Predicting Coronal Mass Ejections Using SDO/HMI Vector Magnetic Data Products and Recurrent Neural Networks

Hao Liu1,2, Chang Liu1,3,4, Jason T. L. Wang1,2, and Haimin Wang1,3,4

1 Institute for Space Weather Sciences, New Jersey Institute of Technology, University Heights, Newark, NJ 07102-1982, USA; hl422@njit.edu, chang.liu@njit.edu, wangj@njit.edu, haimin.wang@njit.edu
2 Department of Computer Science, New Jersey Institute of Technology, University Heights, Newark, NJ 07102-1982, USA
3 Big Bear Solar Observatory, New Jersey Institute of Technology, 40386 North Shore Lane, Big Bear City, CA 92314-9672, USA
4 Center for Solar-Terrestrial Research, New Jersey Institute of Technology, University Heights, Newark, NJ 07102-1982, USA

Received 2019 August 31; revised 2019 December 10; accepted 2019 December 20; published 2020 February 7

Abstract

We present two recurrent neural networks (RNNs), one based on gated recurrent units and the other based on long short-term memory, for predicting whether an active region (AR) that produces an M- or X-class flare will also produce a coronal mass ejection (CME). We model data samples in an AR as time series and use the RNNs to capture temporal information on the data samples. Each data sample has 18 physical parameters, or features, derived from photospheric vector magnetic field data taken by the Helioseismic and Magnetic Imager on board the Solar Dynamics Observatory. We survey M- and X-class flares that occurred from 2010 to 2019 May using the Geostationary Operational Environmental Satellite’s X-ray flare catalogs provided by the National Centers for Environmental Information (NCEI), and select those flares with identified ARs in the NCEI catalogs. In addition, we extract the associations of flares and CMEs from the Space Weather Database of Notifications, Knowledge, Information. We use the information gathered above to build the labels (positive versus negative) of the data samples at hand. Experimental results demonstrate the superiority of our RNNs over closely related machine learning methods in predicting the labels of the data samples. We also discuss an extension of our approach to predict a probabilistic estimate of how likely an M- or X-class flare is to initiate a CME, with good performance results. To our knowledge this is the first time that RNNs have been used for CME prediction.

Unified Astronomy Thesaurus concepts: Solar coronal mass ejections (310); Solar flares (1496); Solar activity (1475); Solar active regions (1974)

1. Introduction

Coronal mass ejections (CMEs) are intense bursts of magnetic flux and plasma that are ejected from the Sun into interplanetary space (Lin & Forbes 2000). They are often associated with solar flares and originate from active regions (ARs) on the Sun’s photosphere where magnetic fields are strong and evolve rapidly. Major CMEs and their associated flares can cause severe influences on the near-Earth environment, resulting in potentially life-threatening consequences (Baker et al. 2004). Therefore, substantial efforts are being invested in developing new technologies for early detection and forecasting of flares and CMEs (Bobra & Ilonidis 2016; Inceoglu et al. 2018).

Both flares and CMEs are believed to be magnetically driven events; evidence shows that they may constitute different manifestations of the same physical process (Harrison 1995; Berkebile-Stoiser et al. 2012; Gosling 2013). However, solar observations over the past few decades have clearly indicated that there may not be a one-to-one correspondence between flares and CMEs, and their relationship is still under active investigation (see, e.g., Yashiro & Gopalswamy 2008; Kawabata et al. 2018). Much effort has been devoted to analyzing the structural properties of coronal magnetic fields, which may play an important role in determining whether an eruption evolves into a CME or remains as a confined flare (Török & Kliem 2005; DeVore & Antiochos 2008; Liu 2008; Baumgartner et al. 2018). In the meantime, many researchers have investigated the relationship between CME productivity and the features of the photospheric magnetic field of flare-productive ARs where the features can be directly derived from photospheric vector magnetograms. For example, Qahwaji et al. (2008) used properties such as flare duration along with machine learning algorithms to predict whether a flare is likely to initiate a CME. Bobra & Ilonidis (2016) used 18 physical features, or predictive parameters, derived from the photospheric vector magnetic field data provided by the Helioseismic and Magnetic Imager (HMI; Schou et al. 2010) on board the Solar Dynamics Observatory (SDO; Pesnell et al. 2011) to forecast whether a CME would be associated with an M- or X-class flare. The flares are classified according to the peak flux (in watts per square meter, W m−2) of 1 to 8 Å X-rays as measured by the Geostationary Operational Environmental Satellite (GOES). These authors found that using a combination of six intensive parameters captured most of the relevant information contained in the photospheric magnetic field. Inceoglu et al. (2018) later developed methods to forecast whether flares would be associated with CMEs and solar energetic particles (SEPs). The authors employed two machine learning algorithms, support vector machines (SVMs), and multilayer perceptrons (MLPs), and showed that SVMs performed slightly better than MLPs in the forecasting task.

