CS F469 Assignment: Fake News Detection on Social Media using Geometric Deep Learning

Anirudh S Chakravarthy *2017A7TS1195P*
Divyam Goel *2017A7TS1196P*
Pradhit Ongole *2017A3PS0188P*
Roshan Roy *2017A7TS1172P*
Yashaswi Pandey *2017B5A70971P*

***Abstract*—Fake news poses a serious threat to society by misleading content consumers with propaganda or outright lies. In this age of social media, misinformation can lead to confusion – people can no longer distinguish between authentic and fake content. In this report, we summarize the work proposed by Monti *et al.* [1] aimed to detect fake news using a Geometric Deep Learning based approach. Additionally, we also reimplement the authors' network architecture on 3 binary graph classification datasets – AIDS, PROTEINS, and SYNTHETIC datasets, achieving competitive performance on all benchmarks.**

***Index Terms*—Geometric Deep Learning, Fake News, Binary Classification, Graph Neural Networks**

I. Introduction

Over the past decade, the use of social media sites as a platform to disseminate news and information has seen a marked increase. For many, social media websites like Twitter or Facebook, are the main source of news about important events happening around the world. This rising influence of social media in the spread of information and formation of public opinion on important matters presents us with the very important and challenging task of tackling the spread of misinformation or 'fake news'.

Fake news articles have two characteristic features. First, they contain verifiably fake or untrue information, and second, they are created with malicious intent. The problem of fake news detection can then be formally stated as labelling a news article as true or false, based on some criteria.

Such articles are often used to spread false information and propaganda regarding political and social events, to manipulate public opinion. Recent examples of this can be seen from the 2016 US presidential elections and the Brexit vote in the UK. Given the ease of creating accounts and sending information over social media, the detection of fake news becomes a problem of utmost importance.

This report reviews the work on fake news detection by Monti *et al.* [1], who propose the use of Geometric Deep Learning to solve the problem of fake news detection. Our implementation is available at https://github.com/anirudh-chakravarthy/IR-Assignment.

II. Related Work

Most of the studies conducted on fake news detection methods can be classified into one of three categories – Content-based, Social Context-based, and Propagation-based.

A. Content-based

Content based approaches are mainly of two types – Knowledge-based and Style-based. Knowledge-based models aim to provide external fact checking of the content of the news article. Methods of fact checking can be employing experts to manually fact check the content (Expert Oriented), asking for a crowd of people to decide the truthfulness, and then aggregating the results (Crowd-sourcing Oriented), or using computational models to classify news based on open source information or knowledge trees [2] [3].

Style-based methods work on the assumption that malicious content writers need to have specific writing styles to influence a large audience. These methods try to capture such styles and use them to classify news as false or true [4] [5] [6].

B. Social Context-based

These methods are mostly used to supplement content-based approaches by using a user's social engagements as auxiliary information [4] [7] [8].

C. Propagation-based

The paper by Monti *et al.* proposes a third kind of method, based on the propagation of fake news on social networks. The authors model the social media spread of news articles in a manner similar to the spread of infectious diseases [9]. Both true and false news articles show characteristic behaviours in their spread patterns [10], and deep learning models could be used to exploit these patterns for fake news detection. There have been previous works in the direction of propagation methods, but these have been based on approaches using graph theoretical concepts of connected components, cliques etc [11].

III. Dataset Description

A. Data collection

The primary sources of data were websites such as Snopes, Politifact, and Buzzfeed which provide fact checking articles. Each of these articles has a claim (the main premise of the article), a label (true, false, partially true etc), and a list of related URLs. All articles which did not have labels of either true or false were not included by the authors in their study.

From the obtained articles, URLs were extracted and labelled. If the article had a label of true and the URL was supporting the claim, then it was also labelled as true. If the article was labelled as false and the URL supported the claim,

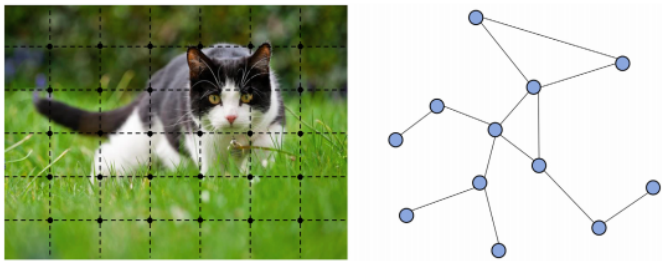


Fig. 1: Image in Euclidean space vs in a non-Euclidean space as demonstrated by Zhou *et al.* [12]

