Solar Flare Forecasting with Deep Learning-based Time Series Classifiers

Anli Ji*, Junzhi Wen†, Rafal Angryk‡, Berkay Aydin§

Department of Computer Science, Georgia State University, Atlanta, GA*†‡§
Email: aji1@student.gsu.edu*, jwen6@student.gsu.edu†, rangryk@cs.gsu.edu‡, baydin2@cs.gsu.edu§

***Abstract*—Over the past two decades, machine learning and deep learning techniques for forecasting solar flares have generated great impact due to their ability to learn from a high dimensional data space. However, lack of high quality data from flaring phenomena becomes a constraining factor for such tasks. One of the methods to tackle this complex problem is utilizing trained classifiers with multivariate time series of magnetic field parameters. In this work, we compare the exceedingly popular multivariate time series classifiers applying deep learning techniques with commonly used machine learning classifiers (i.e., SVM). We intend to explore the role of data augmentation on time series oriented flare prediction techniques, specifically the deep learning-based ones. We utilize four time series data augmentation techniques and couple them with selected multivariate time series classifiers to understand how each of them affects the outcome. In the end, we show that the deep learning algorithms as well as augmentation techniques improve our classifiers performance. The resulting classifiers' performance after augmentation outplayed the traditional flare forecasting techniques.**

***Index Terms*—solar flare prediction, deep neural networks, data augmentation**

I. Introduction

Solar flares are one of the solar magnetic phenomena that suddenly bursts out large amounts of electromagnetic energy from the solar atmosphere. When they combine with other solar activities like coronal mass ejections (CMEs), they may potentially provoke devastating geomagnetic storms, which can cause a massive scale of electrical blackouts, disrupt power grids and radio communications, and even damage sensitive electronic devices in space. Flare phenomena usually occur when the intense magnetic fields of the Sun tangle up and become unstable. The relationship between the photospheric magnetic field and flaring (or eruptive) phenomena still remains questioned. Therefore, researchers have put efforts on predicting the flare event prior to its occurrence to avoid crucial costs and destruction.

Over the past two decades, many machine learning-based algorithms attempted to remedy the problem [1] by issuing binary predictions on major flares (i.e., M- and X-class) instead of considering where the relationship could be derived from. However, most of the predictions are generated by applying point-in-time values within the maps of magnetic field strength (i.e., magnetograms) meaning that only one individual value in multiple physical parameters observations is representative for each solar flare event [2] [3]. Such approaches could be limited in predicting power because it does not take the temporal evolution aspect of these parameters into account. Within the recent proposed classification algorithms, Deep Neural Networks (DNNs) have achieved remarkable performance in the field of computer vision [4], image recognition [5], and natural language processing (NLP) [6] [7] [8]. As one of the modern machine learning architectures, this approach has the ability to automatically transform input data into a relatively more abstract representation and use the composite representation to perform feature extraction or classification during the learning process. This benefits especially for prediction in high dimensional space since it requires less human cognitive process.

Many researchers focus on improving the deep learning model by improving its generalization ability, which is the performance difference of a model between training and testing phase. This means that models with poor generalization ability are easier to cause overfitting than others. To understand or predict solar flares better and issue accurate predictions, there is a need for more data since the high performance of deep learning architecture relies heavily on data volume. However, one major issue for data-driven solar flare forecasting is the limited amounts of high quality data, especially with the flaring instances. Within the most recent records, less than 20% of the active regions produce large flaring (e.g., $\geq$M) instances [9], which lead to insufficient features derived from these instances for training and validating complex prediction models.

Data augmentation is a way to synthetically modify the raw data for the models to recognize generalizable features and hidden patterns and therefore improve the overall performance. The techniques implemented for data augmentation aim to improve the quantity of the datasets so that better models can be trained. With augmented data, models can cover those unexplored input spaces, and therefore improve the generalization ability. Inspired by this idea, our work is further extended to seek the feasibility of applying data augmentation techniques to optimize the performance of different deep learning models. Four most common time series data augmentation techniques are selected and five deep learning architectures are refined to monitor the performance compared to the baseline model that is trained without applying any augmentation techniques. By this, we conducted a multivariate time series classification by utilizing the time series data from a well-known solar flare prediction dataset [10] and synthetically modified the time series instances with various data augmentation techniques.

