line marks the 98.5% level.

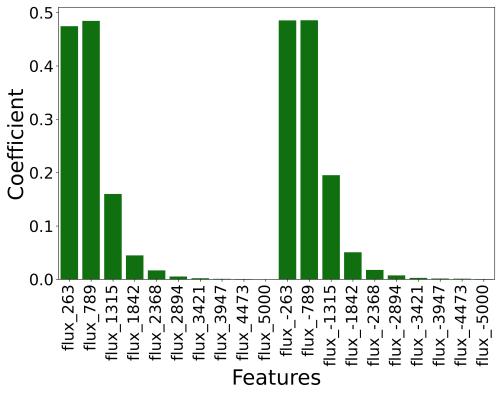
This is a direct reflection of the relevance of the TDA features for the purpose of flare prediction—as well as an indication of how many principal components will be needed as discriminating features for the corresponding ML models.

A detailed analysis of the reduced features is revealing. The coefficients of the first principal component—i.e., the weights of each SHARPs attribute in the coefficients of that basis vector—for one of the ten training sets are shown in Fig. 4.3(a). These results are to some extent consistent with the science: the `R_VALUE` and `USFLUX` attributes, for instance, which are known to be important for flare prediction, are weighted heavily in the first principal component of the SHARPs set. Note, though, that six other quantities are also weighted heavily in Fig. 4.3(a); moreover, this first principal component does not capture much of the variance, so one should not over-interpret its weights. Rather, one must think about *all* of the principal components for each feature set, acting together as a basis set for the new feature space. (A feature that is totally absent from every principal component, of course, is certainly not salient, but a low-weighted feature in the first one could have a high weight in another.) That being said, interpretation of Fig. 4.3(b) is somewhat different because, as Fig. 4.2 shows, the first principal component in the topological feature set captures a very large fraction of the variance, making it somewhat more safe to analyze in isolation. In view of that, it is interesting to note that the topological features at lower magnetic fluxes are weighted more heavily in the first principal component, and with some $+/-$ symmetry. This makes sense in view of the nature of the thresholding used in their construction: a high threshold value removes much of the small-scale structure of the magnetogram, leaving only the highest-magnitude pixels. Finally, note that the first principal component from the combined feature set is a weighted combination of both SHARPs and TDA features. As discussed later, this can have implications regarding model performance.

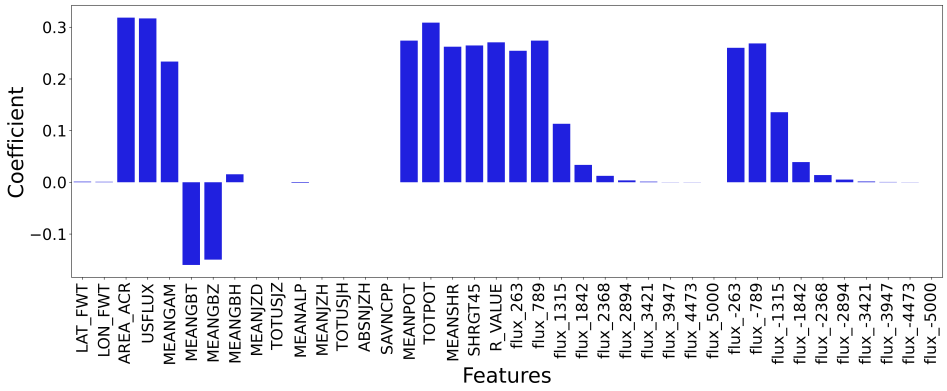
Again, PCA only finds a new basis set for the feature space; it does not produce a feature ranking. The previous paragraph is only to explain how one principal component is represented in terms of the original features, in one particular split of the data. While that narrative offers some possible ties to the physics, it does not represent a comprehensive, formal analysis of the feature



(a) (a) SHARPs features set



(b) (b) Topological feature set



(c) (c) Combined feature set

Figure 4.3: The weights of the SHARPs, topological, and combined features in the first principal component of the corresponding feature set for one of the ten training sets examined in this paper. Labels of the topological features indicate the flux level of the threshold value used to construct those features.

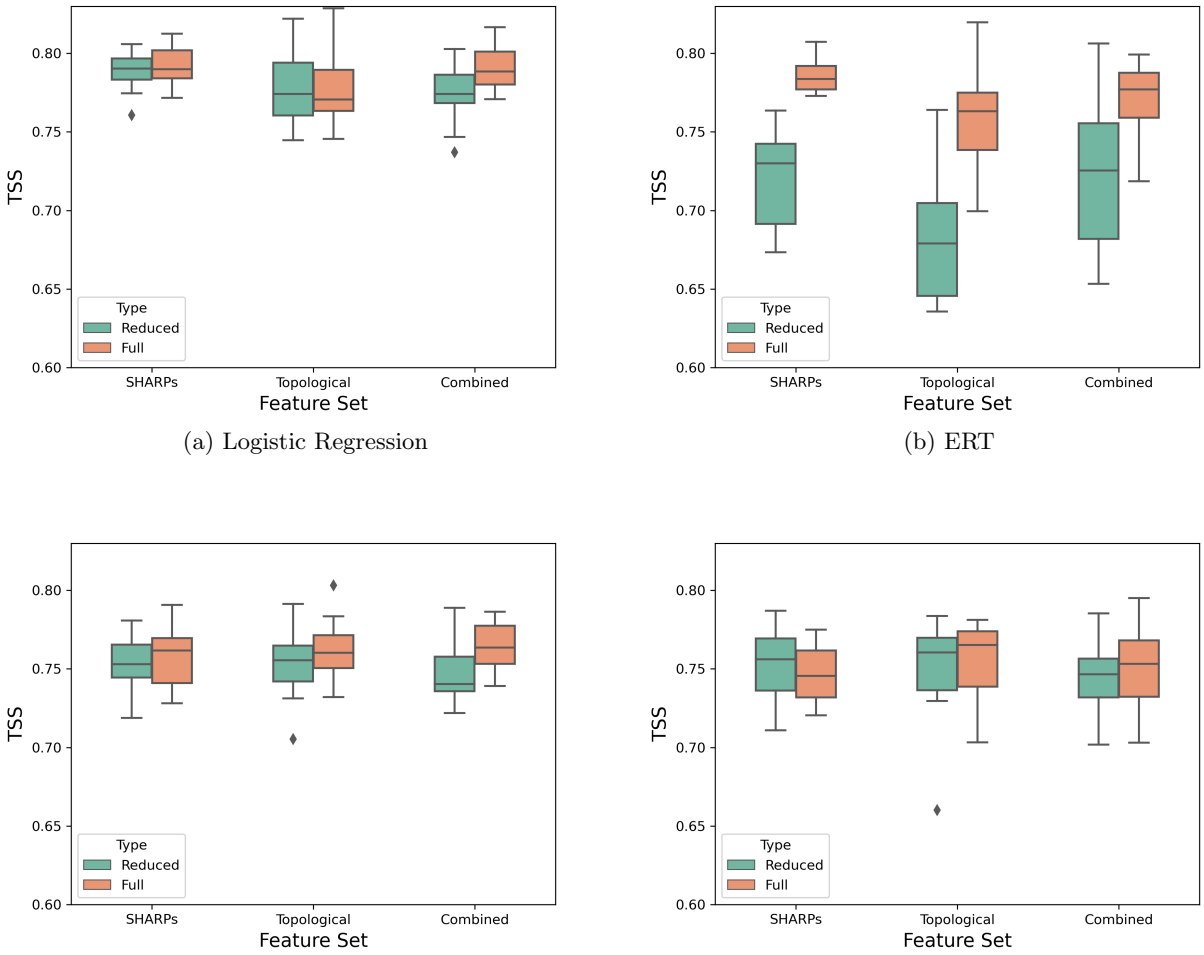
importance in the classification problem. I will discuss a detailed ERT-based feature ranking later in Chapter 5.

I address the problem of model complexity by performing reduction of the dimension of the feature sets, re-processing the data sets to calculate values for the PCA-based features—nine, three, and 11 dimensions from the orthogonal PCA representation for the traditional, topological and combined feature sets, respectively—then re-tuning the model hyperparameters and repeating the train/test procedure. The results are shown in Figure 4.4. For all three feature sets, the performance of the full and reduced-order models are similar for the logistic regression, MLP, and LSTM models, as judged by TSS scores. This extends to the other metrics as well, as demonstrated for the topological features across all four models in Tbl. 4.4. In other words, we can successfully simplify these three types of models by reducing the number of features *without sacrificing performance*. In view of the discussion above about data limitations, this is an obvious advantage, as models that work with smaller feature sets have fewer free parameters that must be learned from the same training data.

The ERT model, however, departs from this pattern, demonstrating marked reductions in TSS scores with the PCA-reduced feature set. This effect is strongest for the topological feature set, followed by the SHARPs feature set, with the combined feature set seeing the smallest impact. A possible explanation for this performance degradation with dimensional reduction lies in the way

Model	Feature Set	Accuracy	HSS_2	F_1	Bias
Logistic Regression	Full	0.87 ± 0.01	0.12 ± 0.02	0.14 ± 0.01	11.88 ± 1.40
	Reduced	0.87 ± 0.02	0.13 ± 0.02	0.15 ± 0.02	11.85 ± 2.04
ERT	Full	0.86 ± 0.02	0.11 ± 0.02	0.13 ± 0.02	13.34 ± 1.93
	Reduced	0.87 ± 0.04	0.11 ± 0.03	0.13 ± 0.03	11.69 ± 3.56
MLP	Full	0.85 ± 0.01	0.11 ± 0.01	0.13 ± 0.01	13.56 ± 1.53
	Reduced	0.86 ± 0.02	0.11 ± 0.02	0.13 ± 0.02	12.77 ± 2.31
LSTM	Full	0.85 ± 0.02	0.11 ± 0.02	0.13 ± 0.02	13.28 ± 2.11
	Reduced	0.85 ± 0.02	0.13 ± 0.02	0.11 ± 0.02	13.14 ± 2.43

Table 4.4: Performance comparison of the four different models trained on the complete topological feature set and its PCA-reduced counterpart. It can be seen that reducing the topological feature set does not impact the model performance.



(a) Logistic Regression

(b) ERT

(c) MLP

(d) LSTM

Figure 4.4: TSS score comparison in the form of box-whiskers plots across various models, trained on the three feature sets and their PCA reduced counterparts over 10 trials. The central line each box represents the median TSS score while the top and bottom edge of the boxes represent the 25 and 75 percentiles over the 10 trials. The whiskers on either sides are the upper and lower bounds for the scores in each experiment, and the dots represent the extremities. After PCA reduction, the SHARPs feature set is reduced from 20 to 9 features, the topological set from 20 to 3 features, and the combined set from 40 to 11 features. For all models except for the ERT, the reduced feature set does just as well as the complete feature set.

ERTs are constructed. In a k-feature ERT, $\sqrt{k}$ randomly chosen features are typically considered when determining the split at each branch point. A reduction in the number of features, then, effectively confines the exploration of the model-fitting search space. This could result in suboptimal splits at each branch point that do not effectively separate the positive and negative samples, thereby impacting the model performance. If this is indeed the case, one can work around it, to some extent, by modifying the training process to instead use all k features for choosing the best split. In this application, doing so raised the TSS scores for the topological and combined feature sets to the point where there is no statistically significant difference between the reduced and the full features, thereby validating the conjecture suggested at the beginning of this paragraph about the effects of the small number of features on the ERT construction.[3] For the dimensional-reduction experiment with the SHARPs feature set, however, increasing the number of features used in the split did *not* have much effect. This disparity is interesting: it suggests that the presence of shape-based features in the combined set—which allows the PCA algorithm to use them in combination with the SHARPs features—helps make up for the shortcomings of those physics-based features. This is in accordance with the PCA results in Figure 4.3(c) of Section 4.4, where both SHARPs and topological features are highly weighted in the first principal component of the combined feature set.

As a next step, one could investigate other methods for feature-set reduction, for example kernel-PCA [85], t-distributed stochastic neighbor embedding (tSNE) [98], uniform manifold approximation and projection (UMAP) [70], etc., that go beyond the linear reduction approach used in PCA. It would be interesting to see if the reduction using these techniques is more efficient than PCA, i.e. whether it leads to a smaller reduced feature set than PCA without impacting performance.

[3] This is a current topic of discussion in the ML community, but no consensus has yet been reached.

4.5 Summary

Machine learning-based solar flare prediction has been a topic of interest to the space weather community for some time now. Various machine-learning models, ranging from simple tools like logistic regression to incredibly complex "deep-learning" models such as multilayer perceptrons (MLPs) and long short-term memories (LSTMs), have been used to map the correlation between physics-based magnetic field features and the flaring probability in the near future. There have been studies that compared different machine-learning flare-forecasting models [6, 35, 74]; however, the dominant focus of these papers has been on performance comparison and they only used the basic SHARPs feature set. Furthermore, none of these approaches performed a systematic, automated hyperparameter tuning process to assure a fair comparison, and only [35] did any hyperparameter tuning at all (using a grid search method on discrete hand-selected values). I addressed some of these issues in this chapter that reproduces the results from [32].

My first objective in this chapter was to systematically compare a set of machine-learning models and determine whether higher complexity is correlated with better performance. Using an automated hyperparameter tuning approach, I compared four models with increasing order of complexity: logistic regression, extremely randomized trees, MLP and LSTM. Across multiple experiments, and using an automated hyperparameter tuning approach to obtain a truly fair comparison, I showed that more-complex models do not perform better for the 24-hour M1.0+ flare-forecasting problem.

Secondly, I evaluated a new shape-based feature set, first introduced in [29], that is composed of quantities extracted from magnetograms using topological data analysis, comparing their results to the standard physics-based SHARPs feature set. The results extend upon my initial studies of this idea [29, 30, 49] by using a far more comprehensive SHARPs feature set, employing four different ML models, and using a systematic hyperparameter tuning methodology to ensure fairness in comparison. This broader and deeper study confirmed that these shape-based features—calculated using abstract universal algorithms that do not encode any assumptions about the underlying

mechanics—provide as much traction to ML models as the set of physics-based attributes that were hand-crafted by experts. Further, the combination of the shape-based and physics-based feature sets does not provide any improvement over the individual sets, confirming that neither feature set provides a significant advantage over the other.