Machine learning is a non-physics-based technology used in predictive analytics. It is a subfield of artificial intelligence, which grants computers abilities to learn from past data and make predictions on unseen future data (Alpaydin 2016; Goodfellow et al. 2016). Many different machine learning-based techniques have been developed for solar flare prediction (see, e.g., Liu et al. 2017, 2019; Florios et al. 2018; Jonas et al. 2018; Nishizuka et al. 2018). However, CME prediction has
been mainly based on SVMs (Bobra & Ilonidis 2016) and MLPs (Inceoglu et al. 2018).

In this paper, we attempt to extend the work of Bobra & Ilonidis (2016) by proposing new machine learning algorithms and applying them to SDO/HMI vector magnetic field data to predict whether an AR that produces an M- or X-class flare will also produce a CME. The machine learning algorithms we explore here include two kinds of recurrent neural networks (RNNs; Hopfield 1982): a long short-term memory (LSTM) network (Hochreiter & Schmidhuber 1997) and a gated recurrent unit (GRU) network (Cho et al. 2014). LSTM cells and GRUs differ in the number and type of gates employed in the networks—an LSTM cell has three gates (input, output, and forget gates) whereas a GRU has two gates (reset and update gates). RNNs can use their internal state (memory) and gates to process sequences of inputs, which makes them suitable for tasks such as speech recognition, handwriting recognition, and time series forecasting (LeCun et al. 2015; Goodfellow et al. 2016). In a CME prediction task, the observations in each AR form time series data, and hence RNNs work well at this task. To our knowledge, this is the first time that RNNs are used for CME prediction.

The rest of this paper is organized as follows. Section 2 describes our data collection scheme and predictive parameters used in the study. Section 3 presents our RNN architectures and algorithms. Section 4 reports experimental results. Section 5 concludes the paper.

2. Data and Predictive Parameters

We adopt the data products, namely Space-weather HMI Active Region Patches (SHARP; Bobra et al. 2014), produced by the SDO/HMI team. These data were released at the end of 2012 (Bobra et al. 2014) and can be found in the hmi.sharpdata series at the Joint Science Operations Center (JSOC). The SHARP data series contains ARs tracked throughout their lifetime and provides many physical parameters suitable for flare/CME predictions.

We collect data samples at a cadence of 12 minutes, where the data samples are retrieved from the hmi.sharp_cea_720s-definitive data series on the JSOC website by using SunPy (SunPy Community et al. 2015). We use the same 18 features, or SHARP parameters, as described in Table 1 of Bobra & Ilonidis (2016) that characterize AR magnetic field properties for CME prediction. These 18 features or predictive parameters include MEANPOL (mean photospheric magnetic free energy), SHRGT45 (fraction of area with shear >45°), TOTPOL (total photospheric magnetic free energy density), USFLUX (total unsigned flux), MEANJZH (mean current helicity), ABSNIZH (absolute value of the net current helicity), SAVNCPP (sum of the modulus of the net current per polarity), MEANALP (mean characteristic twist parameter), MEANSHR (mean shear angle), TOTUSJH (total unsigned current density), MEANGAM (mean angle of field from radial), MEANGBT (mean gradient of vertical field), MEANJZD (mean vertical current density), AREA_ACR (area of strong field pixels in the active region), R_VALUE (sum of flux near polarity inversion line), MEANGBT (mean gradient of total field), and MEANGBH (mean gradient of horizontal field). Because the features have different units and scales, we normalize the feature values as done in Liu et al. (2019). Data samples with incomplete features are excluded from our data set (Bobra & Ilonidis 2016).

Our proposed RNNs require labeled training samples. We survey M- and X-class flares that occurred in the period between 2010 and 2019 May, using the GOES X-ray flare catalogs provided by the National Centers for Environmental Information (NCEI), and select M- and X-class flares with identified ARs in the NCEI flare catalogs. As in Bobra & Ilonidis (2016), flares that are outside ±70° of the central meridian during the peak time of the GOES X-ray flux are excluded from our data set. Flares where the (1) absolute value of the radial velocity of SDO is larger than 3500 m s⁻¹ or (2) HMI data are of low quality as described in Hoeksema et al. (2014) are also excluded from our data set. In this way, we exclude data with noise or low quality, and keep data of high quality in our study. In addition, we extract data from a NASA Space Weather Research Center database called Space Weather Database of Notifications, Knowledge, Information (DONKI)⁵ to label whether or not any given flare produced a CME. This yields a database of 129 M- and X-class flares that are associated with CMEs and 610 M- and X-class flares that are not associated with CMEs.