The rest of the paper is organized as follows. In Section II, we discuss some related work on well-known flare prediction with a focus on deep learning and utilization of data augmentation. In Section III, we explain our research methodology on data collection and preparation as well as our training and evaluation processes. In Section IV, we present our experimental evaluation results for our models. In Section V and VI, we provide a summary of our findings and discuss some future work avenues.

II. Related Work

In the field of solar flare forecasting, there are various studies on training deep learning models on magnetic field data products, either the original magnetic field rasters or the derived metadata products. One major source for such prediction is the Michelson Doppler Imager (MDI) observation program from Solar and Heliospheric Observatory (SOHO) [11], which has been superseded in 2011 for the Helioseisemic and Magnetic Imager (HMI) onboard the Solar Dynamics Observatory (SDO) [12]. As for working with the magnetic field rasters, the study proposed by [13] transformed the X-ray flux time series data from GOES at one-minute cadence into 64×64 Markov Transition Field images and provided the Convolutional Neural Network (CNNs) with these image data to predict flare occurrences. FlareNet [14], a framework designed for experimentation with solar flare prediction, utilizes high dynamic image data directly from SDO where each pixel contains a high dynamic range of flux that can potentially affect gradient updates and encourage overfitting. The problem is addressed by treating each timestamp of temporally adjacent images captured in the same spatial locations as additional image channels and appending side channel information vectors to the fully connected layers to avoid memorization. Sumi et al. [15] also uses soft X-ray data with longer wavelength at one-minute cadences from Geostationary Operational Environmental Satellite (GOES) to predict X-ray flux in continuous target space and evaluates with three DNN architectures. In their research, the hierarchical dense residual network has clearly better performance compared to the baselines and other DNN architectures.

Deep Flare Net-Reliable (DeFN-R) [16], a probabilistic flare forecasting model composed of multilayer perceptrons, uses 79 extracted features from each active region (i.e, AR) observed by the multiwavelength emissions and annotated with flare occurrence labels (i.e, X-, M-, and C-classes). This model forecasts the maximum flare classes of the next 24 hours as well as the occurrence probability of each event. LSTM_DNN Flare Net (LDFN), in [17], also uses a derived dataset where its experimental dataset contains multivariate time series (MVTS) of magnetic field metadata. This approach combines Long Short-Term Memory (LSTM) [18] architecture with multi-scale skip connections. The overall interpretability of this architecture is improved by introducing human cognitive processes based on physical definitions that were extracted and combined with similar ones using deep learning models. Hui et al. [2] proposed an ordinal logistic regression model by using three predictive parameters derived from the MDI magnetograms, with categories of the maximum magnitude of flares, to predict the probability of a given AR in the next twenty-fours hours. As in [19], Huang et al. generates a meta dataset that obtained multiple instruments (SDO/HMI and SOHO/MDI) magnetic field measurements and evaluates with a four layered DNN.

As solar flares are relatively less frequent events, the scarcity of the major solar flares often result in an extreme class-imbalance ratio where the number of flaring samples contained (usually the intensity levels above M-class in NOAA/GOES flare classification) are much higher than the number of non-flaring samples contained (with intensity levels of C, B, and flare-quiet regions). Even with the available data, there still exists the problem of limited high quality data instances for models like deep neural networks, which relies heavily on large scale data. To alleviate these two problems, oversampling and undersampling are the typical approaches to remedy the class imbalance issue [9] [20] [21]. However, a big portion of data is left out when performing undersampling, and a large amount of data is replicated when oversampling is applied [21]. Data augmentation, on the other hand, tackles this class-imbalance issue by modifying existing instances to generate potentially more generalizable features. In this way, more data is introduced to the models and the generalization capabilities of these classifiers can be improved by expanding the model's decision boundaries [22]. Many original proposals of the CNN architectures take advantage of data augmentation. AlexNet [23], for example, archived a benchmark record on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) dataset by using cropping, mirroring, and color augmentation. Other network architectures such as DenseNet [24], Visual Geometry Group network [25], or Residual Networks [26] also uses augmentation.