Lastly, I studied a different aspect of model complexity: the dimensionality of the feature set. Using principal component analysis to find relevant subspaces of the feature space, I compared the performance of models trained on the full feature sets versus the same models trained on their PCA-reduced counterparts. I found that the feature subspaces for both the SHARPs and topological features were significantly shorter than their full spaces, with the topological feature set affording the highest degree of dimensionality reduction to just three components. This implies that the topological feature set is highly anisotropic, with just three dimensions explaining 98.5% of the dataset variance. Further, I found that the PCA-reduced datasets performed just as well when compared with the original feature sets for three out of the four models.[4] In a data-limited situation, this is a major advantage for the effectiveness of machine-learning based solar-flare prediction methods, since more-complex models generally require larger training sets.

[4] The only exception was the ERT model, due to architectural design choices.

Chapter 5

Reducing False Positives in a CNN-based Flare Prediction Models

The focus of the thesis so far has been on featurizing magnetograms of active regions — extracting attributes that discriminate flaring from non-flaring active regions, and using these representative attributes (or *features*) to train machine learning models. In the field of machine learning, there are multiple approaches for extracting features from raw data, one of which is to use domain knowledge of the application. In flare prediction, for example—as described at length in the previous chapter— solar physicists have determined various physics-based features known to be associated with the flaring phenomenon. While this approach works well, one cannot guarantee that the extracted features represent all the information necessary to discriminate between flaring and non-flaring magnetograms. A way around this is to use featurization methods that are domain-independent. I demonstrated this in Chapter 3, where I leveraged topological data analysis (TDA) — an abstract approach from applied mathematics — to extract features that represent the "shape" of magnetograms. These abstract features perform well in predicting flares as seen in Chapter 4, but the combination of topological and domain-specific features does not perform well over either individual set. In this chapter, I explore a different methodology for extracting additional features from raw magnetogram images that is also domain-independent: deep learning convolutional neural networks.

While feature-engineering based ML methods have been a popular approach for solar flare prediction over the last decade, the last few years have seen the application of more complex deep learning models such as convolutional neural networks (CNNs) to this problem. Given raw data,

the CNN models employ spatial convolution operations in a hierarchical manner to extract patterns that are relevant to the task at hand. Unlike the other ML methods explored in the last chapter (logistic regression, extremely randomized trees, etc.), this feature extraction occurs *as part of model training*, instead of the standard two-step process we have seen in the last two chapters (feature extraction and model training). For example, CNNs for flare prediction would automatically extract patterns in the magnetic field that discriminate flaring and non-flaring magnetograms. This is advantageous for the feature-extraction process: any feature learned during training is automatically relevant to the model task (classification or regression). In many standard applications, this approach has been applied with great success, even achieving accuracy surpassing humans [43]. Naturally, multiple efforts in recent times have attempted to apply CNNs to the solar flare prediction task [1, 44, 61, 78, 107, 108]. However, the high imbalance in the flare prediction dataset has posed challenges to these implementations, resulting in models that predict more flares than present in the dataset (or false positives)—aka overforecasting—an issue which I also demonstrated in the context of the standard ML models in Chapter 4.

In this chapter, my goals in designing a CNN-based flare prediction model are two-fold. The first is to provide a preliminary comparison of a CNN model with type of feature-based models covered in the previous chapter, all of which share common feature sets while training. Comparing this with an ML methodology trained solely from raw data would be interesting. My second goal is to combine the two ML methodologies, i.e., leveraging information both from engineered features (using a simple ML model) and from features extracted automatically from raw data by a CNN. The aim is to determine if this combined approach can address the overforecasting issue observed not only in other implementations, but also in previously studied models in this thesis.

As a choice of CNN architecture, I use VGG-16 — a standard convolutional neural network used for classifying images of everyday objects — and modify it for the flare prediction problem. I first experiment with multiple variants of this architecture that process the input data in different ways, and analyse the performance over various splits. I show that the best-performing VGG-16 model is on par with one of the feature-based ML models from Chapter 4, when optimized on

the True Skill Statistic Score (TSS). Additionally, as with the feature engineering results in the previous chapter, I show that this model also overforecasts, that is, produce many false positives during prediction. To address this, I design a novel approach that combines the VGG-16 model with an extremely randomized trees (ERT) model trained on the physics-based and topological features discussed in Chapter 3. This hybrid model is designed in two stages, the flaring probability output of the first VGG-16 stage is combined with the engineered features to train the ERT model in the second stage. After optimizing this combined architecture, I demonstrate its effectiveness in lowering the false positives. As an additional effort towards that goal, I also propose a new metric to replace the standard True Skill Statistic (TSS) for optimizing my models.

The chapter is outlined as follows. I first describe the dataset used for these set of experiments in Section 5.1. Because the CNN-based models are computationally intensive, I downsample the complete dataset of 450,000 samples by a factor of three to shorten the experimentation time. From this dataset, I extract values for the topological and SHARPs feature sets which are used for training the hybrid ML architecture. These are discussed in Section 5.2. Next, I describe in detail the VGG-16 model and its architecture variants, and choose the best performing variant for a final systematic evaluation in Section 5.3. I also introduce a hybrid two-stage ML model for flare prediction that includes the best performing VGG-16 variant as a first stage model, followed by an extremely randomized trees (ERT) model in the second stage. As mentioned, my goal here is to compare it to the first-stage VGG-16 model alone. I next discuss the metrics used in this chapter for comparing the two models in Section 5.4. I also introduce a new metric for tuning the two models, called $\text{TSS}_{\text{scaled}}$, which I claim is more useful than the standard TSS score for reducing false positives. Before comparing VGG-16 alone and the VGG-16+ERT combination, both models needs to be hyperparameter-tuned first. I discuss the hyperparameter tuning process in Section 5.5, also demonstrating the advantage of the $\text{TSS}_{\text{scaled}}$ metric. Post tuning, I show the model performance for the VGG-16 and the hybrid models, specifically focusing on the balance between the true positives (TP) and false positives (FP) scores for each case in Section 5.6, and conclude my findings in Section 5.7.

5.1 Data

For evaluating the CNN-based models, I use a dataset similar to the one described in Chapter 3, but with a couple of changes. That dataset included magnetogram images in each active region at a cadence of 1 hour, resulting in about approximately 450,000 images. For training deep learning models like VGG-16, such a huge dataset can lead to high computational costs. To make the computation more manageable, I reduce the cadence to 3 hours, resulting in a dataset of 157095 images. The second deviation from the previous dataset is the labeling scheme. In both cases, each magnetogram is labeled as flaring if an M1.0+ class flare is produced by the corresponding active region in the next k hours, else it is labeled as non-flaring. In the previous chapter, I used $k = 24$, a popular choice in the ML flare prediction literature. For my CNN studies, I change this to $k = 12$ with the intention of developing short-term forecasting. Since major flares are rare, this labeling, as before, results in an extremely imbalanced dataset. Here, 1561 ($\approx$1%) of the total magnetograms are labeled positive (flaring), and the rest of the 99% are labeled negative.

To train and evaluate the different architectures, I split the dataset into 70% training, 10% validation and 20% testing samples. As in the experiments reported in the previous chapter, the splitting is based on the active region number, so that all images of any given active region are not split between sets. The splitting is repeated 10 times using 10 random seeds; the model is trained, tuned and tested one randomized dataset at a time. The statistics of the performance score are reported across these 10 trials. With this arrangement, the total number of samples in the testing set is approximately 24000. The positive and negative samples for the individual splits are shown in Table 5.6, demonstrating the imbalance across all the splits.

One way of handling dataset imbalance is by data augmentation, which has been used successfully in image classification research [52]. Existing CNN flare prediction models [1, 44, 61, 78, 107, 108] create balanced datasets for training and evaluating the models, either by undersampling the majority class (lower intensity or no-flares) or oversampling the minority class (higher intensity flares) for both the training and testing sets. However, this strategy, while informative, is

not useful from an operational standpoint; measuring the performance on an artificially-balanced testing set is not representative of performance on real-world data, which is highly imbalanced in favor of negative (non-flaring) samples. For example, given a model with a certain false positive rate, the absolute number of false positives increases significantly when changing the model testing from a balanced to an imbalanced test dataset [1] . This leads to a significant discrepancy between the balanced versus imbalanced scenarios especially for metrics associated with false positives. On my dataset, I find that balancing only the training set using augmentation did not improve model performance score on a imbalanced testing set. Accordingly, I do not include data augmentation in any further experiments.

Finally, the magnetogram image dataset is not directly usable as-is with deep learning models. Each image cut-out is variable in size, whereas convolutional neural networks require fixed input dimensions across the entire dataset. There are multiple ways to transform all images to a standard size; in this work, I choose to perform affine transformations [2] : linear transformations that preserve lines and parallelism in an image, but not distances. I experimented with three fixed dimensions for the final image size: 64×64, 128×128 and 256×256. The motivation behind these choices was to ensure equal length on both axes, so as to not stretch the data unnecessarily along one of the dimensions (over the other). Choosing dimensions higher than these also led to out-of-memory errors on our setup. On each of these input configurations, I trained and evaluated individual VGG-16 models, finding that 128×128 is the smallest set of dimensions that require the least memory and processing time without affecting the quality of predictions. I therefore transform all images to 128×128 before using them in all experiments described in this chapter.

5.2 Feature sets

I use the SDO HMI radial field (B_r) magnetograms as a direct input to the CNN model studied in this chapter, and as a source to extract features that are used to train the extremely

[1] false positives = false positive rate $\times$ negative samples
[2] using the standard `OpenCV` package

randomized trees (ERT) model. Here, I use a combination of the same two feature sets that were introduced in Chapter 3 and evaluated in Chapter 4 — the physics-based SHARPs features and the newly introduced topological features — with some slight changes, as summarized in the following section.

5.2.1 Physics-based Features

The first feature set is the standard attributes of an active region cut-out that are available in the metadata of the SDO/HMI SHARPs dataset. These attributes, such as the area, total magnetic flux, magnetic shear, total vertical current, current helicity, etc., are predominantly derived from the physics-based properties of the vector magnetic field in a given magnetogram image. A complete list of these features is available in Table 5.1. This is the same SHARPs feature set introduced earlier in Table 3.2, but with the addition of two new features — `SIZE_ACR` and `SIZE`. I excluded these in my previous studies because I did not expect the size of the AR to impact the flaring probability. Here, I include them for the sake of completeness; however, I show later in this chapter that these two features are indeed not highly correlated with flaring.

5.2.2 Shape-based Features

To complement the physics-based features, I also include the shape-based features extracted using topological data analysis (TDA), described in 3.3. These features measure the number of two-dimensional "holes" in a sub-level thresholded magnetogram at different threhold levels of magnetic flux. In this chapter, I choose equally spaced magnitudes of magnetic flux thresholds for the positive and negative fluxes,

$$thresholds = \{20G,\ 420G,\ 820G,\ 1220G,\ 1620G,\ 2020G,\ 2420G\}.$$

These threshold levels are determined, after some experimentation, to give the best performance of the hybrid ML model. This gives a total of 14 TDA-based features, denoted by `flux_pos_t` and `flux_neg_t`, where $t \in thresholds$.

Acronym	Description	Units
LAT_FWT	Latitude of the flux-weighted center of active pixels	$degrees$
LON_FWT	Longitude of the flux-weighted center of active pixels	$degrees$
AREA_ACR	Line-of-sight field active pixel area	MH
USFLUX	Total unsigned flux	Mx
MEANGAM	Mean inclination angle, gamma	$degrees$
MEANGBT	Mean value of the total field gradient	G/Mm
MEANGBZ	Mean value of the vertical field gradient	G/Mm
MEANGBH	Mean value of the horizontal field gradient	G/Mm
MEANJZD	Mean vertical current density	mA/m^2
TOTUSJZ	Total unsigned vertical current	A
MEANALP	Total twist parameter, alpha	$1/Mm$
MEANJZH	Mean current helicity	G^2/m
TOTUSJH	Total unsigned current helicity	G^2/m
ABSNJZH	Absolute value of the net current helicity	G^2/m
SAVNCPP	Sum of the absolute value of the net currents per polarity	A
MEANPOT	Mean photospheric excess magnetic energy density	$ergs/cm^3$
TOTPOT	Total photospheric magnetic energy density	$ergs/cm^3$
MEANSHR	Mean shear angle (measured using B_{total})	$degrees$
SHRGT45	Percentage of pixels with a mean shear angle greater than 45 degrees	$percent$
R_VALUE	Sum of flux near polarity inversion line	G
NACR	The number of strong LOS magnetic field pixels in the patch	N/A
SIZE_ACR	Projected area of active pixels on image	MH
SIZE	Projected area of patch on image	MH

Table 5.1: The SHARPs feature set, as available in the metadata of the SDO HMI dataset. Abbreviations: Mx is Maxwells, G is Gauss, Mm is Megameters, A is Amperes and MH is microhemispheres. Most of these features are discussed in [10], while the last two are described in [20].

5.3 A Two-stage Machine Learning Pipeline

The baseline machine learning model in this chapter is a convolutional neural network (CNN). CNNs are an effective tool for learning patterns from images which can help them automatically classify the image content. This is advantageous as a way to avoid manual feature engineering of the dataset, or alternatively as a way to complement these manually-engineered features. Here I aim to combine the predictive power of the manually-engineered features (SHARPs and topological) of the magnetograms with the predictive power of features automatically extracted by a CNN model. To do so, I propose a two-stage model architecture. The first stage is a CNN architecture that is trained on the magnetogram images directly and outputs a flaring probability. This flaring probability is then used as a feature, along with the manually-engineered features, to train an ERT. I choose this design on account of its simplicity as well as its ability to separate the prediction capabilities of the CNN features and the engineered features. An additional benefit of ERTs is their ability to rank the relevance of various features in terms of predicting flares. I discuss the two stages below.