3. Methodology

3.1. Prediction Task

Following Bobra & Ilonidis (2016), we intend to use past observations of a flaring AR to predict its future CME productivity. Specifically, we want to solve the following binary classification problem: will an AR that produces an M- or X-class flare within the next T hours also produce a CME associated with the flare? We consider T ranging from 12 to 60 in 12 hr intervals. These prediction times are commonly used by researchers (Ahmed et al. 2013; Bobra & Ilonidis 2016; Inceoglu et al. 2018).

Our data set contains data samples collected within the T hours prior to the peak time of an M- or X-class flare regardless of whether the flare produces a CME. Those data samples collected within the T hours prior to the peak time of an M- or X-class flare that produces a CME belong to the positive class; the other data samples belong to the negative class. Data samples collected in years 2010–2014 are used for training, and those in years 2015–2019 are used for testing. Thus, the training set and test set are disjoint, and hence our proposed

|                      | 12 hr | 24 hr | 36 hr | 48 hr | 60 hr |
|----------------------|-------|-------|-------|-------|-------|
| Training             | 3387  | 16,960| 6323  | 27,281| 9109  | 34,994| 11,613| 41,402| 13,775| 46,994|
| Testing              | 550   | 762   | 922   | 1059  | 1214  | 1253  | 1540  | 1360  | 1814  | 1483  |

Table 1: Numbers of Positive and Negative Samples Collected for Different Hours Used in This Study

http://kauai.ccmc.gsfc.nasa.gov/DONKI/

http://jssc.stanford.edu/

- SDO: Solar Dynamic Observatory
- HMI: Helioseismic and Magnetic Imager
- NCEI: National Centers for Environmental Information
- DONKI: Database of Notifications, Knowledge, Information

Inceoglu et al. (2018)
RNNs will make predictions on ARs that they have never seen before. Table 1 summarizes the numbers of positive and negative samples for different T values used for training and testing respectively.

If a data sample is missing at some time point or if there are insufficient data samples within the T hours prior to the peak time of an M- or X-class flare, we adopt a zero-padding strategy by adding synthetic data samples with zeros for all feature values to yield a complete, gapless time-series data set. This zero-padding method is used after normalizing the feature values. Therefore, the zero-padding method does not affect the normalization procedure. For a given time point t and an AR that produces an M- or X-class flare within the next T hours of t, the proposed RNNs predict whether or not the flare will initiate a CME.

3.2. Prediction Methods

We employ two kinds of RNNs: a GRU network (Cho et al. 2014; Goodfellow et al. 2016) and an LSTM network (Hochreiter & Schmidhuber 1997; Goodfellow et al. 2016). A GRU contains three interactive parts—a memory content, an update gate, and a reset gate—as illustrated in Figure 1, where boldface is used for matrix notations. The new memory content $h_t$ is updated by the previous memory content $h_{t-1}$ and the candidate memory content $\tilde{h}_t$ as follows (Cho et al. 2014; Goodfellow et al. 2016):

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t,$$

where the update gate $z_t$ that determines how much of the past information from previous time steps needs to be passed to the future is calculated as follows (Cho et al. 2014; Goodfellow et al. 2016):

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + B_z),$$

and the reset gate $r_t$ that determines how much of the past information to forget is computed as follows (Cho et al. 2014; Goodfellow et al. 2016):

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + B_r).$$

Here $x_t$ represents the input vector at time step t. The candidate memory content $\tilde{h}_t$ is computed as follows (Cho et al. 2014; Goodfellow et al. 2016):

$$\tilde{h}_t = \tanh(W_h \cdot [r_t \odot h_{t-1}, x_t] + B_h).$$

In the above equations, $W$ and $B$ contain weights and biases respectively, which need to be learned during training; $\cdot$ denotes the concatenation of two vectors; $\sigma(\cdot)$ is the sigmoid function, i.e., $\sigma(z) = \frac{1}{1 + e^{-z}}$; tanh(·) is the hyperbolic tangent function, i.e., tanh(z) = $\frac{e^z - e^{-z}}{e^z + e^{-z}}$; $\odot$ denotes the Hadamard product (element-wise multiplication).