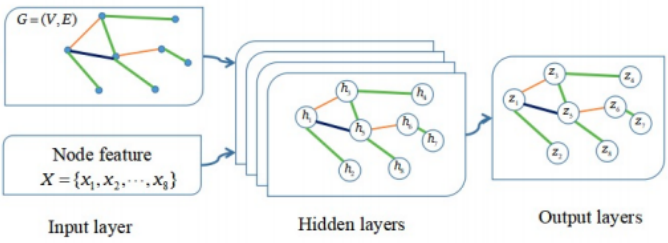


Fig. 2: Typical GNN structure as demonstrated by Guangfeng *et al.* [13]

then it was labelled as false. In case the URL was denying the claim, it was labelled as true if the article was labelled false, and false if the article was labelled as true. Also, any URL not shared at least once on Twitter was discarded. The classification of URLs into supporting or denying was done by trained humans. A cascade corresponding to a URL is defined as the chain of retweets that emerged from the original post. Each cascade was represented by a diffusion tree with the original post as its root [1] [10].

B. Input

These URLs and their cascades were formally represented by graphs G_u. We are given a URL u and a set of tweets $T_1, T_2, ..., T_n$ which contain the given URL. The vertices of the graph G_u are the tweets of the URL (along with user information). There is an edge between two vertices if and only if the author of the tweet i follows the author of tweet j, author of tweet j follows the one of i, if news spreads from j to i, or if news spreads from i to j.

News spreading is defined in the following manner – first, the tweets and their users are arranged in ascending order of tweet sending times. Then, if we are considering the user of tweet i and we've already considered users of tweets 1 to $i-1$, we say that news spreads from j $(1 \leq j \leq i-1)$ if one of the following holds – i follows j and there is no user $> j$ whom i also follows, or i follows no users in the list and j is the user with the highest number of followers [1].

C. Features

The following group of features were attached to the vertices - User Profile, User Activity, Network and Spreading, and Content. User profile activity includes features such as profile settings and self-description provided by the user. User Activity provides information about the various statuses and lists the user has. Social connections of the user, retweet timestamps and other information regarding the cascade spreading tree are captured in the Network and Spreading group. The Content feature is the actual content in the tweet. The feature attached to an edge is the relation type used to create it [1].

IV. Proposed Method

A. Graph Neural Networks

Deep learning techniques have typically demonstrated tremendous success in the domains of Computer Vision and Natural Language Processing. However most model architectures are dominated by CNNs that employ a fundamental assumption — the data exists in a Euclidean, grid-like space (Fig 1). To generalize deep learning to deviate from this assumption into non-Euclidean data (such as graph-structured), deep learning on graphs have come into popularity. Geometric Deep Learning is often coined as an overarching term for these non-Euclidean models.

In general, graph CNNs were introduced as a replacement for the well-known grid-based convolution operation, with aggregation around the vertices of a graph (Fig 2). This operation is achieved by Bruna *et al.* [14] towards spectral CNNs by representing filters as spectral parameters that can be trained. Later on, other work leveraging simple vector operations instead of decomposition of Laplacians was introduced to tackle the high computational complexity of spectral CNNs.

Graphs serve as a very abstract representation of the relationships and interactions in real-life processes. Thus, geometric deep learning techniques (often in the form of GNNs) have seen good success in wide-ranging problems such as Computer Graphics, Computer Vision, Life Sciences, and Pure Sciences.

B. Model Architecture

As demonstrated in Fig. 3, the proposed model receives data input in the form of extracted features from true and fake URLS, as described in III. With this work, we implement the model as introduced by Monti *et al.* [1]. The deep-learning based architecture is described in detail below.

The authors use a four-layered Graph Convolutional Neural Network (GCNN) that includes a two convolutional layers (both output features with resolution of 64), combined with two additional fully-connected layers (one with output feature resolution of 32, the other with feature resolution of 2) in order to achieve the target prediction of class: true vs false news. In each of the two convolution layers, one graph attention head, as demonstrated in [15] was used to create the filters. Additionally, to achieve dimensionality reduction from data in high dimensions, mean pooling was employed. The authors justify the choice to use Scaled Exponential Linear Units (SELUs) as the non-linearity in the entire model architecture, due to the fact that SELUs cannot reach a dead activation state. After experimentation with different types of loss functions (Hinge loss, cross-entropy loss, tangent loss, and exponential loss),