5.3.1 Stage I: Convolutional Neural Network

The first stage of my hybrid model is a VGG-16 model [89] — a deep CNN designed to classify images into 1000 pre-defined categories. This model was developed for classifying everyday images from ImageNet[27], a dataset that contains 14-million high resolution images of everyday objects belonging to over 1000 categories. Here, I modify the the VGG-16 architecture, an example of which is shown in Fig 5.1, to classify magnetogram images into two classes — flaring and non-flaring.[3]

The input sample is not a single RGB image, but can either be a vector magnetogram image ($[B_r, B_\phi, B_\theta]$), the radial component of the magnetogram (B_r), or a temporal stack of four consecutive magnetograms separated by a cadence of three hours ($[B_{r,t}, B_{r,t-3}, B_{r,t-6}, B_{r,t-9}]$), depending on the selected configuration (described below). Accordingly, the input layer of the VGG-16 model

[3] One could alternatively start with a pretrained VGG-16 model, i.e. a model with pre-assigned weights learnt from training on the ImageNet database, and fine tune those weights by additional training on the magnetogram dataset. I believe, however, that the weights are not necessarily transferrable since the fundamental patterns of everyday objects are different from magnetogram patterns, so I use the VGG-16 architecture without any pretrained weights.

is modified to have different number of channels (3, 1, or 1/4 respectively) instead of the three needed for standard RGB images. As an example, let us consider an architecture designed for an input of four channels (this will end up being the final configuration we use, as will become clear later). The first part of the model consists of 13 2D convolution layers, each designed with a filter size of 3×3 pixels and a one-pixel padding. These convolution layers are interspersed with five max-pooling layers, with the pooling performed over a window of 2×2 pixels. Each convolution layer operation preserves the size of that map, while each max-pooling layer halves the size of the feature map. The stack of convolution layers is followed by a series of three fully connected layers. The first two of these contain 4096 output channels each. In the original VGG-16 architecture, this is followed by a third layer with 1000 output channels (for the 1000 categories of ImageNet). I modify this to have just two output channels instead—for the flaring and non-flaring classes, in accordance with the described labeling scheme. A final softmax layer normalizes the output of the two channels to take values between $[0, 1]$, representing the probability of flaring and non-flaring in the next k hours for a given input sample, where k is the forecast window. For this study I set $k = 12$, since I am interested in developing short-term forecasts 12 hours into the future.

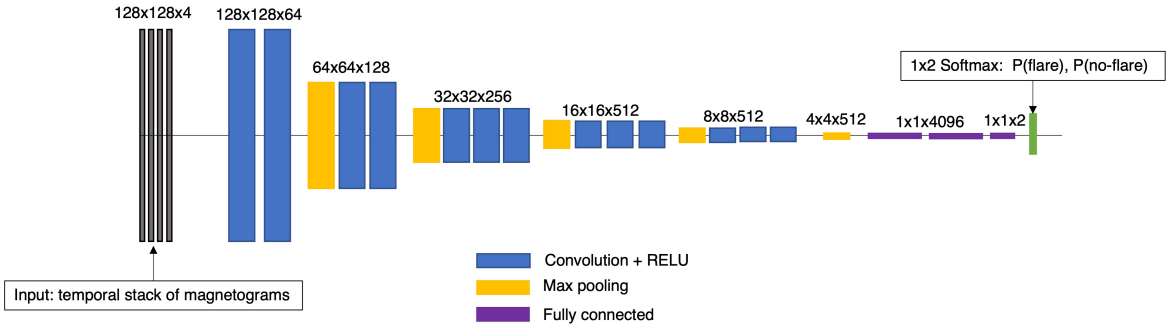


Figure 5.1: The VGG-16 architecture, as adapted for the solar flare prediction problem. In this figure, the input to the VGG-16 model is a temporal stack of four magnetograms. For other experiments, this is changed accordingly. The output of the model is two complementary neurons representing the flaring and non-flaring probability of the input sample.

I investigate four variations on the model architecture and the input data format.

(1) C_1: Using an input stack consisting of the $[B_r, B_\phi, B_\theta]$ components of the vector magne-

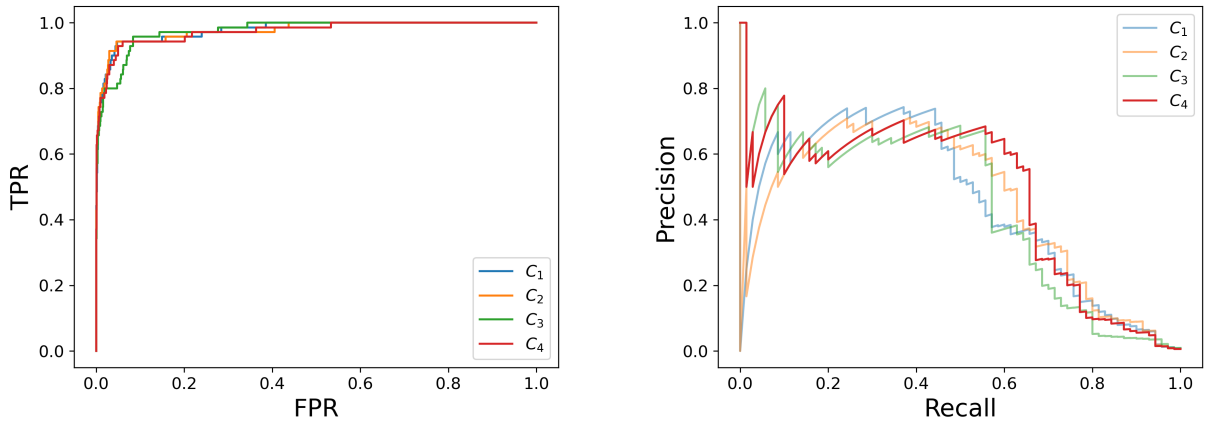
togram and setting the number of input channels of the VGG-16 model to three.

(2) C_2: Using a single component B_r at a single time instance as the input data sample, setting the number of input channels of the VGG-16 model to one.

(3) C_3: Using the temporal stacked configuration $[B_{r,t}, B_{r,t-3}, B_{r,t-6}, B_{r,t-9}]$ as the input data per sample (as described above), and setting the number of input channels of the VGG-16 model to 1. Each component in the temporal stack is operated on by the convolutions and dense layers individually to generate four feature representations. An LSTM layer is introduced before the output layer to process the sequence of the four representations. The input configuration is heuristically chosen after experimenting with a few configurations of different sequence lengths and temporal spacing, and determining the best performing configuration in terms of the model validation score. Given enough computational resources, an optimal configuration can be determined instead through a more rigorous hyperaparameter tuning process (such as the one demonstrated in Chapter 4).

(4) C_4: Using the temporal stacked configuration $[B_{r,t}, B_{r,t-3}, B_{r,t-6}, B_{r,t-9}]$ as the input data per sample (as described above), and setting the number of input channels of the VGG-16 model to four. All components of the temporal stack are treated as individual channels in the input layer. Such a setup leads to a single feature representation at the final layer that is acted upon by the softmax function, as opposed to the four representations generated from the temporal stack in C_3.

To evaluate these four configurations, I choose a single dataset splitting configuration of magnetogram images based on their timestamp: all magnetograms recorded from years 2010-2014 are assigned to the training set, magnetograms from 2015 are used for validation, while magnetograms in the years 2016-2017 are allocated to the testing set. The rationale behind using a single split is to provide a quick first-cut analysis of the architectures without incurring a high computational cost of a thorough one (as we do with multiple random seeds later). All four configurations are modeled



(a) ROC AUC

(b) PR AUC

Figure 5.2: ROC AUC and PR AUC curves reported for the performance of the four VGG-16 configurations on the data test set.

in `Pytorch`, using a weighted focal loss function ($\alpha = \frac{1}{11}$, $\gamma = 2$) [63] with an additional L_2 regularization weight decay factor $\beta = 0.001$. The weights of the models are updated with an `Adagrad` optimizer [33], using an initial learning rate of 0.0001 and a batch size of 64. A cosine annealing learning rate scheduler is used for adjusting the learning rate through 10 epochs of training. Values for these parameters are chosen through hand-tuning such that the validation set performance is optimal.

For each input sample, all configurations output the probability of flaring in the next 12 hours. For a categorical output, a threshold should be applied to the output to categorize the input in a binary fashion (flaring/non-flaring). This is especially useful when comparing the performance with another model that also has a categorical output, using metrics discussed in Section 5.4. However, in this case, since all the model configurations are probabilistic, I choose two metrics that are independent of the final choice of threshold for comparing the performance. First is the area under the curve (AUC) of the Receiver Operating Characteristic Curve (ROC) [45]. ROC plots are generated by varying the probability threshold between 0 and 1, and plotting the true positive rate (TPR) versus the false positive rate (FPR). For a given threshold, the former represents the

Configuration	ROC AUC	PR AUC
C_1: $[B_r, B_\phi, B_\theta]$	0.967	0.43
C_2: B_r	0.965	0.43
C_3: B_r stack w/LSTM	0.975	0.43
C_4: B_r stack as channels	0.974	0.46

Table 5.2: Performance of the VGG-16 model variants discussed in Section 5.3.

fraction of the positive samples correctly classified, while the latter represents the portion of negative samples incorrectly classified (formulae for both are provided in Table 5.3). This generates a curve in the 2D plane. The AUC of this ROC plot is a value between 0 and 1 (1 indicating a perfect model) and thus represents the summarized performance of the model over various thresholds. The second metric is the Precision-Recall AUC (PR AUC)[12], generated by determining the area under the curve of the Precision versus Recall plot for the range of thresholds between 0 and 1. Here, Recall is the same as the TPR, whereas Precision is the fraction of correct predictions amongst all the samples marked as positive (definitions provided in Table 5.3). PR AUC also varies between 0 and 1, just like ROC AUC, with 1 indicating the most predictive model. The ROC and PR curves for the four different configurations are shown in Fig. 5.2, while the corresponding AUC metrics are reported in Table 5.2.

Looking at the ROC plots in Fig. 5.2(a), the profiles for all configurations are pretty close to each other, making it hard to determine the best performing one. This is also reflected in the ROC AUC values in Table 5.2, which are more or less the same. However, the PR curve for configuration C_4 is, on average, located above others, especially for recall values below 0.2 and above 0.5 approximately. This is reflected in the PR AUC score for C_4, which is higher than others. This makes C_4 a more suitable choice for the final VGG-16 architecture. These results indicate that using all three components of the magnetic field (B_r, B_ϕ, B_θ) is just as better as using the B_r component alone. Additionally, between C_3 and C_4, both of which use a temporal sequence of four magnetograms as input, the LSTM configuration C_3 performs worse than using the sequence as individual channels of the CNN input layer as in C_4. A probable reason for this is the short sequence of only four data samples being used for training the LSTM model, suggesting that treating them

as channels is better. Note that increasing the sequence length to better train the LSTM would require more computation.

Using the AUC metrics above was an acceptable approach since all models being compared were generating probabilistic forecasts. However, for the purposes of comparison with the hybrid model architecture proposed next, which produces categorical flare/no-flare event predictions, I convert the probabilistic output into a categorical output by defining an optimal flare event threshold. Some methods choose an arbitrary threshold of 0.5 to do this [57] (as I did in Chapter 4) while others use an automatically-determined threshold based on the Receiver Operating Characteristic by determining where the ROC curve is furthest from the diagonal. [4] . Here, I choose a different approach: I convert the threshold to a hyperparameter that can be optimized using a validation set. This is a useful strategy for CNN tuning and evaluation. Instead of going through a time-consuming process of tuning numerous hyperparameters such as the regularization weight decay and the loss weights, I fix them to reasonable values (with some hand-tuning), and determine an ideal threshold for the chosen configuration that optimizes certain metrics.

5.3.2 Stage II: Extremely Randomized Trees (ERT) model

While there have been CNN implementations for solar-flare forecasting proposed in recent years, the novelty in my approach is the combined use of the features extracted through convolutions together with engineered features based on the physics and the shape of the magnetogram, which are covered in Section 5.2. To explore this, I generate a feature set that concatenates the output of the VGG-16 model (the probability of a flare) with these engineered features (20 SHARPs and 14 TDA) to obtain a combined feature set with 35 elements.

I use this feature set to train an extremely randomized trees (ERT) model [37] in the second stage. An ERT (introduced earlier in Section 4.1) is a tree-based model built as a hierarchical structure of nodes that successively perform the operation of separating the dataset into two classes. The entire dataset is "fed" to the root of this tree and undergoes a sequence of splitting operations

[4] Logically, this coincides with the threshold that maximizes the TSS score.

at the intermediate nodes. At each node, the incoming dataset is separated into two subsets — termed "left" and "right" — based on a feature thresholding criterion. That is, a random subset of m features is chosen from the entire candidate feature set, together with m random splits (one for each feature). For each split at node t—let us call it s_t—the quality (of that split) is determined by computing the reduction in some "impurity" metric of the dataset given by —

$$\Delta i(s_t, t) = i(t) - p_L \times i(t_L) - p_R \times i(t_R). \tag{5.1}$$

Here, the impurity function $i(t)$ quantifies the degree of class intermixing for a given dataset input into node t. Correspondingly, the impurities for the left and the right subsets from the split are denoted by $i(t_L)$ and $i(t_R)$ respectively. $p_L = N_{t_L}/N_t$ and $p_R = N_{t_R}/N_t$ represent the proportions of the dataset arriving at node n of size N_t, split into the left (size N_{t_L}) subset and right subset (size N_{t_R}) respectively. Of the m splits, the one that maximizes $\Delta i(s_t, t)$ is chosen. For the definition of impurity, I use the standard Gini impurity index, as described in [81]. In the context of our setup, if $N_{t,+}$ and $N_{t,-}$ are the number of positive and negative samples arriving at node t, the Gini impurity index is calculated as

$$i(t) = 1 - \left(\frac{N_{t,+}}{N_t}\right)^2 - \left(\frac{N_{t,-}}{N_t}\right)^2. \tag{5.2}$$

Whether the two subsets are further subject to splitting at the next level is determined by an important hyperparameter in the training process known as the `min_impurity_decrease_index`, denoted by Δi_{min}. A dataset at any point in the tree is split further using an additional node if

$$\Delta i(s_t, t) \geq \Delta i_{min}.$$

In the context of this problem, Δi_{min} determines how the model balances between the true positive rate (TPR) and false positive rate (FPR, also called the False Alarm Rate), which are the fraction of positive samples correctly and incorrectly classified respectively by the model. A low value of Δi_{min} results in low FPR but a low TPR as well, whereas a high Δi_{min} raises the TPR at the cost of increased FPR as well. Just as with the threshold in the CNN stage, I tune this hyperparameter using a validation set (discussed in Section 5.5).