Our GRU network contains a GRU layer with m GRUs (in the study presented here, m is set to 20). We add an attention layer with m neurons above the GRU layer to focus on information in relevant time steps as done in Liu et al. (2019). We then add a fully connected layer with 100 neurons on top of the attention layer. Finally, the output layer with one neuron, which is activated by the sigmoid function, produces predicted values. Our LSTM network is similar to the GRU network except that the GRU layer is replaced by an LSTM layer with m LSTM cells. This is reminiscent of the LSTM network presented in Liu et al. (2019).

Let $x_t$ represent the data sample collected at time point t. During training, for each time point t, we take m consecutive data samples $x_{t-m+1}, x_{t-m+2}, ..., x_{t-1}, x_t$ from the training set and use the m consecutive data samples to train the proposed RNNs including the GRU and LSTM networks. The label of these m consecutive data samples is defined to be the label of the last data sample $x_t$. Thus, if $x_t$ belongs to the positive class, then the input sequence $x_{t-m+1}, x_{t-m+2}, ..., x_{t-1}, x_t$ is defined as positive; otherwise the sequence is defined as negative. Because the data samples are collected continuously at a cadence of 12 minutes and missing values are filled up by our zero-padding strategy, the input sequence spans m/5 hr.

Also during training, we use a weighted cross-entropy cost function for optimizing model parameters, where the cost function is computed as follows (Goodfellow et al. 2016):

$$J = \sum_{n=1}^{N} \omega_0 y_n \log(\hat{y}_n) + \omega_1 (1 - y_n) \log(1 - \hat{y}_n).$$

Here, N is the total number of sequences each having m consecutive data samples in the training set, $\omega_0$ and $\omega_1$ are the weights of the positive and negative classes respectively, which are derived from the ratio of the sizes of the positive and negative classes with more weight given to the minority class.\footnote{Refer to the training data in Table 1. The minority class is the positive class.}

We use $y_n$ and $\hat{y}_n$ to denote the observed probability (which is equal to 1 if the nth sequence belongs to the positive class) and the estimated probability of the nth sequence, respectively.

The proposed RNN methods are implemented in Python, TensorFlow, and Keras. A mini-batch strategy (Goodfellow et al. 2016) is used to achieve faster convergence during backpropagation. The optimizer used is RMSprop, which is a method for gradient descent, where the learning rate is set to 0.001. The batch size is set to 256 and the number of epochs is set to 20. The length of each input sequence, m, is set to 20, meaning that every time 20 consecutive data samples are used as input to our RNNs. The hyperparameter values, the optimizer, and the cost function in Equation (5) are chosen to maximize the TSS scores to be defined in section 4.1.
During testing, to predict whether a given AR that produces an M- or X-class flare within the next \( T \) hours of a time point \( t \) will also produce a CME associated with the flare, we take \( x_t \) and its preceding \( m-1 \) data samples, and then feed the \( m \) consecutive test data samples \( x_{t-m+1}, x_{t-m+2}, \ldots, x_{t-1}, x_t \) into the trained RNNs. The output of the RNNs, i.e., the predicted result, is a scalar number with a value of 1 or 0, indicating that \( x_t \) is positive (i.e., the AR will also produce a CME associated with the flare) or \( x_t \) is negative (i.e., the AR will not produce a CME associated with the flare). This value is determined by comparing the probability calculated by the sigmoid function in the output layer of the RNNs with a threshold. If the probability is greater than or equal to the threshold, then \( x_t \) is predicted to be positive; otherwise \( x_t \) is predicted to be negative. It should be pointed out that the way we use the \( m \) consecutive test data samples \( x_{t-m+1}, x_{t-m+2}, \ldots, x_{t-1}, x_t \) to predict whether a given AR that produces an M- or X-class flare within the next \( T \) hours of a time point \( t \) will also produce a CME associated with the flare is different from the previously published machine learning methods (Bobra & Ilonidis, 2016), which used only the test data sample \( x_t \) to make the prediction.