Our work implements various data augmentation techniques for solar flare prediction via five deep learning models. This not only provide a multivariate schema for such deep learning models but also intend to improve their performance using augmented data instances. We hope this work affords researchers and practitioners in related fields more exposure to multivariate time series analysis for space weather forecasting.

III. Methodology

As solar flares are mainly electromagnetic phenomena, they are usually predicted by employing the adequate magnetic field parameters. Inspired by Bobra and Couvidat that applied a large set of SDO/HMI Vector magnetic field data for predicting M- and X-class flares [27], Space Weather Analytics for Solar Flares (SWAN-SF) [10] is a comprehensive multivariate time series dataset consists of physical parameters derived from solar photospheric magnetic fields in the Space weather HMI Active Region Patches (SHARPs) data series.

There are five partitions from the SWAN-SF dataset, where every partition contains a similar number of instances labeled with X- and M-class flares. All data in the partitions are time segmented, which means that all data points are temporally non-overlapping time series slices and none of the active regions cross in time or space. In this study, we used Partition 1, 2, 3, and 5 for training purposes and excluded Partition 4 for evaluating the model performance since this partition has a relatively more representative class imbalance ratio compared to the overall dataset. Every reported flare in each partition is associated with a designated active region identifier and related time series data of the magnetic field parameters extracted from this active region. As flare intensity can be classified logarithmically by their peak X-ray flux, we assigned level X and M flare as the flaring classes while level C, B, and flare-quiet regions (labeled as N) as the non-flaring classes. Additionally, five parameters employed in this study are USFLUX (Total unsigned flux), TOTUSJH (Total unsigned current helicity), ABSNJZH (Absolute value of the net current helicity), SAVNCPP (Sum of the modulus of the net current per polarity) and TOTBSQ (Total magnitude of Lorentz force), which are shown to be most representative of the solar activity in [27].

This dataset will be utilized as our benchmark dataset throughout this study for consistency between our results.

A. Models Selection

In order to be compatible with the five selected physical parameters, we choose the commonly used machine learning classifiers – Support Vector Machine (i.e., SVM) [28] and five well-known deep learning classifiers that are able to classify multivariate time series instances.

Baseline: SVM with Summary Statistics
We adopt a SVM classifier, which makes use of summary statistical parameter values derived from our benchmark SWAN-SF dataset. From each original physical parameter employed above, a total of 43 summary statistics data is generated based on every individual time series. This results in a frame of data with the same amount of flaring versus non-flaring instances in each partition.

One Direction CNN (ODCNN) is designed as one of the most common CNNs where convolutional layers are applied on a layer-by-layer architecture to the given time series [29]. In this framework, we create the model based on a sequential classifier meaning that the layers in this classifier are linearly stacked and do not communicate with layers other than the previous and next one. Therefore, the model flexibility is limited as it only focuses from the outputs of previous layer. Each layer contains weights that passed to the posterior layer and is followed by an activation function that allows the model to take nonlinear relationships into account. We set up most standard 5 filters with length of 3 and a Rectified Linear Unit (ReLU) as the activation function [30].

Encoder for time series classification, originally proposed in [31], is a Fully Connected Convolutional Neural Networks (FCNs) that uses dense connections between layers. We generate 128, 256, and 512 filters in the first three convolutional layers with respective lengths of 5, 11, and 21. Each of the convolution layers is operated by an instance normalization [32] and the result from such normalization is then fed into the Parametric Rectified Linear Unit (PReLU) [33] activation function. Furthermore, a dropout operation is applied after each activation function as well as a max pooling with length of 2. The main difference of such architecture from a traditional FCN is replacing the last pooling layer with an attention mechanism that allows the network to learn which parts of the time series are more important for the classification task. This mechanism is implemented by multiplying the input time series with another time series (that has gone through a softmax function). It is worth mentioning that both of the two time series multiplied have the same length, where only the second time series is acting as the weight for the first one. In this way, the network can learn the importance of each timestamp in its corresponding weights.

Residual Network (ResNet) is another architecture originally proposed by [34], which builds shortcut residual connection between consecutive convolutional layers. This architecture reduces the vanishing gradient effect [35] by infusing a linear shortcut between the input and output of each residual block so that the gradient can flow through these shortcuts and make the training smoother. In our case, three residual blocks are used where each residual block contains three convolutions followed by a ReLU activation function. Each of our convolutions consists of 64 filters and the filter lengths are set to 8, 5, and 3, respectively. The output from all three convolutions will be added back to the input of each block and combined as a whole for the final fully-connected layer.