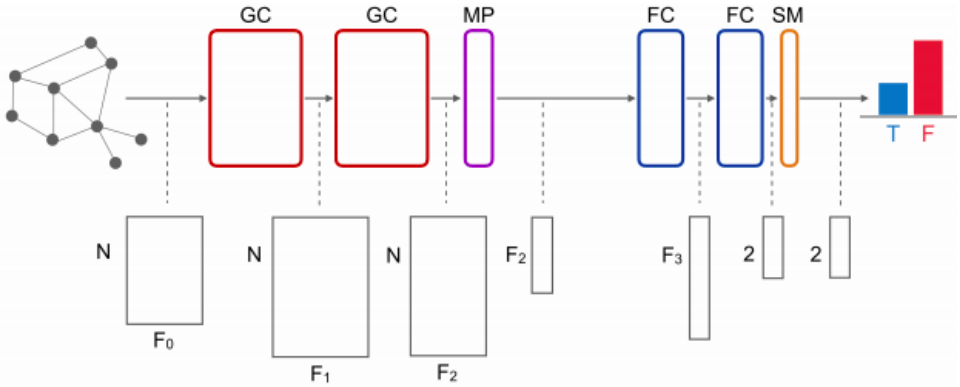


Fig. 3: Model Architecture as proposed by Monti *et al* [1]. GC = Graph Convolution layer, MP = Global Mean Pooling Layer, FC = Fully Connected Layer, SM = SoftMax layer.

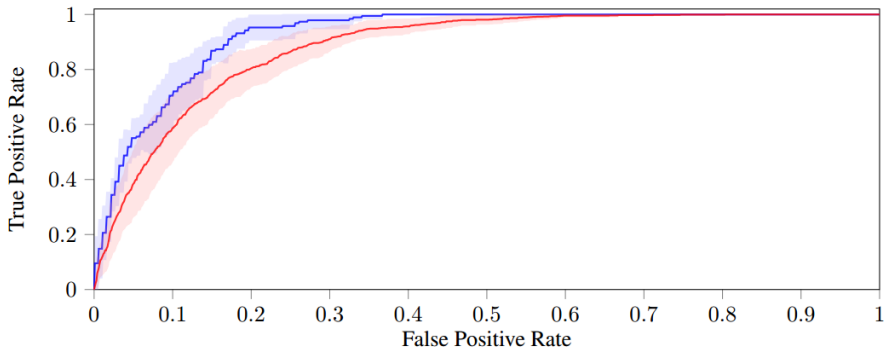


Fig. 4: ROC curves for URL-wise (blue line) and cascade-wise (red line) classification for a 24 hour diffusion time. Shaded area represents standard deviation in the performance.

hinge loss was proven to outperform the others. Regularization techniques were not employed in this formulation.

V. Reported Results

Fake news detection was done in two ways: *URL-wise* and *cascade-wise*. In *URL-wise* detection, a URL containing a news story was labelled as true or false based on all the Twitter cascades it generates. In *cascade-wise* detection, the truth or falsity of a URL was determined from only one cascade arising from it. All other cascades arising from that URL inherit this same label. For both of these methods, the same five randomized training/validation/test splits are used. Training was done using AMSGrad as the optimizer with a learning rate of $5e-4$ and a mini batch size of one in both cases.

A. URL-wise detection

The average number of URLs in the training, validation, and test sets were 677, 226, and 226 URLs respectively. 83.26% of the URLs were labelled as true and 16.74% as false. After training the neural network on 25,000 iterations, the performance of the model was studied by plotting the ROC curve between false positive (fraction of true news predicted as false) and true positive(fraction of true news predicted as true). As the true positive rate increases, the rate of false positive also increase, so there exists a trade-off between the two. The aggregate measure of $92.70 \pm 1.80\%$ was obtained by calculating the area under the ROC curve (ROC AUC) shown in Fig 4.

1) Time-based analysis: As the URL-wise method is a propagation-based fake news detection method, it is also important to take into consideration the amount of time for which the news is allowed to spread. For achieving this, the cascades arising from the URLs were truncated after a time t, where t represents the cascade duration, which varies from 0 to 24 hours in 1 hour intervals. The model was trained with different values of t to analyze the effect of time on the performance of the model. To reduce the bias and hence the error, five-fold cross validation was used. The performance of the model increased as the value of t increases and saturated after a cascade duration of 15 hours as depicted in Fig 5. After 15 hours the model was able to encompass 86% of the cascade size.

The mean ROC AUC metric, however, takes only around 2 hours to increase to 90%. As this metric increases sharply from 0 to ≥ 1 hour, it also indicates the importance of time in the performance of the model.