4. Results

4.1. Metrics and Experimental Setup

Given an AR that produces an M- or X-class flare within the next \( T \) hours of a time point \( t \) and a data sample \( x_t \) observed at time point \( t \), we define \( x_t \) to be a true positive (TP) if our RNNs predict that \( x_t \) is positive, and \( x_t \) is indeed positive, i.e., the M- or X-class flare produces, or is associated with, a CME. We define \( x_t \) to be a false positive (FP) if our RNNs predict that \( x_t \) is positive while \( x_t \) is actually negative, i.e., the M- or X-class flare does not produce, or is not associated with, a CME. We say \( x_t \) is a true negative (TN) if our RNNs predict \( x_t \) to be negative and \( x_t \) is indeed negative; \( x_t \) is a false negative (FN) if our RNNs predict \( x_t \) to be negative while \( x_t \) is actually positive. We also use TP (FP, TN, FN, respectively) to represent the total number of true positives (false positives, true negatives, false negatives, respectively) produced by our RNNs.

The performance metrics used in this study include the following:

\[
\text{Recall} = \frac{TP}{TP + FN},
\]

\[
\text{Precision} = \frac{TP}{TP + FP},
\]

\[
\text{Accuracy (ACC)} = \frac{TP + TN}{TP + TN + FP + FN},
\]

\[
\text{Heidke Skill Score (HSS)} = \frac{2(TP \times TN - FP \times FN)}{(TP + FN)(FN + TN) + (TP + FP)(FP + TN)},
\]

\[
\text{True Skill Statistics (TSS)} = \frac{TP}{TP + FN} - \frac{FP}{TN + FP}.
\]

The HSS (Heidke, 1926) is used to measure the fractional improvement of our prediction over the random prediction (Florios et al., 2018). The TSS score is the recall minus the false alarm rate (Bloomfield et al., 2012). We also use the area under the curve (AUC) in a receiver operating characteristic (ROC) curve (Marzban 2004), which represents the degree of separability, indicating how well a method is capable of distinguishing between two classes with the ideal value of one. These performance metrics are commonly used when dealing with binary classification problems such as the one at hand as defined at the beginning of Section 3.1. In general, the larger HSS, TSS, and AUC score a binary classification method has, the better performance the method achieves.

To gain a better understanding of the behavior of the proposed RNNs, we adopt the following cross-validation methodology. We partition the training (test, respectively) set into 10 equal-sized folds. For every two training (test, respectively) folds \( i \) and \( j \), \( i \neq j \), fold \( i \) and fold \( j \) are disjoint; furthermore, fold \( i \) and fold \( j \) contain approximately the same number of positive training (test, respectively) data samples and approximately the same number of negative training (test, respectively) data samples. In the \( i \)th run, \( 1 \leq i \leq 10 \), all training data samples except those in training fold \( i \) are used to train a model, and the trained model is used to make predictions on all test data samples except those in test fold \( i \). We calculate the performance metric values based on the predictions made in the \( i \)th run. There are 10 runs. The means and standard deviations over the 10 runs are calculated and recorded.

4.2. Feature Assessment

We conduct a series of experiments to analyze the importance of each of the 18 features studied here using the cross-validation methodology described above and the feature assessment method introduced in Section 4.3 of Liu et al. (2019). Each time only one feature is used to make predictions. The probability threshold used by our RNNs is set to maximize the TSS score in each run. There are 10 runs and the corresponding mean TSS score is calculated and recorded. There are 18 features, so 18 mean individual TSS scores are recorded. These 18 mean individual TSS scores are sorted in descending order, and the 18 corresponding features are ranked from the most important (with the highest mean individual TSS score) to the least important (with the lowest) accordingly.

Table 2 presents the 18 features ranked by our GRU and LSTM networks respectively for different \( T \) values where \( T \) ranges from 12 to 60 in 12 hr intervals. It can be seen from the table that MEANPOT, SHRTG45, ABSNJZH, and SAVNCPP are consistently ranked in the top 10 list by both LSTM and GRU networks. In particular, MEANPOTS plays the most important role in CME prediction, and is ranked as the top one in all cases. Other features such as TOTPOT, USFLUX, MEANJZH, MEANALP, and MEANSHR also show strong predictive power and are ranked in the top 10 list in most cases. Compared to the top 10 lists of Bobra & Ilonidis (2016), who used a different feature ranking method for \( T = 24 \), 48 (see Table 1 of Bobra & Ilonidis 2016), these lists overlap to some extent (seven features simultaneously occur in the top 10 lists given by Bobra & Ilonidis 2016 and our methods).