Multi Channel Deep Convolutional Neural Network (MCCNN) was proposed by [36] where the convolutions of the model are parallelized and applied individually to each channel (i.e., dimension) of the input time series. Each channel of the input series will pass two convolutional stages of 8 filters. Each filter's length is 5 and followed by a ReLU activation function as well as a max pooling operation with length of 2. The output from these channels will be concatenated together and fed into a fully-connected layer of 732 neurons. At last, a softmax classifier is generated where the number of neurons matches the number of classes.

Inception [37] is designed to tackle the problem of stacking large numbers of convolutional layers by implementing multiple kernels with different sizes that operate on the same stage. In our case, we set the kernel sizes to 3, 5, 8, 11, 17 and generate a module for saving these kernels. Each of these different convolution layers works differently. For example, convolution size of 1x1 allows the network to learn the patterns across the input series while convolution size of 3x3 (and 5x5) allows the network to learn spatial patterns across all dimensions of the input including height, width, and depth. Similar to the MCCNN mentioned above, this network also ends with a max pooling operation as well as a ReLU activation function.

B. Data Augmentation Techniques

In the second part of this research, we explore the feasibility of using time series augmentation techniques to improve the performance of deep learning-based multivariate time series classifiers. We will use four well-known augmentation techniques [38] described as follows.

Jittering is a type of noise injection, in which a matrix of random values are drawn from a Gaussian distribution and added onto the original data. A general form for using this technique is provided below:

$$x' = x_1 + \epsilon_1, ..., x_t + \epsilon_t, ..., x_T + \epsilon_T \tag{1}$$

where ϵ is the Gaussian noise added to the time series with length of T. In this case, the standard deviation α of the selected Gaussian distribution is a hyperparameter that needs to be determined before training the model. This method increases the generalization of the networks [39] [40] by assuming that the testing data are similar to the training data but only with a difference of a factor of noise. It has been mostly used for testing the robustness of a given model against noise and ideally improve model performance.

On the other end of the spectrum, we also used **smoothing**, which works by applying a weighted average to the original data. This method often reduces noise and can benefit more for models that are strongly impacted by noisy data points.

Scaling is a method based on enlarging (or shortening) the global magnitude with a random scalar. The equation of this method is defined as:

$$x' = \alpha x_1, ..., \alpha x_t, ..., \alpha x_T \tag{2}$$

where α is the random value drawn from the Gaussian distribution. Similar to jittering, scaling also introduces noise to the data. However, instead of adding individual noise directly on each data point, scaling multiplies the entire time series with a scaling parameter α (which is also drawn from a Gaussian distribution). In such cases, the scaling parameter α is the hyperparameter of the model.

Magnitude Warping is another method that aims at changing the magnitude of each time series. The formulation of this method is defined as:

$$x' = \alpha_1 x_1, ..., \alpha_t x_t, ..., \alpha_T x_T \tag{3}$$

where α_1 to α_T is a series interpolated by a cubic spline $S(u)$. The only difference of such method is that each time series magnitude is warped by a smooth curve in range of 0 to 1. This is based on the idea that increasing or decreasing random regions in the time series can add small fluctuations into the dataset. Moreover, instead of many other transformation-based methods that use only one hyperparameter, this method assumes random transformation is practical and uses two pre-defined hyperparameters instead.

IV. Experimental Evaluation

As for the model evaluation, we implemented multiple metrics as well as some essential skill score measurements. In these measurements, TP (true positives), TN (true negatives), FP (false positives), and FN (false negatives) are used in standard settings. Positive stands for occurrence of a large flare (i.e., X- and M-class) while negative stands for comparatively smaller flare and flare-quite region (i.e., C-, B-class, and flare-quite regions). Note here TP represents when the model predicts correctly with the positive class while TN is where the model predicts correctly with the negative class. In space weather forecasting, some classical measures like accuracy and precision should not be considered as a key score to be relied upon because such measure scores can be significantly impacted by the imbalanced class distribution. In the case of substantially larger amount of non-flaring classes than the flaring classes, a classifier can always predict a flare will not occur and still reach a high accuracy score. Similarly, by introducing more negative samples to the dataset while maintaining the same amount of positive samples, the number of FP (false positives) may potentially increase, which will result in a lower precision score.