The two-stage model is summarized in Fig. 5.3. In the context of this two-stage model, I hereon denote the VGG-16 model as *CNN-Only* and the ERT stage as *CNN+ERT* (since it uses the VGG-16 probability output as a feature).

5.4 Metrics

With a categorical forecast (flare/no-flare), we can compute the standard confusion matrix on the testing set predictions: true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN). Using the entries of this matrix, one can define various metrics, some of which appear in Table 5.3. Most of these metrics are standard in the flare prediction literature. A popular one among these is the true skill statistic score (TSS), equal to the difference TPR − FPR, where true positive rate (TPR) is the fraction of TP among all positive samples, and the false positive rate (FPR) is the fraction of FP among all negative samples. TSS provides some utility to the flare prediction problem because it is insensitive to dataset imbalance, and is a better indicator of the model performance than the standard accuracy [6, 9]. However, optimizing the TSS score often leads to overforecasting in the model; models optimized on TSS tend to improve TPR at the cost of also slightly increasing the FPR. A slight increase in FPR, however, can lead to a significant increase in the absolute false positives FP, since the number of negative samples is huge, thereby impacting other metrics like precision or F1, that are sensitive to FP.

To address this problem, I define a new metric for model optimization, $\mathrm{TSS}_{\mathrm{scaled}}$, given by

$$\mathrm{TSS}_{\mathrm{scaled}} = \mathrm{TPR} - \frac{\mathrm{TPR}_{\mathrm{max}}}{\mathrm{FPR}_{\mathrm{max}}}\mathrm{FPR}, \tag{5.3}$$

where $\mathrm{TPR}_{\mathrm{max}}$ and $\mathrm{FPR}_{\mathrm{max}}$ are the maximum values of the TPR and FPR, respectively, determined over the range of model hyperparameters. The scaling factor $\frac{\mathrm{TPR}_{\mathrm{max}}}{\mathrm{FPR}_{\mathrm{max}}}$ applied to the FPR term is a quantity greater than 1 for my hyperparameter choices; it is designed to penalize increase in false positives more than the TSS score does. As I will show in the next section, this provides a better balance between TPR and FPR.

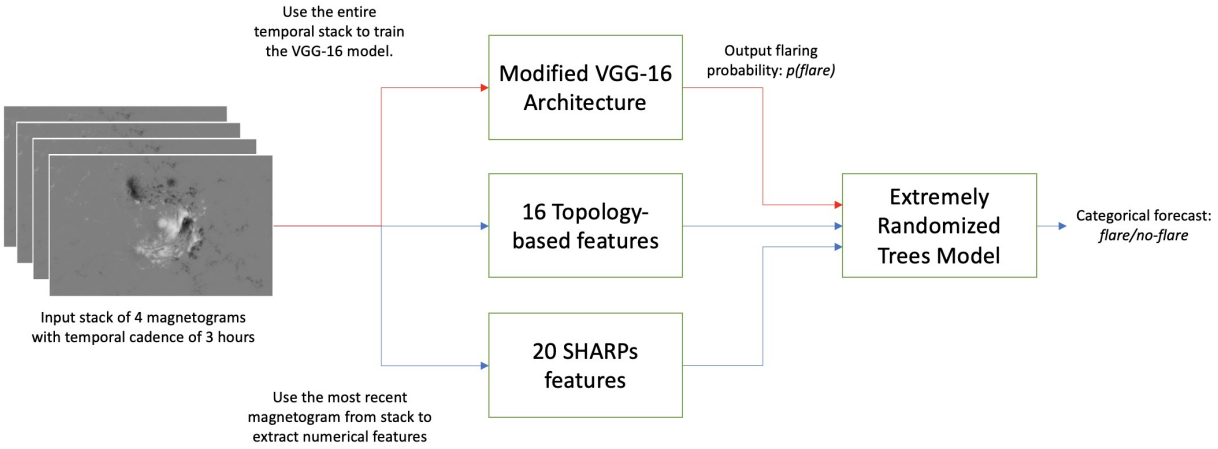
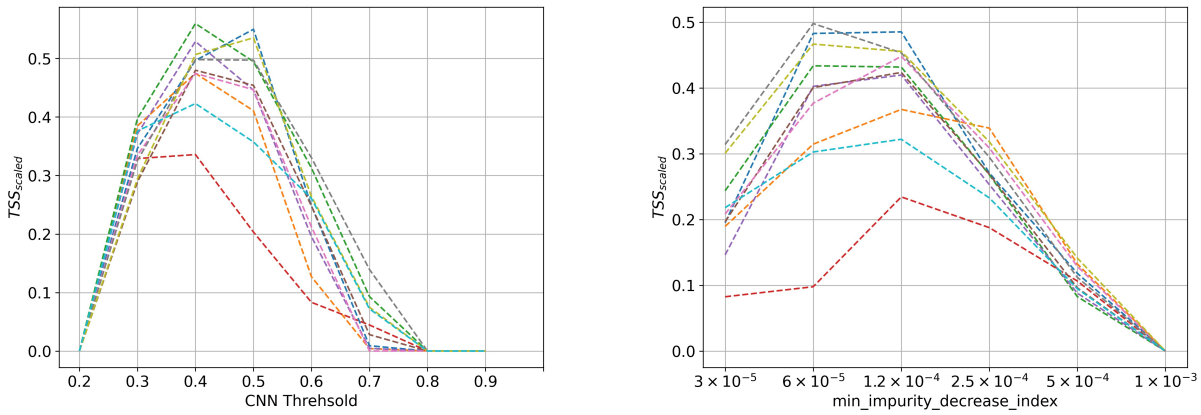


Figure 5.3: My two-stage model for solar flare prediction. The input is a temporal stack of B_r magnetograms from SDO/HMI which is both fed to a custom CNN model and analyzed for feature vectors. The CNN model outputs the probability of flaring with the 12-hour forecast window and this probability is combined with the feature vectors to create a single feature vector input to the ERT model. The output of the ERT model is a binary event prediction.

Metric	Formula
Recall (TPR)	$\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$
False Positive/Alarm Rate (FPR)	$\frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}}$
Accuracy	$\frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}$
Precision	$\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$
True Skill Statistic (TSS)	$\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} - \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}} = \mathrm{TPR} - \mathrm{FPR}$
Heidke Skill Score (HSS_2)	$\frac{2(\mathrm{TP}\times\mathrm{TN}-\mathrm{FP}\times\mathrm{FN})}{(\mathrm{TP}+\mathrm{FP})(\mathrm{FP}+\mathrm{TN})+(\mathrm{TP}+\mathrm{FN})(\mathrm{FN}+\mathrm{TN})}$

Table 5.3: Metrics used for evaluating the binary forecasting models.

5.5 Hyperparameter Tuning



(a) Stage-1 CNN-only

(b) Stage-2 CNN+ERT

Figure 5.4: Hyperparameter tuning across multiple seeds for both stages of the hybrid flare prediction model. In both stages, hyperparameters that optimize the $\text{TSS}_{\text{scaled}}$ metric are determined.

Hyperparameter tuning is essential for optimizing machine leaning model performance. For both the stages, I carried out simple hand-tuning experiments to identify the hyperparameters that significantly affect the TPR-FPR balance. In the case of the first stage CNN-only model, it is the threshold for converting probabilistic to categorical forecast. This is obvious, since the threshold decides exactly the cut-off point for labeling a probability prediction as flaring or non-flaring; any prediction with a probability greater than the threshold is labeled as flaring. For a threshold of 0, all samples are labeled as flaring, making both TPR = 1 and FPR = 1. For a threshold of 1, all samples are labeled as non-flaring, making both entities equal to zero. Since an ideal configuration requires TPR = 1 and FPR = 0, neither of these extreme cases are optimal, and an intermediate threshold needs to be selected. In the case of the second stage ERT model, it is the `min_impurity_decrease_index` parameter. Low values of this parameter leads to lower FPR and TPR values, while raising the value of the parameter increases both TPR and FPR. This is not as intuitive in terms of explanation as the CNN threshold. A possible explanation is that, for higher values of `min_impurity_decrease_index`, less splitting occurs, causing the model to

clump many non-flaring samples with flaring-like characteristics together with the flaring samples, and mark them as flaring. This configuration implies accurate prediction of the positive class, but misprediction of the negative class, thus resulting in high TPR and FPR. On reducing the `min_impurity_decrease_index`, the model performs further splitting, breaking up this set into two subsets. The splitting separates out the majority of the non-flaring samples from the flaring ones (reducing the FPR), but also inaccurately mispredicting some of the flares as non-flares (reducing the TPR).

The reader might recall that I used an automated systematic hyperparameter tuning process for optimizing the ML models in Chapter 4. This was possible because those models were relatively simpler (fewer trainable parameters) than VGG-16. For example, the most complex LSTM model (which took days to optimize and train), contained 7382 trainable parameters, compared to millions of parameters in the standard VGG-16 model. The computational cost of automatically tuning the VGG-16 model is thus significantly higher, even with a reduced dataset. To work around this, I implement a simpler hand-tuning algorithm in work reported in this chapter. Doing a grid search over the hyperparameter space, I determine the optimal hyperparameter that gives the best results on a single validation set (as opposed to k-fold cross-validation), and then use it to evaluate the models on the testing set.

This process is performed individually on the VGG-16 and ERT. For the VGG-16 stage, the only hyperparameter is the threshold used for converting a probabilistic prediction output into a categorical one. To tune the model, the VGG-16 model is trained once, and probabilistic outputs are generated for all samples in the validation set. The threshold hyperparameter is then applied to convert all probabilistic outputs to binary outputs for the entire validation set, which can then be used to quantify the model performance in terms of categorical forecast metrics discussed in Section 5.4. I study two of these metrics — TSS and $\text{TSS}_{\text{scaled}}$ — over eight evenly-spaced thresholds in the interval $[0.2, 0.9]$. The final choice of the threshold hyperparameter is one that maximizes the metric of choice on the validation set. In case of the ERT model in the second stage, the single important hyperparameter is the `min_impurity_decrease_index`. Unlike the VGG-16

model, the ERT has to be trained once for every hyperparameter choice in the set before it is evaluated on the validation set. Six values of `min_impurity_decrease_index` are chosen for tuning — $[3 \times 10^{-5}, 6 \times 10^{-5}, 1.2 \times 10^{-4}, 2.5 \times 10^{-4}, 5 \times 10^{-4}, 1 \times 10^{-3}]$. The value that optimizes a chosen metric is the used as the final model setting.

I iterate this procedure for both stages over 10 dataset splits obtained from 10 random seeds. For both stages, I perform tuning primarily to optimize the newly proposed metric — $\text{TSS}_{\text{scaled}}$ — and compare it with the model tuned on the standard TSS score. The results for the $\text{TSS}_{\text{scaled}}$-tuned model are shown in Fig. 5.4. From the figure, it is clear that for the majority of the dataset combinations, a single value of the hyperparameter maximizes $\text{TSS}_{\text{scaled}}$. For the first stage, this is a threshold of 0.4; for the second, `min_impurity_decrease_index`= 0.00012. Performing a similar optimization study for the TSS score instead yields an optimal threshold of 0.3 for stage 1 and an optimal `min_impurity_decrease_index`= 0.001. The difference in hyperparameter values is in accordance with the preference of TSS-optimized and $\text{TSS}_{\text{scaled}}$-optimized configurations: TSS-optimized models tend towards higher TPR and FPR values than the $\text{TSS}_{\text{scaled}}$ models. As explained above, high TPR/FPR values occur at lower value settings of the threshold in the CNN model, and for higher values of `min_impurity_decrease_index` in the ERT model.

Model stage	Optimized metric	Optimal hyperparameter	TPR	FPR
CNN-only	TSS	threshold = 0.3	0.90 ± 0.05	0.13 ± 0.02
CNN-only	$\text{TSS}_{\text{scaled}}$	threshold = 0.4	0.75 ± 0.09	0.06 ± 0.01
CNN+ERT	TSS	$\Delta i_{min} = 0.001$	0.90 ± 0.05	0.12 ± 0.02
CNN+ERT	$\text{TSS}_{\text{scaled}}$	$\Delta i_{min} = 0.00012$	0.65 ± 0.11	0.03 ± 0.01

Table 5.4: The mean and standard deviation values of TPR and FPR on the validation set (10% of the full dataset or $\approx 15,000$ samples) for the two models. The statistics are shown separately using TSS and $\text{TSS}_{\text{scaled}}$ metrics for optimizing hyperparameters of these models. The statistics are generated over 10 trials with 10 different random dataset splits. The CNN+ERT hyperparameter `min_impurity_decrease_index` is denoted by Δi_{min}.

The TPR and FPR statistics on the validation set across the 10 trials for each of the these hyperparameter choices are shown in Table 5.4. It can be seen that optimizing $\text{TSS}_{\text{scaled}}$ rather than TSS on average reduces FPR by a factor of approximately 2-4 while only reducing the TPR

Model	Accuracy	TSS	HSS_2	F_1	Bias
Logistic regression	0.89 ± 0.02	0.82 ± 0.01	0.10 ± 0.03	0.11 ± 0.03	17.13 ± 4.10
VGG-16	0.87 ± 0.01	0.77 ± 0.04	0.11 ± 0.03	0.12 ± 0.04	14.78 ± 4.36

Table 5.5: Comparison of the VGG-16 and the Logistic regression models for a 12-hour flare prediction problem. While the accuracy and the TSS are slightly worse, other metrics such as HSS_2, F_1 and Bias are better in the VGG-16 model.

by a factor of approximately 1.2-1.4. In other words, hyperparameters using TSS_{scaled} offer a more favorable result in terms of the TPR-FPR balance, so I use it henceforth in this thesis (i.e., threshold = 0.40 for the first stage, `min_impurity_decrease_index`= 0.00012 for the second).