Next, mean cumulative TSS scores are calculated according to the ranked features. Specially, the mean cumulative TSS score of the top \( k \) features is calculated for \( 1 \leq k \leq 18 \), most important features are equal to the mean TSS score of using the top \( k \) most important features altogether for CME prediction. We calculate the mean cumulative TSS scores for \( T = 12, 24, 36, 48, \) and 60 hr. It is found that using all the 18 features together does not yield the
highest mean cumulative TSS scores. In fact, using the top 16, 12, 9, 14, and 5 features for $T = 12, 24, 36, 48,$ and 60 hr respectively yields the highest mean cumulative TSS scores, achieving the best performance for our GRU network. Using the top 15, 12, 8, 15, and 6 features for $T = 12, 24, 36, 48,$ and 60 hr respectively yields the highest mean cumulative TSS scores for our LSTM network. This happens probably because low ranked features are noisy features, and using them may deteriorate the performance of the methods. In subsequent experiments, we use the best features for each method.

### 4.3. Performance Comparison

We compare our proposed RNNs with three closely related machine learning methods including an MLP (Inceoglu et al. 2018), an SVM (Bobra & Ilonidis 2016), and the random forest algorithm (RF) (Liu et al. 2017). MLP and SVM have been previously used for CME prediction (Bobra & Ilonidis 2016; Inceoglu et al. 2018). RF has been used in flare prediction with good performance (Liu et al. 2017, 2019; Florios et al. 2018). These three machine learning methods are inherently probabilistic forecasting models (Florios et al. 2018) in the sense that each of them predicts a probability. Since we are dealing with a binary classification problem, as defined at the beginning of Section 3.1, we convert each probabilistic forecasting model into a binary classification model (Jonas et al. 2018; Nishizuka et al. 2018) by comparing the predicted probability with a threshold. If the predicted probability is greater than or equal to the threshold, then the model predicts that a flare will produce, or is associated with, a CME; otherwise, the model predicts that the flare will not produce a CME.

MLP consists of an input layer, an output layer, and two hidden layers both with 200 neurons. SVM uses the radial basis function (RBF) kernel. RF has two parameters: mtry (number of features randomly selected to split a node) and ntree (number of trees to grow in the forest). We vary the values of ntree $\in \{300, 500, 1000\}$ and mtry $\in \{2, 8\}$, and set ntree to 500 and mtry to 3. The hyperparameter and parameter values used by these three related machine learning methods are chosen to maximize their TSS scores. As in the proposed RNNs, we use data samples collected in years 2010–2014 for training and data samples in years 2015–2019 for testing. To deal with the imbalanced data sets described in Table 1, we give more weight to the minority class during training, as done for the RNNs. The same cross-validation methodology as described in Section 4.1 is adopted to evaluate the performance of the three related machine learning methods.

Table 3 presents the confusion matrix, which lists mean TP, FP, FN, TN (with standard deviations enclosed in parentheses)
of the five machine learning methods for different $T$ values, where $T$ ranges from 12 to 60 hr in 12 hr intervals. The probability thresholds used by the machine learning methods are set to maximize their TSS scores. Table 4 presents the mean performance metric values (with standard deviations enclosed in parentheses) of the five machine learning methods. The best performance metric values are highlighted in boldface. Figure 2 shows the ROC curves for the five machine learning methods. It can be seen from Table 4 and Figure 2 that our GRU and LSTM networks perform better than the three related machine learning methods in terms of HSS, TSS, AUC, and ROC curves. There is no clear distinction between the GRU and LSTM networks. These results indicate that both of the proposed RNNs are suitable for solving the binary classification problem at hand.

4.4. Probabilistic Forecasting

The three related machine learning methods are inherently probabilistic forecasting models. Our proposed RNNs can be easily converted from a binary classification model to a probabilistic forecasting model as follows. Instead of comparing the probability, calculated by the sigmoid function in the output layer of the RNNs, with a threshold, the RNNs simply output the probability. For a given time point $t$ and an AR that will produce an M- or X-class flare within the next $T$ hours of $t$, this output now represents a probabilistic estimate of how likely it is that the flare will initiate a CME.

We use the Brier Score (BS) (Brier 1950) and the Brier Skill Score (BSS) (Wilks 2010) to quantitatively assess the performance of a probabilistic forecasting model, where

$$
BS = \frac{1}{N} \sum_{n=1}^{N} (y_n - \hat{y}_n)^2,
$$

$$
BSS = 1 - \frac{\text{mean}(y - \hat{y})^2}{\text{mean}(y - \bar{y})^2}.
$$

Here $N$ is the total number of sequences each having $m$ consecutive data samples in the test set, $y_n$ and $\hat{y}_n$ denote the observed probability and the estimated probability of the $n$th sequence respectively as defined in Equation (5), and $\bar{y} = \frac{1}{N} \sum_{n=1}^{N} y_n$ is the average value of all the observed probabilities. The values of BS range from 0 to 1, with the perfect score being 0. The values of BSS range from minus infinity to 1, with the perfect score being 1.