Two main skill score measurements we used for model evaluation in the experiments are True Skill Statistic score (TSS) and Heidke Skill Score (HSS). True Skill Statistic score (shown in Eq. 4) compares the difference between the probability of detection (Recall) and the probability of false detection (the ratio of inaccurate predictions of non-flaring/negative class over all the actual negative class).

$$TSS = \frac{TP}{TP + FN} - \frac{FP}{FP + TN} \tag{4}$$

Heidke Skill Score (shown in Eq. 5) measures the improvement of the forecast over a class-imbalance-aware random forecast where $P = TP + FN$ and $N = FP + TN$. HSS values range from -1 to +1 where the value of 1 indicates the ideal forecasting and the value of -1 indicates an all incorrect forecasting. The closer the value to 0, the model has less power to distinguish between labels, meaning it has no skill over a class-imbalance-aware random prediction.

$$HSS = \frac{2 \cdot ((TP \cdot TN) - (FN \cdot FP))}{P \cdot (FN + TN) + N \cdot (TP + FP)} \tag{5}$$

Three experiments are designed in this study discovering the performance of time series classifiers as well as how such deep learning models are affected after applying standard and synthetic augmented data instances.

A. Experiment 1: Comparison of selected DNNs with baseline SVM

In the first experiment, we evaluated our baseline SVM model (trained with the summary statistical data) and the other

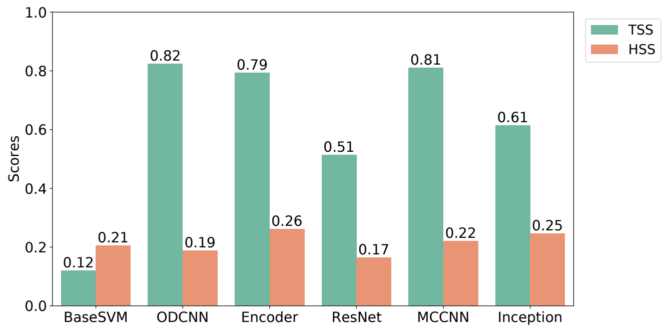

Fig. 1. TSS and HSS results for baseline SVM classifier and the five selected deep learning classifiers.

five selected time series deep learning models (trained with time series data).

From the results shown in Fig 1, models that are trained with deep neural networks have better performance than the baseline SVM model. It demonstrates that TSS increased from 12% to an average of 80% for ODCNN, Encoder, and MCCNN. Although we do not see a significant improvements in HSS, there is still a up to 5% improvement in Encoder, MCCNN, and Inception models. This strongly suggests that the overall performance of the baseline can be improved using deep learning techniques.

B. *Experiment 2: Augmentation*

As for the second experiment, we modified the training samples with each augmentation techniques and trained the five aforementioned deep learning models with these generated samples. This is designed to evaluate the augmentation techniques itself with the original data instances. The total amount of instances are obtained the same as original. Then, we compared these results within our previous models, which are trained without infusing any augmentation instances.

As presented in Fig 2, models that are trained with augmented data perform generally better than the original. However, Inception, ResNet, and Encoder do not have a strong ability to distinguish between flaring and non-flaring classes when they are exposed to additional noise (in the case of jittering experiments) in the dataset. As for comparison, smoothing works the opposite way by reducing potentially noisy data points and achieve a 4% improvement in TSS. It shows a slightly better performance than the baseline as regards to these models. On the other hand, for models that are not affected much by noise (e.g., MCCNN and ODCNN), the augmentation method of jittering actually helps to improve the model performance more than smoothing. Besides jittering, the other two methods of scaling and magnitude warping have similar performance in the results for Encoder and MCCNN as both of these methods scale the range of the data.

C. *Experiment 3: Oversampled synthetic augmentation*

In the third experiment, we generated synthetic augmented instances by stacking our original data with our modified data.