5.6 Results

5.6.1 Comparison with feature-engineering based models

Before I discuss the performance of the VGG-16 model and the two-stage model, I provide a brief comparison of the VGG-16 model with the least complex model from Chapter 4: logistic regression. Since all of my models in that chapter were optimized for the TSS score, I re-optimize the VGG-16 model by tuning on the TSS metric as well, using a threshold of 0.30 (a threshold that maximizes TSS as discussed in Section 4.2), and compare the results against those of logistic regression for the 12-hour forecasting problem provided in Table 4.3. The results are summarized in Table 5.5. Of all the metrics in the comparison, F_1 and Bias are computed according to the definitions in Chapter 4 [5] . It should be highlighted that the dataset and tuning methodology for these models is slightly different. The logistic regression model is trained on a much larger set, and hyperparameter-tuned with an automated algorithm. For reasons of high computational complexity, the VGG-16 model, on the other hand, is trained on one-third the dataset used by the former, and also hand-tuned. While the fairness of comparison here does not hold up to the standards of Chapter 4, it should give the reader a rough idea of where the two models stand relative to one another. Comparing the two rows in Table 5.5, it can be seen that the VGG-

[5] These are only used for comparison with logistic regression but are ignored for later analysis since they show trends that are correlated with HSS_2.

16 model has a slightly lower accuracy and TSS score than logistic regression. However, logistic regression does worse as measured by other metrics, such as HSS_2, F_1 and Bias. Overall, neither model significantly outperforms the other, which further reinforces my conclusions from the last chapter: high computation complexity does not necessarily provide better performance for the flare prediction problem. Secondly, the comparison indicates that features extracted from the convolution process of the VGG-16 model do not perform any better than the topological and SHARPs features engineered from the images. Finally, while one model seems to overforecast more than the other, both, on the whole, have a high degree of overforecasting, leading to low values of HSS_2 and F_1. In other words, the CNN model does not resolve the overforecasting issues identified in the previous chapter. I discuss methods to address this next.

5.6.2 Tackling overforecasting: CNN-Only versus CNN+ERT

My design for the two-stage hybrid model described in section 5.3, which combines a CNN and an ERT, was motivated by the need to reduce overforecasting. To evaluate its success, I now compare it with the VGG-16 model (CNN-Only) with the two-stage hybrid model (CNN+ERT), optimizing both models on the TSS_{scaled} metric to address the overforecasting problem discussed previously. Note that optimizing the CNN-Only model also optimizes the first stage of the CNN+ERT architecture; this can be alternatively considered as a comparison between two stages of the hybrid model. For the 10 dataset splits, I separately determine the confusion matrix on the test set for each of the two stages, and calculate the metrics discussed in Table 5.3. The optimal hyperparameters for each stage, as derived in Section 5.5, are used. I then evaluate the change in these various metrics between stages, i.e. using only the CNN and then appending the ERT to it. The raw values of six metrics are shown in Fig. 5.5 in the form of box plots. Looking at these two metrics in box plots Fig. 5.5(a) and (b), we observe that TPR is slightly decreased with the addition of the ERT stage to the model. On the other hand, the addition of the ERT reduces FPR significantly, thus reducing the overforecasting nature of the model. It should also be observed that the FPR box plots for the CNN-Only and CNN+ERT architecture are non-overlapping, demonstrating that

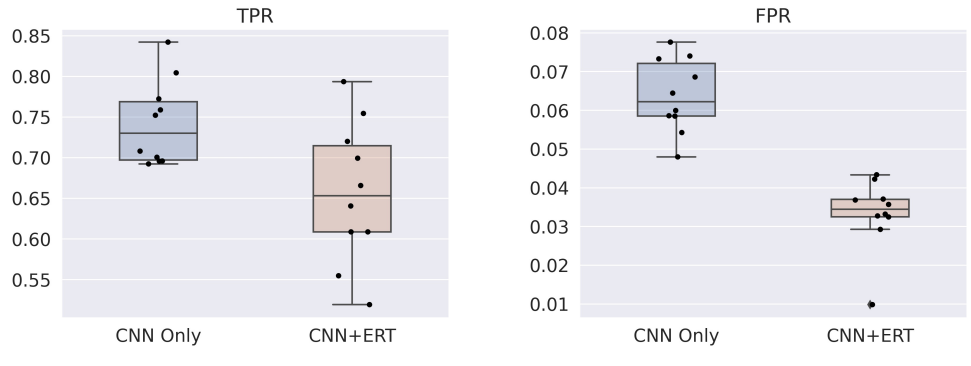
the improvement is significant. The TPR and FPR trends are encouraging, but these normalized metrics indicate a part of the story. Values of the raw confusion matrix (TP, TN, FP and FN) for each individual dataset split in Table 5.6 show a significant decrease in the FP values ($\approx$ 670 on average) with only a minor decrease in TP ($\approx$ 17 on average) for the CNN+ERT model. Further, the table also shows that this decrease is consistent across all dataset splits. These changes in TP and FP scores impact other metrics both positively and negatively. For example, the Precision and the HSS_2 score in Fig. 5.5(c) and (f) respectively are better overall for the CNN+ERT architecture, whereas the Recall and TSS are overall slightly worse due to the dominance of TPR in calculating these metrics (Fig. 5.5 d,e).

Table 5.7 shows the average percentage improvement across all the dataset splits, which summarizes the results in Fig. 5.5. The true positive rate is decreased (on average) by 12%, while the false positive rate improves by 48%. Again, this impacts the derived metrics in different ways. For example, the more popular TSS metric is decreased by an average of 8%, and similarly the recall is decreased by an average of 12%. On the other hand, we see very large improvements in precision ($\approx$69%) and HSS_2 ($\approx$56%). These results indicate that the two-stage model provides a better prediction, especially in terms of FP and associated metrics.

5.6.3 Feature Ranking

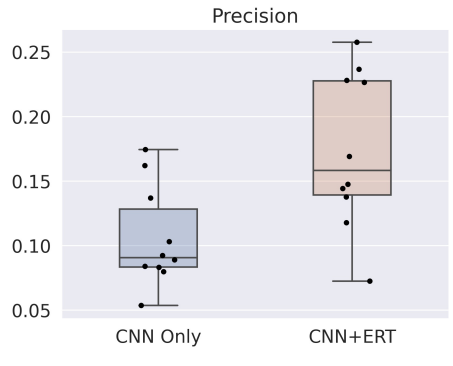
It is useful to know which features in a feature set play an important role in the model prediction. I use the Gini impurity index extracted from the ERT model to determine how much each feature is successful in separating the positive and negative labels across all the nodes of the tree. The relative rankings are shown in Fig. 5.6. While there are other ways for performing multivariate feature ranking, e.g. the linear discriminant analysis from [56] — leveraging the information that is implicit in our ERT prediction model to also perform feature ranking makes sense.

Fig. 5.6 shows a mix of features from the topological and SHARPs sets in the top-20 ranking features. The highest ranking feature, however, is the VGG-16 output probability `cnn_prob`, which

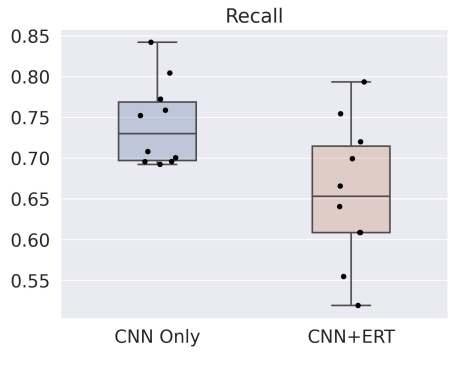


(a) TPR

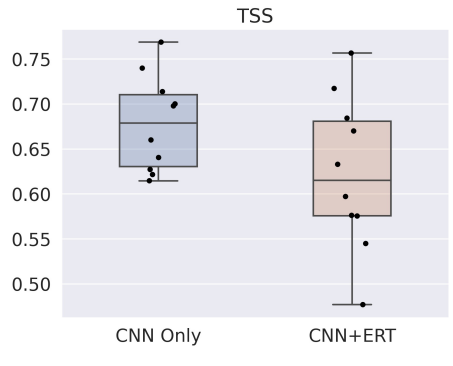
(b) FPR



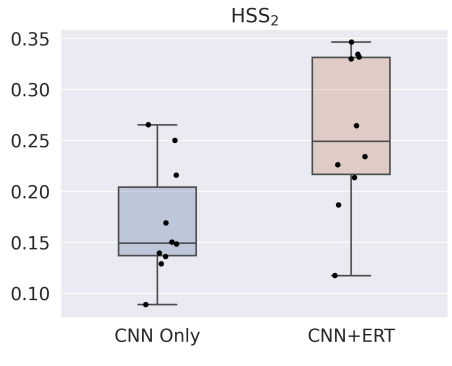
(c) Precision



(d) Recall



(e) TSS



(f) HSS_2

Figure 5.5: Performance comparison between CNN-Only and CNN+ERT models across six different metrics. For each metric boxplot, the 10 dots shown represent the individual score of each of the dataset splits.

Random seed	P	N	Architecture	TP	TN	FP	FN
Seed 100	286	23345	CNN-only	217	21977	1368	69
			CNN+ERT	200	22662	683	86
Seed 200	207	23768	CNN-only	145	22343	1425	62
			CNN+ERT	126	22979	789	81
Seed 300	137	22283	CNN-only	97	21214	1069	40
			CNN+ERT	76	22064	219	61
Seed 400	347	22760	CNN-only	261	21525	1235	86
			CNN+ERT	231	22015	745	116
Seed 500	228	22787	CNN-only	192	21117	1670	36
			CNN+ERT	172	21942	845	56
Seed 600	156	24532	CNN-only	108	22628	1904	48
			CNN+ERT	81	23496	1036	75
Seed 700	325	22187	CNN-only	251	20889	1298	74
			CNN+ERT	234	21395	792	91
Seed 800	217	24007	CNN-only	151	22360	1647	66
			CNN+ERT	139	22966	1041	78
Seed 900	207	22425	CNN-only	144	20765	1660	63
			CNN+ERT	126	21697	728	81
Seed 1000	184	23509	CNN-only	148	21994	1515	36
			CNN+ERT	146	22643	866	38

Table 5.6: Confusion matrix results for the CNN-Only and CNN+ERT architectures, shown for each of the 10 individual dataset splits. P and N represent the total positive and negative samples in the testing set of each split.

Metric	% average change in metric between stages
Recall (TPR)	-12 ± 6.9
False Positive/Alarm Rate (FPR)	-48 ± 12.4
Accuracy	3 ± 0.7
Precision	69 ± 16.7
TSS	-8 ± 7.0
HSS_2	56 ± 35.7

Table 5.7: Percent change in metrics of the performance of the two-stage model (CNN+ERT) over a single stage CNN-only model, along with the standard deviation, summarized over 10 dataset experiments.

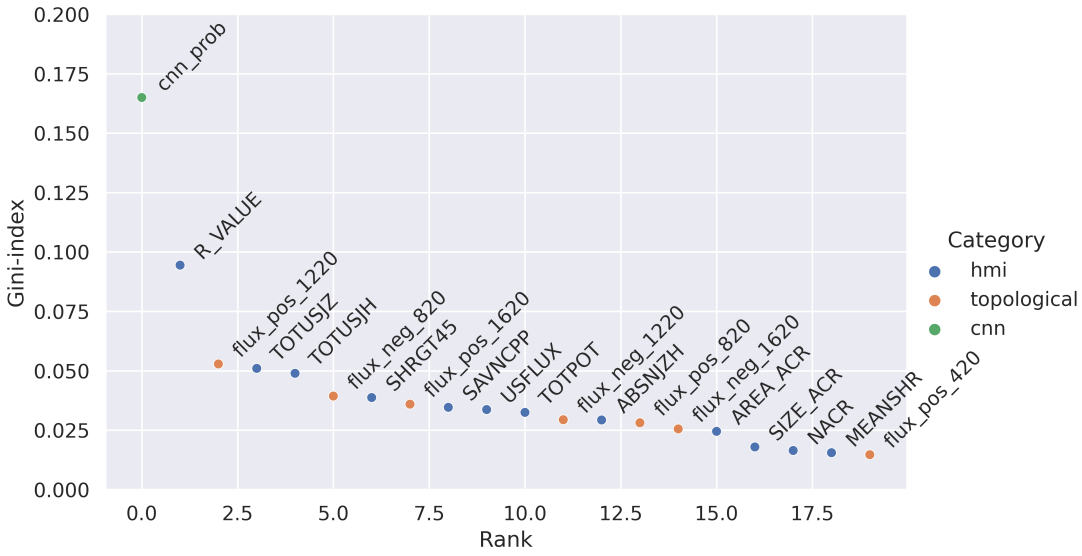


Figure 5.6: Feature ranking using the Gini impurity index from the ERT model for the top 20 features.

indicates that the VGG-16 by itself is able to discriminate a significant of flaring from non-flaring samples by learning patterns from the raw data. The R_VALUE feature from the HMI feature-set ranks highly (second only to the VGG-16 probability). This makes sense because this quantity corresponds to the magnetic flux from the neutral line where magnetic reconnection (and thus coronal mass ejections) occur. With regard to the topological feature set, the β_1 counts in the range of 800G to 1600G are shown to be important. Finally, it should be mentioned in passing that the two additional SHARPs parameter included in this chapter — SIZE and SIZE_ACR — do not rank highly in this analysis, so their inclusion does not obfuscate the comparison between the results described here and the feature-engineering based models in the previous chapter.

5.7 Summary

In recent years, quite a few CNN-based deep learning models have been applied to the flare-prediction problem. While these methods have achieved some success, extreme dataset imbalance in this problem remains a significant challenge in the development of these models, causing them to produce a high number of false positives in relation to true positives. While most of these models artificially balance the training and testing sets (by oversampling flares or undersampling non-flares), such an evaluation approach does not reflect a real-world scenario. To address this problem, I have proposed a novel two-staged machine learning model for predicting M1.0+ class flares in the next 12 hours. The first stage is a state-of-the-art VGG-16 convolutional neural network model that outputs a flaring probability by extracting features from raw magnetogram images. The output of this model is then used as input to an extremely randomized trees model in the second stage, along with various engineered physics-based and topological features extracted from the magnetogram. Not only does this architecture combine the predictive power of features extracted with different approaches (CNNs, topology-based, physics-based) — a first-of-its-kind model in this field — I show that it helps mitigate the overforecasting problem.