Table 5 presents the mean BS and BSS scores and standard deviations of the five machine learning methods for different $T$ values where $T$ ranges from 12 to 60 hr in 12 hr intervals. It can be seen from the table that the proposed GRU and LSTM networks are comparable, and again outperform the three related machine learning methods in terms of BS and BSS. These results in Tables 4 and 5 suggest that our RNNs work well when used as either binary classification models or probabilistic forecasting models.
5. Discussion and Conclusions

We develop two RNNs, where one is a GRU network and the other is an LSTM network, for CME prediction. Given a time point \( t \) and an AR that produces an M- or X-class flare within the next \( T \) hours of \( t \) where \( T \) ranges from 12 to 60 in 12 hr intervals, our RNNs, when used as binary classification models, can predict whether the AR will also produce a CME associated with the flare. In addition, our RNNs, when used as probabilistic forecasting models, can produce a probabilistic estimate of how likely it is that the M- or X-class flare will initiate a CME.

We build a data set of samples, gathered from the JSOC website, in the period from 2010 to 2019 May; each data sample has 18 magnetic parameters provided by SHARP. We use the data samples during the years 2010–2014 for training, and the data samples during the years 2015–2019 for testing. The training set and test set are disjoint, and hence our RNNs can make predictions on ARs that they have never seen before. With extensive experiments, we evaluate the performance of the RNNs and compare them with three closely related machine learning methods—MLP (Inceoglu et al. 2018), RF (Liu et al. 2017), and SVM (Bobra & Ilonidis 2016)—using different performance metrics. All these machine learning methods including ours can be used as binary classification models or probabilistic forecasting models. The main results are summarized as follows.

1. Solar data samples in an AR are modeled as time series here. Unlike the previous method (Bobra & Ilonidis 2016), which uses one data sample gathered at the time point \( t \) to make a prediction, our RNNs use the \( m \)
data samples gathered at \( t \) and the preceding \( m - 1 \) time points to make prediction (\( m \) is set to 20 in the study presented here). To our knowledge, this is the first time that RNNs, which can capture dependences in the temporal domain of the data samples, are used for CME prediction.

2. We evaluate the importance of each of the 18 magnetic features, or features, used in this study. Our experimental results show that using the most important 5–16 features, depending on different \( T \) values, can achieve better performance than using all the 18 features together. These results are consistent with the literature, which indicate that using fewer, high-quality features is often better than using all, including low-quality features (Alpaydin 2016; Bobra & Ilonidis 2016; Goodfellow et al. 2016). Developing effective feature ranking and selection procedures has been an active area of research in machine learning and related fields. In general, to find the optimal feature subset among \( n \) features, one has to try all \( 2^n - 1 \) combinations of the \( n \) features. This exhaustive search algorithm becomes impractical when \( n \) is large, as in our case where \( n = 18 \). Consequently, various heuristics based on statistics, randomization, optimization, sampling, clustering, machine learning, evolutionary computation, genetic algorithms, branch and bound algorithms, and principal component analysis, to name a few, have been developed (Mitra et al. 2002; Guevrey et al. 2003; Guyon & Elisseeff 2003; Liu et al. 2004, 2020; Oh et al. 2004; Olden et al. 2004; Somol et al. 2004; Yoon et al. 2005; Chandrashekar & Sahin 2014; Xue et al. 2016; Fisher et al. 2018; Al-Tashi et al. 2019). In this work, we use the single feature testing heuristic originating from Laing et al. (2012) to rank and select features. In related work, Bobra & Ilonidis (2016) used an F-score heuristic to rank and select features. Identifying the best feature selection heuristic with the optimal performance in terms of accuracy and execution time remains an open problem. We plan to investigate this problem further in the future.

3. Our GRU and LSTM networks are comparable; there is no clear distinction between them in terms of the performance metrics studied here. Both of the networks outperform the related machine learning methods of MLP, RF, and SVM whether they are used as binary classification models or probabilistic forecasting models. These findings are based on the data collection scheme in which data samples in years 2010–2014 are used for training and those in years 2015–2019 are used for testing. To further understand the behavior of the machine learning methods, we have performed additional experiments as follows. In each experiment, data samples collected in one of the ten years during the period 2010–2019 are used for testing and data samples in all the other nine years together are used for training. There are ten years in the period and hence there are ten experiments. The average values of the performance metrics are calculated. The results based on these additional experiments are consistent—our GRU and LSTM networks are comparable, and both of them perform better than the related machine learning methods MLP, RF, and SVM.