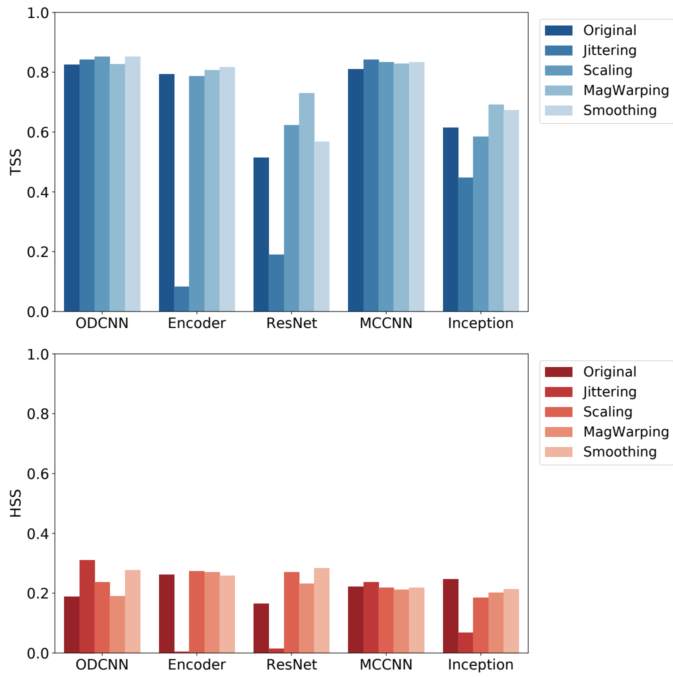

Fig. 2. TSS and HSS results for five selected deep learning classifiers on augmented samples.

The oversampled instances will be fed as training samples into models for training and then evaluated based on the measurements.

As the results shown in Fig. 3, the oversampled synthetic data has the ability to improve certain performance of the model (like ResNet). Both TSS and HSS scores for ResNet have a steady increase especially by using smoothing and scaling. However, the augmentation techniques also do not show much effect for models like Encoder, ODCNN, and MCCNN. The average improvement for these three models are less than 5% compared to our baseline model. It is worth noticing that the performance of Inception is not consistent. The model was boosted around 10% in both measuring scores with scaling and magnitude warping, but decreased about 5% with the use of smoothing.

V. Remarks

From the experiments conducted above, we can observe that the overall performance of our deep learning classifiers have achieved better than the commonly used machine learning classifier. Even with the least improvement from ResNet model, there is still around a 40% increase in TSS. Such improvement might be due to the improvement on True Negative Rate where the deep learning models has better ability to specify the True Negatives correctly.

From the experiment of comparing original and synthetic modified data, we can observe that only applying jittering on the original data does not improve the performance for models that are strongly influenced by the noise. On the other hand, applying smoothing raises up the performance for these noise

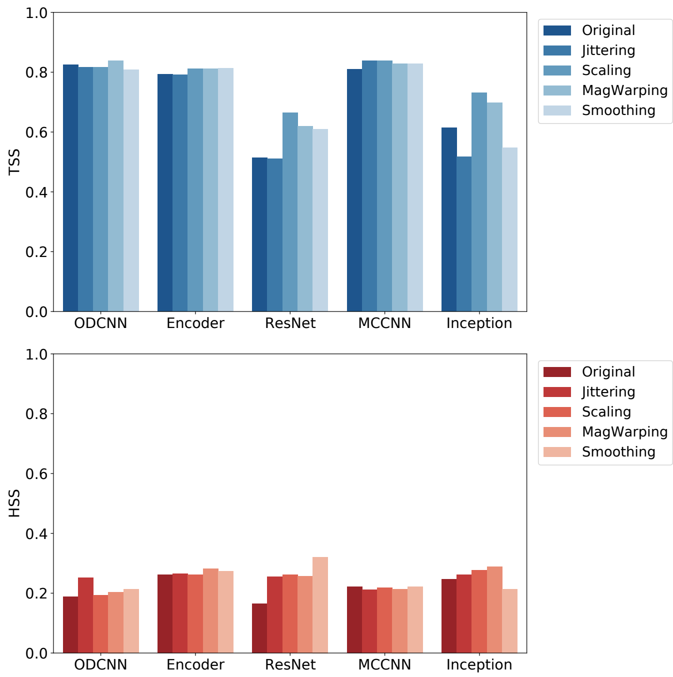

Fig. 3. TSS and HSS results for five selected deep learning classifiers on synthetic oversampling samples.

sensitive models. It can also be obeserved that by training these noise sensitive models (i.e., Encoder, ResNet, Inception) with synthetic oversampled dataset, jittering actually provides comparable results as the baseline models especially in HSS measurements.