My first important contribution is the impact evaluation of various dataset manipulations and modeling strategies on the performance of a CNN-only model (i.e. the first stage VGG-16

model). Primary among the dataset manipulations is dataset augmentation. I find that performing augmentation of the minority class (using standard rotation and polarity swapping) on the dataset for this model yields no improvement in predictive skill. It should be noted that, unlike some studies which incorrectly augment both training and testing sets as explained in Section 5.1, I perform augmentation only on the training set. I also explore the use of temporal sequences of different components of the magnetogram data in the VGG-16 model, showing that the B_r sequence, alone, is just as predictive as the full stack (B_r, B_θ, and B_ϕ), indicating redundancy between these components of the field. Finally, when modeling on a temporal sequence of image data, I show that adding an LSTM layer after the VGG-16 stage causes the overall model to perform worse than a strategy that uses the sequence as channels to the VGG-16 input layer. Finally, I also show that the VGG-16 model by itself does not do any better than the significantly less complex logistic regression model trained on SHARPs and topological features from the previous chapter. This implies that not only does the high complexity of VGG-16 have no impact on performance when compared with the simple logistic regression, but the two different feature extraction approaches (engineered versus the CNN-based features) are comparable. In order to study both effects separately — feature extraction and complexity — one could first extract CNN-based features using an autoencoder, and then use those to train feature-based models described in Chapter 4, like in [20]. This would then provide a fair comparison between the two featurization methods on a model of fixed complexity.

The second focus of the study in this chapter is the use of binary categorization metrics for evaluating flare prediction models. Tuning hyperparameters to optimize on the commonly-used TSS metric alone can lead to a model that highly overforecasts, i.e. one with many false positives. To address this, I propose a modified alternative metric — $\text{TSS}_{\text{scaled}}$, which I showed reduces the false positive rate in optimized models.

The third major contribution from this chapter is the incorporation of an ERT model as a second stage. The ERT model is trained on a feature set that includes the output probability from the VGG-16 model from the first stage together with various engineered features. This hybrid design offers various advantages. First, it combines the prediction power of the features learned

automatically by VGG-16 from magnetogram images with the engineered features described in the earlier chapters of this thesis. To the best of my knowledge, this is a first-of-its-kind approach in the flare prediction literature that combines different feature extraction techniques into a single model. Secondly, the two-stage model has significantly lower false positive rates compared to the VGG-16 model alone: it reduces the false positives (≈48%) without significantly reducing the true positives (≈12%). Finally, the ERT model provides a natural way to rank the forecasting capability of the various features. Using this method, I found that two features considerably outrank the others. The most highly ranked is the VGG-16 output probability, indicating that the first-stage model is skillfully predictive of flares. The second feature is the `R_VALUE` parameter — the total flux in the polarity inversion line — a feature designed by solar physicists for discriminating flaring from non-flaring active regions [86].

Chapter 6

Conclusions and Future Work

Accurate solar eruption forecasting has become ever imperative given the significant impact of space weather on human activity, especially the electric grid. Humanity's expanding space program has become a second important reason to monitor space weather. Latest events concerning the loss of multiple Space-X satellites due to a geomagnetic storm are evidence enough to stress this point [68]. Since solar eruptions occur around complex magnetic field structures called active regions (ARs), computationally modeling these solar eruptions the traditional way involves numerically solving for the AR magnetic field starting from an initial condition. The biggest problem with this approach is that current instrumentation only observes the magnetic field data on the photosphere, while the rest of the 3-dimensional magnetic field in the upper solar atmosphere is missing. While the photospheric field data is informative as useful indicator for solar eruptions, it does not provide a complete initial condition for numerical simulations of the future evolution of the system.

This is where machine learning models come to the rescue: the magnetic field observations while insufficient for numerical modeling are well suited for developing ML models. Over the last couple of decades, ML approaches to solar flare prediction have taken off, resulting in a slew of approaches that determine a correlation between solar observations (photospheric, chromospheric and coronal properties) with an upcoming solar flare (eruption). Of course, these models are not perfect. Noisy observations, lack of sufficient data, and a high dataset imbalance (due to flaring being a rare event phenomenon) has generally led to under-performing models, leaving a lot to be desired in this field. A 2016 workshop to systematically compare many of these models showed that

none of the models significantly outperform others [6]; two similar workshop papers from 2019 also showed that none of the methods outperform human-in-the-loop solar flare forecasting methods [57, 58]. Clearly, ML-based solar flare prediction is far from being a solved, and it poses many open challenges.

This thesis aims not to solve this problem in a general way, but rather to investigate the current limitations of the existing methodology and address a few of these important challenges. I first address the form of the input data to the ML models. Majority of the existing ML approaches use the standard set of properties extracted from the magnetic field properties — called *SHARPs* — that are motivated by the physics of the AR. These carefully designed physics-based features are believed to be a good way to characterize the state of the AR. In this thesis, I proposed a new and quite different set of features. Inspired from the McIntosh and Hale classification systems that characterize ARs using their shape, I applied topological data analysis (TDA) — a tool from applied mathematics that describe shapes of arbitrary discrete data — to AR magnetic field images (Chapter 3). The TDA-based features I extracted using this abstract methodology—which require no hand-crafting by human experts—describe "holes" present in these images at different thresholds. A preliminary analysis using a single ML model showed that these newly proposed features performed better in 24-hour forecasts of major flares than the standard SHARPs features [29].

As a next step, I designed a systematic model training, optimization, and evaluation framework to evaluate and compare the performance of the TDA and SHARPs feature sets using multiple ML models. While numerous ML models trained on various feature sets have been implemented in the solar flare prediction literature, an important aspect that is often missing in these studies is hyperparameter tuning, especially when comparing the performance of different ML models. To address this, I designed a pipeline for automatically selecting optimal hyperparameters to provide the best True Skill Statistic (a popular metric to measure model performance on imbalanced datasets) for each model and feature set combination using a k-fold cross validation. In this study, reported in [32] and covered in Chapter 4, I also used a more comprehensive SHARPs feature set to

incorporate important features missed in the preliminary study. I chose four models in increasing order of complexity: logistic regression, extremely randomized trees, multilayer perceptron (MLP) and long short-term memory (LSTM). Two key findings were the result of this study. Firstly, neither the TDA-based or the SHARPs-based feature set outperformed the other across all of the ML models. While this differed from my preliminary analysis, it validated my claims about the power of TDA for featurizing magnetic field images: abstract domain-independent features extracted from the magnetogram perform just as well as the carefully chosen features by domain experts. Further, the combination of the two feature sets performed just as well as the individual sets alone, indicating that neither feature set adds informative value to the other from the point of view of solar flare forecasting. It should be mentioned that another study by [93] influenced from my work have extracted TDA-based features from the neutral line regions and similarly showcased their importance. Lastly, with the aim of reducing complexity of these models, I investigated the use of principal component analysis to reduce the feature sets. Using the same tuning framework, I showed that the less-complex models trained on PCA-reduced versions of the feature sets performed almost as well as the original models trained on the full feature sets. I also showed that the TDA-based feature set was the most reducible, with a PCA-reduced version of just three features (components), highlighting yet another advantage of TDA-based analysis.

The last part of this thesis explored an important missing component in my thesis so far: the application of deep learning models that extract patterns directly from the magnetogram images (as opposed to featurizing the images first). To that end, I investigated the performance of VGG-16, a standard convolutional neural network (CNN) architecture, for flare forecasting. When optimizing VGG-16 to maximize the True Skill Statistic (TSS), the model exhibited a key problem also observed in the previously studied models in Chapter 4: overforecasting or high false positives. In both cases, this was a side effect of a highly imbalanced dataset. I proposed two solutions to this problem. The first was a modification of the TSS metric used to optimize the models, the scaled-TSS score, that additionally penalizes false positives. Optimizing on this metric reduced overforecasting, drastically reducing the false positives without significantly reducing the true positives. Building on this, I

developed a novel hybrid architecture which combined the features learned automatically by VGG-16 with the SHARPs and TDA features. This hybrid model had two stages: a first-stage VGG-16 model that outputs a flaring probability, which is then combined with TDA and SHARPs features to train an extremely randomized trees model in the second stage. This architecture further improved on the false positive rate, over the single-stage VGG-16 model. Use of the ERT architecture in the second stage also allowed for ranking on all the features, which showed the VGG-16 flaring probability as the most important feature followed by the `R_VALUE` parameter—the total magnetic flux in the neutral line region. This indicated that the VGG-16, by itself, was quite capable in discriminating flaring from non-flaring active magnetograms.

This thesis covers a range of ML methods and various features based around magnetic field observations of active regions. The hybrid model proposed towards the end of this thesis in a sense encompasses different kinds of information extracted from magnetograms (physics-based, topological and CNN-based) that can be useful for predicting flares. Different modeling strategies have also been employed in the form of ML models with varying degrees of complexity. I conjecture that any further exploration in magnetogram-based modeling is unlikely to show performance improvements over the current models presented in the thesis; rather, the field will have to move towards chromospheric and coronal observations as well. These data contain useful information about the AR structure as well, for example coronal loops and micro-flares, which could be potential indicators of impending major flares, and are available in the Atmospheric Imaging Assembly (AIA) products from the SDO mission [60]. Future work should involve employing some of the ML strategies described in this thesis (featurization and modeling) to AIA data. In addition to the AIA-based observations, another non-HMI feature that could be incorporated in these models is the flaring history, i.e. whether a magnetogram observation produced a C, M or an X-class flare in the past m hours. These history-based features — which are in a sense, persistence-based — have been incorporated successfully in certain works, e.g. [47, 65].

With data images available across different channels (multiple AIA wavelengths, HMI magnetograms, dopplergrams and continuum images), developing deep learning models that extract

information across different multiple disparate channels (and possibly also across temporally sequential instances) will be challenging. Newer strategies, such as 4D Minkowski ConvNets [21] and self attention models [106], which can efficiently extract patterns from high dimensional raw input data, will need to be employed. Additionally, I have focused on developing ML models with a forecasting window of 12/24 hours from an AR observation, but would be useful to develop ML models for shorter windows (3-12 hours). Such a solar flare warning product would be more useful to space weather customers such as airlines and radar operators who can then take rapid remediating actions. With a short forecasting window, though, the dataset imbalance problem becomes higher, worsening the overforecasting problem. Whether strategies discussed in Chapter 5 to mitigate overforecasting also apply for shorter windows will need to be validated.

Lastly, the introduction of the $\text{TSS}_{\text{scaled}}$ metric in this thesis opens up avenues for exploring more useful metrics by operational forecasters to optimize and evaluate ML models. The $\text{TSS}_{\text{scaled}}$ metric was a modification of the TSS score (= TPR − FPR) that scaled the FPR with a factor greater than unity chosen through an exploration of different hyperparameter values. One could alternatively choose a user-defined scaling factor k to modify the TSS, e.g., $\text{TSS}_{\text{modified}} = \text{TPR} - k \times \text{FPR}$. The forecaster could then choose different values of k based on their priorities of TPR-FPR balance, the FPR-associated penalty being directly proportional to k. The value of k could also be determined by taking into account the economic costs associated with false positives and false negatives. For example, while the cost associated with the damage from a missed flare or eruption (false negative) can be very high (power grid failures, spacecraft damages, etc.), the accrued cost associated with frequent multiple false alarms (false positives) could surpass the false negative cost (power grid adjustments, flight cancellations, etc.). Further, the damages or costs associated with flares of different intensities may vary, and this would need to be accounted for evaluating flare prediction systems that go beyond a simple binary major flare forecast (M1.0+ flare/no-flare). Designing such a metric would therefore require a close collaboration between solar physicists, ML scientists and economists.

ML-based solar flare prediction is an interesting and challenging field, evolving rapidly with

the availability of richer data generated from ever-improving solar instrumentation and advances in the field of machine learning. In this thesis, I have explored various strategies for improving ML-based flare forecasting using magnetic field observations. These include developing innovative featurization techniques, novel ML architectures, a systematic evaluation methodology and new tuning metrics for reducing overforecasting — an inherent problem arising in this problem due to an imbalanced dataset. With the evolution of this field, I envision these techniques being incorporated to produce reliable and efficient models for real-time solar flare forecasting.

Bibliography

[1] A. K. Abed, R. Qahwaji, and A. Abed. The Automated Prediction of Solar Flares from SDO Images using Deep Learning. Advances in Space Research, 67(8):2544–2557, 2021. ISSN 0273-1177. doi: https://doi.org/10.1016/j.asr.2021.01.042.

[2] H. Adams, T. Emerson, M. Kirby, R. Neville, C. Peterson, P. Shipman, S. Chepushtanova, E. Hanson, F. Motta, and L. Ziegelmeier. Persistence Images: A Stable Vector Representation of Persistent Homology. Journal of Machine Learning Research, 18(1):218–252, Jan. 2017. ISSN 1532-4435. doi: 10.5555/3122009.3122017.

[3] G. Aulanier, E. Pariat, and P. Démoulin. Current Sheet Formation in Quasi-Separatrix Layers and Hyperbolic Flux Tubes. Astronomy & Astrophysics, 444:961–976, 2005. doi: 10.1051/0004-6361:20053600.

[4] G. Barnes and K. D. Leka. Photospheric Magnetic Field Properties of Flaring vs. Non-Flare Quiet Active Regions. III. Magnetic Charge Topology Models. The Astrophysical Journal, 646:1303–1318, 2006. doi: 10.1086/504960.

[5] G. Barnes, D. W. Longcope, and K. D. Leka. Implementing a Magnetic Charge Topology Model for Solar Active Regions. The Astrophysical Journal, 629(1):561, 2005. doi: 10.1086/431175.

[6] G. Barnes, K. D. Leka, C. J. Schrijver, T. Colak, R. Qahwaji, O. W. Ashamari, Y. Yuan, J. Zhang, R. T. J. McAteer, D. S. Bloomfield, P. A. Higgins, P. T. Gallagher, D. A. Falconer, M. K. Georgoulis, M. S. Wheatland, C. Balch, T. Dunn, and E. L. Wagner. A Comparison of Flare Forecasting Methods. I. Results from the "All-Clear" Workshop. The Astrophysical Journal, 829:89, Oct. 2016. doi: 10.3847/0004-637X/829/2/89.

[7] J. Béland and K. Small. Space Weather Effects on Power Transmission Systems: The Cases of Hydro-Québec and Transpower New Zealand Ltd. In I. A. Daglis, editor, Effects of Space Weather on Technology Infrastructure, pages 287–299, Dordrecht, 2005. Springer Netherlands. ISBN 978-1-4020-2754-3. doi: 10.1007/1-4020-2754-0_15.