Based on our experimental results, we conclude that the proposed RNNs are valid methods for CME prediction. It should be pointed out that the CME prediction is performed based on the assumption that an M- or X-class flare already exists. In practice, how does one know whether an AR will produce an M- or X-class flare within the next \( T \) hours of some time point \( t \)? This question can be answered by using a flare prediction method (e.g., Liu et al. 2017, 2019; Florios et al. 2018; Jonas et al. 2018; Nishizuka et al. 2018). Our software package has two components. The first component is our previously developed deep learning program (Liu et al. 2019), where one can use the program to predict whether there is an M- or X-class flare within the next \( T \) hours of \( t \). If the answer is yes, then one can use the RNN tools developed here to predict whether the flare produces, or is associated with, a CME. In future work, we plan to further extend the software package to predict other events (e.g., filament eruptions, SEPs).

We thank the referees for very helpful and thoughtful comments. We also thank the team of \( SDO/HMI \) for producing vector magnetic field data products. The flare catalogs were prepared by and made available through NOAA NCEI. The CME event records were provided by DONKI. The related machine learning methods studied here were implemented in Python. This work was supported by NSF grant AGS-1927578. C.L. and H.W. acknowledge the support of NASA under grants NNX16AF72G, 80NSSC17K0016, 80NSSC18K0673, and 80NSSC18K1705.

**ORCID iDs**

Chang Liu \( \text{https://orcid.org/0000-0002-6178-7471} \)
Jason T. L. Wang \( \text{https://orcid.org/0000-0002-2486-1097} \)
Haimin Wang \( \text{https://orcid.org/0000-0002-5233-565X} \)

**References**

Ahmed, O. W., Qubwai, R., Colak, T., et al. 2013, SoPh, 283, 157
Alpaydin, E. 2016, Machine Learning: The New AI (Cambridge, MA: MIT Press)
Al-Tashi, Q., Abdulkadir, S. J., Rais, H. M., Mirjalli, S., & Alhussian, H. 2019, IEEE Access, 7, 39496
Baker, D. N., Daly, E., Daglis, I., Kappenman, J. G., & Panasyuk, M. 2004, SpWea, 2, S02004
Baumgartner, C., Thalmann, J. K., & Veronig, A. M. 2018, ApJ, 853, 105
Berkebile-Stoiser, S., Veronig, A. M., Bein, B. M., & Temmer, M. 2012, ApJ, 753, 88
Bloomfield, D. S., Higgins, P. A., McAteer, R. T. J., & Gallagher, P. T. 2012, ApJ, 747, L41
Bobra, M. G., & Ilonidis, S. 2016, ApJ, 821, 127
Bobra, M. G., Sun, X., Hoeksema, J. T., et al. 2014, SoPh, 289, 3549
Brier, G. W. 1950, MWRv, 78, 1
Chandrashekar, G., Dimopoulos, I., & Lek, S. 2003, Ecological Modelling, 160, 249
Chandrashekar, G., & Sahin, F. 2014, CSE, 40, 16
Cho, K., van Merriënboer, B., Gulcehre, C., et al. 2014, in Proc. 2014 Conf. on Empirical Methods in Natural Language Processing (EMNLP), ed. A. Moschitti, B. Pang, & W. Daelemans (Doha: Association for Computational Linguistics), 1724
DeVore, C. R., & Antiochos, S. K. 2008, ApJ, 680, 740
Fisher, A., Rudin, C., & Dominici, F. 2018, arXiv:1801.01489
Florios, K., Kontogiannis, I., Park, S.-H., et al. 2018, SoPh, 293, 28
Guevrey, M., Dimopoulos, I., & Lek, S. 2003, Ecological Modelling, 160, 249
Goodfellow, I. J., Bengio, Y., & Courville, A. C. 2016, Deep Learning (Cambridge, MA: MIT Press)
Gosling, J. T. 2013, in Coronal Mass Ejections, ed. N. Crooker, J. A. Joselyn, & J. Feynman, Vol. 99 (Washington, DC: AGU), 9
Guyon, I., & Elisseeff, A. 2003, Journal of Machine Learning Research, 3, 1157
Harrison, R. A. 1995, A&A, 304, 585
Hexilek, P. 1926, Geografiska Annaler, 8, 301