VI. Conclusion

In this work, we have observed the performance of utilizing deep learning models onto multivariate time series data for solar flare forecasting. We also explored the feasibility of applying various data augmentation techniques on these models. Our results revealed that the model performance slightly improves when applying with augmentation. However, individual augmentation technique coupled with deep learning models showed fluctuations, making some baseline models perform better than their data augmented counterparts. There is still a high number of false positives exists in our evaluation phase, which highly impacts our results of TSS and HSS scores. This is a common problem in flare forecasting, which poses itself as a tradeoff between false positives and false negatives [20]. The possibility of combining different techniques to improve the performances also remains. Our work can be extended by adding more parameters to the classifiers or implementing different models that are more robust.

Acknowledgments

This project is supported in part under two NSF awards #2104004 and #1931555 jointly by the Office of Advanced Cyberinfrastructure within the Directorate for Computer and Information Science and Engineering, the Division of Astronomical Sciences within the Directorate for Mathematical and Physical Sciences, and the Solar Terrestrial Physics Program and the Division of Integrative and Collaborative Education and Research within the NSF Directorate for Geosciences.

References

[1] 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*, vol. 656, no. 2, pp. 1173–1186, Feb. 2007. [Online]. Available: https://doi.org/10.1086/510282

[2] H. Song, C. Tan, J. Jing, H. Wang, V. Yurchyshyn, and V. Abramenko, "Statistical assessment of photospheric magnetic features in imminent solar flare predictions," *Solar Physics*, vol. 254, no. 1, pp. 101–125, Nov. 2008. [Online]. Available: https://doi.org/10.1007/s11207-008-9288-3

[3] D. Yu, X. Huang, H. Wang, and Y. Cui, "Short-term solar flare prediction using a sequential supervised learning method," *Solar Physics*, vol. 255, no. 1, pp. 91–105, Feb. 2009. [Online]. Available: https://doi.org/10.1007/s11207-009-9318-9

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[6] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014.

[7] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," Jan. 2015, 3rd International Conference on Learning Representations, ICLR 2015 ; Conference date: 07-05-2015 Through 09-05-2015.

[8] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., vol. 26. Curran Associates, Inc., 2013.

[9] A. Ahmadzadeh, B. Aydin, D. J. Kempton, M. Hostetter, R. A. Angryk, M. K. Georgoulis, and S. S. Mahajan, "Rare-event time series prediction: A case study of solar flare forecasting," in *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. IEEE, Dec. 2019.

[10] R. A. Angryk, P. C. Martens, B. Aydin, D. Kempton, S. S. Mahajan, S. Basodi, A. Ahmadzadeh, X. Cai, S. F. Boubrahimi, S. M. Hamdi, M. A. Schuh, and M. K. Georgoulis, "Multivariate time series dataset for space weather data analytics," *Scientific Data*, vol. 7, no. 1, Jul. 2020. [Online]. Available: https://doi.org/10.1038/s41597-020-0548-x

[11] P. H. Scherrer, R. S. Bogart, R. I. Bush, J. T. Hoeksema, A. G. Kosovichev, J. Schou, W. Rosenberg, L. Springer, T. D. Tarbell, A. Title, C. J. Wolfson, and I. Z. and, "The solar oscillations investigation - michelson doppler imager," *Solar Physics*, vol. 162, no. 1-2, pp. 129–188, Dec. 1995. [Online]. Available: https://doi.org/10.1007/bf00733429

[12] O. W. Ahmed, R. Qahwaji, T. Colak, P. A. Higgins, P. T. Gallagher, and D. S. Bloomfield, "Solar flare prediction using advanced feature extraction, machine learning, and feature selection," *Solar Physics*, vol. 283, no. 1, pp. 157–175, Nov. 2011. [Online]. Available: https://doi.org/10.1007/s11207-011-9896-1

[13] T. A. M. H. Nagem, R. Qahwaji, S. Ipson, Z. Wang, and A. S. Al-Waisy, "Deep learning technology for predicting solar flares from (geostationary operational environmental satellite) data," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 1, 2018. [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2018.090168

[14] S. McGregor, D. B. Dhuri, A. Berea, and A. Muñoz-Jaramillo, "Flarenet: A deep learning framework for solar phenomena prediction," 2017.