[8] F. Benvenuto, M. Piana, C. Campi, and A. M. Massone. A Hybrid Supervised/Unsupervised Machine Learning Approach to Solar Flare Prediction. The Astrophysical Journal, 853(1):90, Jan. 2018. doi: 10.3847/1538-4357/aaa23c.

[9] M. G. Bobra and S. Couvidat. Solar Flare Prediction Using SDO/HMI Vector Magnetic Field Data with a Machine-Learning Algorithm. The Astrophysical Journal, 798(2):135, Jan. 2015. doi: 10.1088/0004-637x/798/2/135.

[10] M. G. Bobra, X. Sun, J. T. Hoeksema, M. Turmon, Y. Liu, K. Hayashi, G. Barnes, and K. D. Leka. The Helioseismic and Magnetic Imager (HMI) Vector Magnetic Field Pipeline: SHARPs – Space-Weather HMI Active Region Patches. Solar Physics, 289(9):3549–3578, Sep. 2014. doi: 10.1007/s11207-014-0529-3.

[11] L. E. Boucheron, A. Al-Ghraibah, and R. T. J. McAteer. Prediction of Solar Flare Size and Time-to-Flare Using Support Vector Machine Regression. The Astrophysical Journal, 812: 51, Oct. 2015. doi: 10.1088/0004-637X/812/1/51.

[12] K. Boyd, K. H. Eng, and C. D. Page. Area Under the Precision-Recall Curve: Point Estimates and Confidence Intervals. In H. Blockeel, K. Kersting, S. Nijssen, and F. Železný, editors, Machine Learning and Knowledge Discovery in Databases, pages 451–466, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-40994-3.

[13] P. Bubenik. Statistical Topological Data Analysis Using Persistence Landscapes. Journal of Machine Learning Research, 16(1):77–102, Jan. 2015. ISSN 1532-4435. doi: 10.5555/2789272.2789275.

[14] C. Campi, F. Benvenuto, A. M. Massone, D. S. Bloomfield, M. K. Georgoulis, and M. Piana. Feature Ranking of Active Region Source Properties in Solar Flare Forecasting and the Uncompromised Stochasticity of Flare Occurrence. The Astrophysical Journal, 883(2):150, Sep. 2019. doi: 10.3847/1538-4357/ab3c26.

[15] E. Camporeale. The Challenge of Machine Learning in Space Weather: Nowcasting and Forecasting. Space Weather, 17(8):1166–1207, 8 2019. ISSN 1542-7390. doi: 10.1029/2018SW002061.

[16] R. M. Caplan, C. Downs, J. A. Linker, and Z. Mikic. Variations in finite-difference potential fields. The Astrophysical Journal, 915(1):44, Jul. 2021. doi: 10.3847/1538-4357/abfd2f.

[17] M. Carrière, M. Cuturi, and S. Oudot. Sliced Wasserstein Kernel for Persistence Diagrams. In D. Precup and Y. W. Teh, editors, Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research, pages 664–673. PMLR, Aug. 2017. URL `https://proceedings.mlr.press/v70/carriere17a.html`.

[18] M. Carriere, F. Chazal, Y. Ike, T. Lacombe, M. Royer, and Y. Umeda. PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures. In Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, volume 108 of Proceedings of Machine Learning Research, pages 2786–2796. PMLR, Aug. 2020. URL `https://proceedings.mlr.press/v108/carriere20a.html`.

[19] F. Chazal, B. T. Fasy, F. Lecci, A. Rinaldo, and L. Wasserman. Stochastic Convergence of Persistence Landscapes and Silhouettes. In Proceedings of the Thirtieth Annual Symposium on Computational Geometry, SOCG'14, pages 474:474–474:483, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2594-3. doi: 10.1145/2582112.2582128.

[20] Y. Chen, W. B. Manchester, A. O. Hero, G. Toth, B. DuFumier, T. Zhou, X. Wang, H. Zhu, Z. Sun, and T. I. Gombosi. Identifying Solar Flare Precursors Using Time Series of SDO/HMI Images and SHARP Parameters. Space Weather, 17(10):1404–1426, 2019. ISSN 1542-7390. doi: 10.1029/2019SW002214.

[21] C. Choy, J. Gwak, and S. Savarese. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2019.

[22] T. Cinto, A. L. S. Gradvohl, G. P. Coelho, and A. E. A. da Silva. A Framework for Designing and Evaluating Solar Flare Forecasting Systems. Monthly Notices of the Royal Astronomical Society, 495(3):3332–3349, May 2020. ISSN 0035-8711. doi: 10.1093/mnras/staa1257.

[23] T. Colak and R. Qahwaji. Automated Solar Activity Prediction: A Hybrid Computer Platform Using Machine Learning and Solar Imaging for Automated Prediction of Solar Flares. Space Weather, 7(6), 2009. doi: 10.1029/2008SW000401.

[24] M. D. Crown. Validation of the NOAA Space Weather Prediction Center's Solar Flare Forecasting Look-up Table and Forecaster-Issued Probabilities. Space Weather, 10(6), 2012. doi: https://doi.org/10.1029/2011SW000760.

[25] V. de Silva and R. Ghrist. Coverage in Sensor Networks Via Persistent Homology. Algebraic & Geometric Topology, 7:339–358, 2007. doi: 10.2140/agt.2007.7.339.

[26] P. Demoulin, J. C. Henoux, E. R. Priest, and C. H. Mandrini. Quasi-Separatrix Layers in Solar Fares. I. Method. Astronomy & Astrophysics, 308:643–655, 1996.

[27] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

[28] M. L. DeRosa, C. J. Schrijver, G. Barnes, K. D. Leka, B. W. Lites, M. J. Aschwanden, T. Amari, , A. Canou, J. M. McTiernan, S. Régnier, J. K. Thalmann, G. Valori, M. S. Wheatland, T. Wiegelmann, M. C. M. Cheung, P. A. Conlon, M. Fuhrmann, B. Inhester, and T. Tadesse. A Critical Assessment of Nonlinear Force-Free Field Modeling of the Solar Corona for Active Region 10953. The Astrophysical Journal, 696(2):1780–1791, 2009. doi: 10.1088/0004-637x/696/2/1780.

[29] V. Deshmukh, T. E. Berger, E. Bradley, and J. D. Meiss. Leveraging the Mathematics of Shape for Solar Magnetic Eruption Prediction. Journal of Space Weather and Space Climate, 10:13, 2020. doi: 10.1051/swsc/2020014.

[30] V. Deshmukh, T. Berger, J. Meiss, and E. Bradley. Shape-based Feature Engineering for Solar Flare Prediction. Proceedings of the AAAI Conference on Artificial Intelligence, 35(17): 15293–15300, May 2021. URL `https://ojs.aaai.org/index.php/AAAI/article/view/17795`.

[31] V. Deshmukh, N. Flyer, K. V. D. Sande, and T. Berger. Decreasing False Alarm Rates in ML-based Solar Flare Prediction using SDO/HMI Data, 2021.

[32] V. Deshmukh, S. Baskar, E. Bradley, T. Berger, and J. D. Meiss. Machine Learning Approaches to Solar-Flare Forecasting: Is Complex Better? arXiv preprint arXiv:2202.08776, 2022.

[33] J. Duchi, E. Hazan, and Y. Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. Journal of Machine Learning Research, 12(null):2121–2159, Jul. 2011. ISSN 1532-4435. doi: 10.5555/1953048.2021068.

[34] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological Persistence and Simplification. pages 454–463, 2000. doi: 10.1109/SFCS.2000.892133.

[35] K. Florios, I. Kontogiannis, S.-H. Park, J. A. Guerra, F. Benvenuto, D. S. Bloomfield, and M. K. Georgoulis. Forecasting Solar Flares Using Magnetogram-Based Predictors and Machine Learning. Solar Physics, 293(2):28, Feb. 2018. doi: 10.1007/s11207-018-1250-4.

[36] P. Gallagher, Y.-J. Moon, and H. Wang. Active-Region Monitoring and Flare Forecasting—I. Data Processing and First Results. Solar Physics, 209:171–183, 09 2002. doi: 10.1023/A:1020950221179.

[37] P. Geurts, D. Ernst, and L. Wehenkel. Extremely Randomized Trees. Machine Learning, 63 (1):3–42, Apr. 2006. doi: 10.1007/s10994-006-6226-1.

[38] R. Ghrist. Barcodes: The Persistent Topology of Data. Bulletin of the American Mathematical Society, 45(1):61–75, 2008. doi: 10.1090/S0273-0979-07-01191-3.

[39] I. Goodfellow, Y. Bengio, and A. Courville. Deep Learning. The MIT Press, 2016. ISBN 0262035618. doi: 10.5555/3086952.

[40] N. Gopalswamy, L. Barbieri, E. W. Cliver, G. Lu, S. P. Plunkett, and R. M. Skoug. Introduction to Violent Sun-Earth Connection Events of October–November 2003. Journal of Geophysical Research: Space Physics, 110(A9), 2005. doi: 10.1029/2005JA011268.

[41] J. A. Guerra, S. Park, P. T. Gallagher, I. Kontogiannis, M. K. Georgoulis, and D. S. Bloomfield. Active Region Photospheric Magnetic Properties Derived from Line-of-Sight and Radial Fields. Solar Physics, 293(1):9, Jan. 2018. doi: 10.1007/s11207-017-1231-z.

[42] G. E. Hale, F. Ellerman, S. B. Nicholson, and A. H. Joy. The Magnetic Polarity of Sun-Spots. The Astrophysical Journal, 49:153, Apr. 1919. doi: 10.1086/142452.

[43] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

[44] X. Huang, H. Wang, L. Xu, J. Liu, R. Li, and X. Dai. Deep Learning Based Solar Flare Forecasting Model. I. Results for Line-of-sight Magnetograms. The Astrophysical Journal, 856(1):7, Mar. 2018. doi: 10.3847/1538-4357/aaae00.

[45] I. Jolliffe and D. Stephenson. Forecast Verification: A Practitioner's Guide in Atmospheric Science. Wiley, 2012. ISBN 978-1-119-96107-9.

[46] E. Jonas, M. Bobra, V. Shankar, J. Todd Hoeksema, and B. Recht. Flare Prediction Using Photospheric and Coronal Image Data. Solar Physics, 293(3):48, 2018. ISSN 1573-093X. doi: 10.1007/s11207-018-1258-9.

[47] E. Jonas, M. Bobra, V. Shankar, J. Todd Hoeksema, and B. Recht. Flare Prediction Using Photospheric and Coronal Image Data. Solar Physics, 293(3):48, Mar 2018. doi: 10.1007/s11207-018-1258-9.

[48] T. Kaczynski, K. Mischaikow, and M. Mrozek. Computational Homology, volume 157 of Applied Mathematical Sciences. Springer-Verlag, New York, 2004.

[49] I. S. Knyazeva, F. A. Urtiev, and N. G. Makarenko. On the Prognostic Efficiency of Topological Descriptors for Magnetograms of Active Regions. Geomagnetism and Aeronomy, 57 (8):1086–1091, 2017. ISSN 1555-645X. doi: 10.1134/S0016793217080126.

[50] I. Kontogiannis, M. K. Georgoulis, S.-H. Park, and J. A. Guerra. Testing and Improving a Set of Morphological Predictors of Flaring Activity. Solar Physics, 293(6):96, Jun. 2018. doi: 10.1007/s11207-018-1317-2.

[51] H. Koskinen, E. Tanskanen, R. Pirjola, A. Pulkkinen, C. Dyer, D. Rodgers, P. Cannon, J. Mandeville, D. Boscher, and A. Hilgers. Space Weather Effects Catalogue. volume 2, 2001.

[52] S. Kotsiantis, D. Kanellopoulos, P. Pintelas, et al. Handling Imbalanced Datasets: A Review. GESTS International Transactions on Computer Science and Engineering, 30(1):25–36, 2006.

[53] H. Künzel. Zur Klassifikation von Sonnenfleckengruppen. Astronomische Nachrichten, 288: 177, Dec. 1965.

[54] G. Kusano, Y. Hiraoka, and K. Fukumizu. Persistence Weighted Gaussian Kernel for Topological Data Analysis. 48:2004–2013, Jun. 2016. URL `https://proceedings.mlr.press/v48/kusano16.html`.

[55] T. Le and M. Yamada. Persistence Fisher Kernel: A Riemannian Manifold Kernel for Persistence Diagrams. 31, 2018. URL `https://proceedings.neurips.cc/paper/2018/file/959ab9a0695c467e7caf75431a872e5c-Paper.pdf`.

[56] K. D. Leka and G. Barnes. Photospheric Magnetic Field Properties of Flaring versus Flare-Quiet Active Regions. IV. A Statistically Significant Sample. The Astrophysical Journal, 656: 1173–1186, Feb. 2007. doi: 10.1086/510282.

[57] K. D. Leka, S.-H. Park, K. Kusano, J. Andries, G. Barnes, S. Bingham, D. S. Bloomfield, A. E. McCloskey, V. Delouille, D. Falconer, P. T. Gallagher, M. K. Georgoulis, Y. Kubo, K. Lee, S. Lee, V. Lobzin, J. Mun, S. A. Murray, T. A. M. Hamad Nageem, R. Qahwaji, M. Sharpe, R. A. Steenburgh, G. Steward, and M. Terkildsen. A Comparison of Flare Forecasting Methods. II. Benchmarks, Metrics, and Performance Results for Operational Solar Flare Forecasting Systems. The Astrophysical Journal, 243(2):36, Aug. 2019. doi: 10.3847/1538-4365/ab2e12.

[58] K. D. Leka, S.-H. Park, K. Kusano, J. Andries, G. Barnes, S. Bingham, D. S. Bloomfield, A. E. McCloskey, V. Delouille, D. Falconer, P. T. Gallagher, M. K. Georgoulis, Y. Kubo, K. Lee, S. Lee, V. Lobzin, J. Mun, S. A. Murray, T. A. M. Hamad Nageem, R. Qahwaji, M. Sharpe, R. A. Steenburgh, G. Steward, and M. Terkildsen. A Comparison of Flare Forecasting Methods. III. Systematic Behaviors of Operational Solar Flare Forecasting Systems. The Astrophysical Journal, 881(2):101, Aug. 2019. doi: 10.3847/1538-4357/ab2e11.