[15] S. Dey and O. Fuentes, "Predicting solar x-ray flux using deep learning techniques," in *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1–7.

[16] N. Nishizuka, K. Sugiura, Y. Kubo, M. Den, and M. Ishii, "Deep flare net (DeFN) model for solar flare prediction," *The Astrophysical Journal*, vol. 858, no. 2, p. 113, May 2018. [Online]. Available: https://doi.org/10.3847/1538-4357/aab9a7

[17] T. Han, Q. Peng, Y. Shen, H. Li, and Y. Gu, "A deep learning model with multi-scale skip connections for solar flare prediction combined with prior information," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, Dec. 2019. [Online]. Available: https://doi.org/10.1109/bigdata47090.2019.9005508

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[19] 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*, vol. 856, no. 1, p. 7, Mar. 2018.

[20] A. Ahmadzadeh, B. Aydin, M. K. Georgoulis, D. J. Kempton, S. S. Mahajan, and R. A. Angryk, "How to train your flare prediction model: Revisiting robust sampling of rare events," *The Astrophysical Journal Supplement Series*, vol. 254, no. 2, p. 23, May 2021.

[21] A. Ahmadzadeh, M. Hostetter, B. Aydin, M. K. Georgoulis, D. J. Kempton, S. S. Mahajan, and R. Angryk, "Challenges with extreme class-imbalance and temporal coherence: A study on solar flare data," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, Dec. 2019.

[22] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, Jul. 2019. [Online]. Available: https://doi.org/10.1186/s40537-019-0197-0

[23] A. Krizhevsky, "Learning multiple layers of features from tiny images," pp. 32–33, 2009. [Online]. Available: https://www.cs.toronto.edu/ kriz/learning-features-2009-TR.pdf

[24] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2261–2269.

[25] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[27] M. G. Bobra and S. Couvidat, "Solar flare prediction using sdo/hmi vector magnetic field data with a machine-learning algorithm," *The Astrophysical Journal*, vol. 798, no. 2, p. 135, Jan. 2015. [Online]. Available: https://doi.org/10.1088/0004-637x/798/2/135

[28] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[29] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 07 2006. [Online]. Available: https://doi.org/10.1162/neco.2006.18.7.1527

[30] A. F. Agarap, "Deep learning using rectified linear units (relu)," *arXiv preprint arXiv:1803.08375*, 2018.

[31] J. Serrà, S. Pascual, and A. Karatzoglou, "Towards a universal neural network encoder for time series," *CoRR*, vol. abs/1805.03908, 2018. [Online]. Available: http://arxiv.org/abs/1805.03908

[32] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," *CoRR*, vol. abs/1607.08022, 2016. [Online]. Available: http://arxiv.org/abs/1607.08022

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, 2015.

[34] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 1578–1585, 2017.

[35] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 06, no. 02, pp. 107–116, Apr. 1998.

[36] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Time series classification using multi-channels deep convolutional neural networks," in *Web-Age Information Management*. Springer International Publishing, 2014, pp. 298–310. [Online]. Available: https://doi.org/10.1007/978-3-319-08010-9_33

[37] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *CoRR*, vol. abs/1512.00567, 2015. [Online]. Available: http://arxiv.org/abs/1512.00567

[38] B. K. Iwana and S. Uchida, "An empirical survey of data augmentation for time series classification with neural networks," *CoRR*, vol. abs/2007.15951, 2020. [Online]. Available: https://arxiv.org/abs/2007.15951

[39] C. M. Bishop, "Training with noise is equivalent to tikhonov regularization," *Neural Computation*, vol. 7, no. 1, pp. 108–116, Jan. 1995. [Online]. Available: https://doi.org/10.1162/neco.1995.7.1.108

[40] G. An, "The effects of adding noise during backpropagation training on a generalization performance," *Neural Computation*, vol. 8, no. 3, pp. 643–674, Apr. 1996. [Online]. Available: https://doi.org/10.1162/neco.1996.8.3.643