[59] Leka, K.D., Barnes, Graham, and Wagner, Eric. The NWRA Classification Infrastructure: description and extension to the Discriminant Analysis Flare Forecasting System (DAFFS). J. Space Weather Space Clim., 8:A25, 2018. doi: 10.1051/swsc/2018004.

[60] J. R. Lemen, A. M. Title, D. J. Akin, P. F. Boerner, C. Chou, J. F. Drake, D. W. Duncan, C. G. Edwards, F. M. Friedlaender, G. F. Heyman, N. E. Hurlburt, N. L. Katz, G. D.

Kushner, M. Levay, R. W. Lindgren, D. P. Mathur, E. L. McFeaters, S. Mitchell, R. A. Rehse, C. J. Schrijver, L. A. Springer, R. A. Stern, T. D. Tarbell, J.-P. Wuelser, C. J. Wolfson, C. Yanari, J. A. Bookbinder, P. N. Cheimets, D. Caldwell, E. E. Deluca, R. Gates, L. Golub, S. Park, W. A. Podgorski, R. I. Bush, P. H. Scherrer, M. A. Gummin, P. Smith, G. Auker, P. Jerram, P. Pool, R. Soufli, D. L. Windt, S. Beardsley, M. Clapp, J. Lang, and N. Waltham. The Atmospheric Imaging Assembly (AIA) on the Solar Dynamics Observatory (SDO), pages 17–40. Springer US, New York, NY, 2012. ISBN 978-1-4614-3673-7. doi: 10.1007/978-1-4614-3673-7_3.

[61] X. Li, Y. Zheng, X. Wang, and L. Wang. Predicting Solar Flares Using a Novel Deep Convolutional Neural Network. The Astrophysical Journal, 891(1):10, Feb. 2020. doi: 10.3847/1538-4357/ab6d04.

[62] R. Liaw et al. Tune: A Research Platform for Distributed Model Selection and Training. arXiv e-prints, art. arXiv:1807.05118, 2018.

[63] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar. Focal Loss for Dense Object Detection. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Oct. 2017.

[64] D. C. Liu and J. Nocedal. On The Limited Memory BFGS Method for Large Scale Optimization. Mathematical Programming, 45(1):503–528, Aug. 1989. ISSN 1436-4646. doi: 10.1007/BF01589116.

[65] H. Liu, C. Liu, J. T. L. Wang, and H. Wang. Predicting Solar Flares Using a Long Short-term Memory Network. The Astrophysical Journal, 877(2):121, Jun. 2019. doi: 10.3847/1538-4357/ab1b3c.

[66] D. W. Longcope. Topological Methods for the Analysis of Solar Magnetic Fields. Living Reviews in Solar Physics, 2(1):7, 2005. doi: 10.12942/lrsp-2005-7.

[67] T. M. Loto'aniu, H. J. Singer, J. V. Rodriguez, J. Green, W. Denig, D. Biesecker, and V. Angelopoulos. Space Weather Conditions During the Galaxy 15 Spacecraft Anomaly. Space Weather, 13(8):484–502, 2015. doi: 10.1002/2015SW001239.

[68] T. Malik. SpaceX Says a Geomagnetic Storm Just Doomed 40 Starlink Internet Satellites. Space.com. URL `https://www.space.com/spacex-starlink-satellites-lost-geomagnetic-storm`.

[69] R. Martinez-Cantin. BayesOpt: A Bayesian Optimization Library for Nonlinear Optimization, Experimental Design and Bandits. Journal of Machine Learning Research, 15(1):3735–3739, Jan. 2014. ISSN 1532-4435. doi: 10.5555/2627435.2750364.

[70] L. McInnes, J. Healy, N. Saul, and L. Großberger. UMAP: Uniform Manifold Approximation and Projection. Journal of Open Source Software, 3(29):861, 2018. doi: 10.21105/joss.00861.

[71] P. S. McIntosh. The Classification of Sunspot Groups. Solar Physics, 125:251–267, 1990. doi: 10.1007/BF00158405.

[72] T. R. Metcalf, M. L. DeRosa, C. J. Schrijver, G. Barnes, A. A. van Ballegooijen, T. Wiegelmann, M. S. Wheatland, G. Valori, and J. M. McTiernan. Nonlinear Force-Free Modeling of Coronal Magnetic Fields. II. Modeling a Filament Arcade and Simulated Chromospheric and Photospheric Vector Fields. Solar Physics, 247(2):269–299, 2008. doi: 10.1007/s11207-007-9110-7.

[73] K. P. Murphy. Machine Learning: A Probabilistic Perspective. The MIT Press, 2012. ISBN 0262018020.

[74] N. Nishizuka, K. Sugiura, Y. Kubo, M. Den, S. Watari, and M. Ishii. Solar Flare Prediction Model with Three Machine-Learning Algorithms using Ultraviolet Brightening and Vector Magnetograms. The Astrophysical Journal, 835:156, Feb. 2017. doi: 10.3847/1538-4357/835/2/156.

[75] N. Nishizuka, K. Sugiura, Y. Kubo, M. Den, and M. Ishii. Deep Flare Net (DeFN) Model for Solar Flare Prediction. The Astrophysical Journal, 858(2):113, May 2018. doi: 10.3847/1538-4357/aab9a7.

[76] N. Nishizuka, Y. Kubo, K. Sugiura, M. Den, and M. Ishii. Operational Solar Flare Prediction Model using Deep Flare Net. Earth, Planets and Space, 73(1):1–12, 2021. doi: 10.1186/s40623-021-01381-9.

[77] S. Odenwald, J. Green, and W. Taylor. Forecasting the Impact of an 1859-Calibre Superstorm on Satellite Resources. Advances in Space Research, 38(2):280 – 297, 2006. ISSN 0273-1177. doi: 10.1016/j.asr.2005.10.046.

[78] E. Park, Y.-J. Moon, S. Shin, K. Yi, D. Lim, H. Lee, and G. Shin. Application of the Deep Convolutional Neural Network to the Forecast of Solar Flare Occurrence Using Full-disk Solar Magnetograms. The Astrophysical Journal, 869(2):91, Dec. 2018. doi: 10.3847/1538-4357/aaed40.

[79] W. D. Pesnell, B. J. Thompson, and P. C. Chamberlin. The Solar Dynamics Observatory (SDO). pages 3–15, 2012. doi: 10.1007/978-1-4614-3673-7_2.

[80] E. R. Priest and P. Démoulin. Three-dimensional Magnetic Reconnection without Null Points. 1. Basic Theory of Magnetic Flipping. Journal of Geophysical Research, 100:23443–23464, 1995. doi: 10.1029/95JA02740.

[81] L. E. Raileanu and K. Stoffel. Theoretical Comparison between the Gini Index and Information Gain Criteria. Annals of Mathematics and Artificial Intelligence, 41(1):77–93, May 2004. ISSN 1573-7470. doi: 10.1023/B:AMAI.0000018580.96245.c6.

[82] J. Reininghaus, S. Huber, U. Bauer, and R. Kwitt. A Stable Multi-Scale Kernel for Topological Machine Learning. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 4741–4748, Jun. 2015. doi: 10.1109/CVPR.2015.7299106.

[83] B. D. Ripley. The Second-Order Analysis of Stationary Point Processes. Journal of Applied Probability, 13(2):255–266, 1976. doi: 10.2307/3212829.

[84] P. H. Scherrer, J. Schou, R. I. Bush, A. G. Kosovichev, R. S. Bogart, J. T. Hoeksema, Y. Liu, T. L. Duvall, J. Zhao, A. M. Title, C. J. Schrijver, T. D. Tarbell, and S. Tomczyk. The Helioseismic and Magnetic Imager (HMI) Investigation for the Solar Dynamics Observatory (SDO). Solar Physics, 275(1):207–227, Jan. 2012. ISSN 1573-093X. doi: 10.1007/s11207-011-9834-2.

[85] B. Schölkopf, A. Smola, and K.-R. Müller. Kernel Principal Component Analysis. In W. Gerstner, A. Germond, M. Hasler, and J.-D. Nicoud, editors, Artificial Neural Networks — ICANN'97, pages 583–588, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg. doi: 10.1007/BFb0020217.

[86] C. J. Schrijver. A Characteristic Magnetic Field Pattern Associated with All Major Solar Flares and Its Use in Flare Forecasting. The Astrophysical Journal Letters, 655(2):L117, 2007. doi: 10.1086/511857.

[87] C. J. Schrijver, M. L. DeRosa, T. R. Metcalf, Y. Liu, J. McTiernan, S. Régnier, G. Valori, M. S. Wheatland, and T. Wiegelmann. Nonlinear Force-Free Modeling of Coronal Magnetic Fields Part I: A Quantitative Comparison of Methods. Solar Physics, 235(1-2):161–190, 2006. doi: 10.1007/s11207-006-0068-7.

[88] C. J. Schrijver, M. L. DeRosa, T. Metcalf, G. Barnes, B. Lites, T. Tarbell, J. McTiernan, G. Valori, T. Wiegelmann, M. S. Wheatland, T. Amari, G. Aulanier, P. Démoulin, M. Fuhrmann, K. Kusano, S. Régnier, and J. K. Thalmann. Nonlinear Force-Free Field Modeling of a Solar Active Region around the Time of a Major Flare and Coronal Mass Ejection. The Astrophysical Journal, 675(2):1637–1644, Mar. 2008. doi: 10.1086/527413.

[89] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv e-prints, art. arXiv:1409.1556, Sept. 2014.

[90] G. Singh, F. Memoli, and G. Carlsson. Topological Methods for the Analysis of High Dimensional Data Sets and 3D Object Recognition. In M. Botsch, R. Pajarola, B. Chen, and M. Zwicker, editors, Eurographics Symposium on Point-Based Graphics. The Eurographics Association, 2007. ISBN 978-3-905673-51-7. doi: 10.2312/SPBG/SPBG07/091-100.

[91] J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian Optimization of Machine Learning Algorithms. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, Advances in Neural Information Processing Systems, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper/2012/file/05311655a15b75fab86956663e1819cd-Paper.pdf.

[92] H. Sun, W. Manchester, Z. Jiao, X. Wang, and Y. Chen. Interpreting LSTM Prediction on Solar Flare Eruption with Time-Series Clustering. arXiv preprint arXiv:1912.12360, 2019.

[93] H. Sun, W. Manchester IV, and Y. Chen. Improved and Interpretable Solar Flare Predictions With Spatial and Topological Features of the Polarity Inversion Line Masked Magnetograms. Space Weather, 19(12):e2021SW002837, 2021. doi: 10.1029/2021SW002837.

[94] L. Tarr and D. Longcope. Calculating Energy Storage Due to Topological Changes in Emerging Active Region NOAA AR 11112. The Astrophysical Journal, 749(1):64, Mar. 2012. doi: 10.1088/0004-637x/749/1/64.

[95] L. Tarr, D. Longcope, and M. Millhouse. Calculating Separate Magnetic Free Energy Estimates for Active Regions Producing Multiple Flares: NOAA AR11158. The Astrophysical Journal, 770(1):4, Jun. 2013. doi: 10.1088/0004-637X/770/1/4.

[96] T. J. Teisberg and R. F. Weiher. Valuation of Geomagnetic Storm Forecasts: An Estimate of the Net Economic Benefits of a Satellite Warning System. Journal of Policy Analysis and Management, 19(2):329–334, 2000. ISSN 02768739, 15206688. URL http://www.jstor.org/stable/3325618.

[97] S. G. Trevor Maynard, Nell Smith. Solar Storm Risk to the North American Electric Grid. Technical report, Lloyd's, 2013.

[98] L. Van der Maaten and G. Hinton. Visualizing Data Using t-SNE. Journal of machine learning research, 9(11), 2008.

[99] Y. M. Wang and N. R. Sheeley Jr. On Potential Field Models of the Solar Corona. Astrophysical Journal, 392:310, 1992. doi: 10.1086/171430.

[100] M. S. Wheatland. A Bayesian Approach to Solar Flare Prediction. The Astrophysical Journal, 609(2):1134–1139, Jul. 2004. doi: 10.1086/421261.

[101] T. Wiegelmann and T. Sakurai. Solar Force-Free Magnetic Fields. Living Reviews in Solar Physics, 9:5, 2012. doi: 10.12942/lrsp-2012-5.

[102] X. Xu, J. Cisewski-Kehe, S. B. Green, and D. Nagai. Finding Cosmic Voids and Filament Loops Using Topological Data Analysis. Astronomy and Computing, 27:34–52, 2019. doi: 10.1016/j.ascom.2019.02.003.

[103] X. Yang, G. Lin, H. Zhang, and X. Mao. Magnetic Nonpotentiality in Photospheric Active Regions as a Predictor of Solar Flares. The Astrophysical Journal, 774(2):L27, Aug. 2013. doi: 10.1088/2041-8205/774/2/l27.

[104] D. Yu, X. Huang, H. Wang, Y. Cui, Q. Hu, and R. Zhou. Short-Term Solar Flare Level Prediction Using a Bayesian Network Approach. The Astrophysical Journal, 710(1):869, 2010.

[105] Y. Yuan, F. Shih, J. Jing, and H. Wang. Solar Flare Forecasting using Sunspot-Groups Classification and Photospheric Magnetic Parameters. Proceedings of the International Astronomical Union, 6:446 – 450, 08 2010. doi: 10.1017/S1743921311015742.

[106] H. Zhao, J. Jia, and V. Koltun. Exploring Self-Attention for Image Recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2020.

[107] Y. Zheng, X. Li, and X. Wang. Solar Flare Prediction with the Hybrid Deep Convolutional Neural Network. The Astrophysical Journal, 885(1):73, Nov. 2019. doi: 10.3847/1538-4357/ab46bd.

[108] Y. Zheng, X. Li, Y. Si, W. Qin, and H. Tian. Hybrid Deep Convolutional Neural Network with One-Versus-One Approach for Solar Flare Prediction. Monthly Notices of the Royal Astronomical Society, 507(3):3519–3539, 07 2021. ISSN 0035-8711. doi: 10.1093/mnras/stab2132.