2) Model age-based analysis: The performance of the model was analyzed for a situation in which a trained model

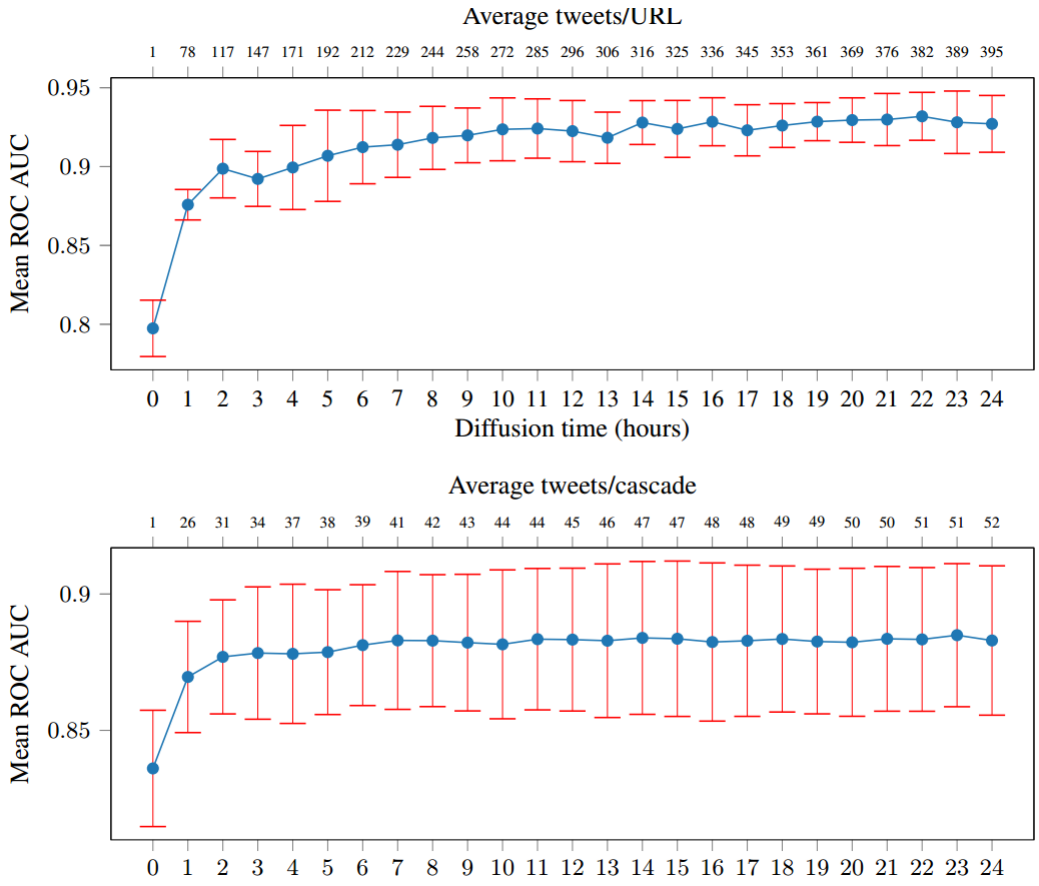


Fig. 5: Mean ROC AUC as a function of cascade diffusion time for URL-wise (top) and cascade-wise detection (bottom).

has to classify tweets several days or months after the model was trained. For this purpose, the dataset was divided into two parts, the training/validation set consisting of 80% of the URLs and test set consisting of 20% of the URLs. Testing was done on partially overlapping subsets of the test set based on their time windows, ensuring that the average date of consecutive subsets was at least 14 days apart. This ensured that there were more subsets to test upon and each of the subsets had at least 24% of the URLs in the test set. The results of this analysis are shown in Fig 6.

3) Feature based analysis: The features provided to the model along with the inputs: user profile, user activity, network and spreading, and content, have varying impacts on the performance of the model. To analyze the importance of each feature, an ablation study was conducted using backward feature selection. User profile and network and spreading allow the model to achieve ROC AUC of 90% and so are the most important feature groups. The performance of the model increases only slightly after the introduction of content and user activity.

B. Cascade-wise detection

The average number of cascades in the training, validation, and test sets were 3586, 1195, and 1195 cascades respectively. 81.73% of these were labelled as true and 18.27% were labelled as false. Upon training for 50,000 iterations, the performance of the model was studied similar to *URL-wise detection* by calculating the area under ROC curve shown in Fig 4. The mean ROC AUC was calculated to be 88.30% $\pm$ 2.74%.

1) Cascade size-based analysis: The size of the cascade in this method has an important role to play in the performance of the model. Smaller cascades do not indicate any clear diffusion pattern, and so they must be neglected. Cascades of various minimum sizes were used to study the performance of the model. The performance saturated after a minimum cascade size of 6 tweets. After removing cascades with less than 6 tweets, the data set had a total of 5976 cascades on which the model performance was calculated.

2) Time-based analysis: A time-based analysis similar to that done for *URL-wise detection* was performed. The analysis depicted that the performance of the model saturates after a cascade duration of only 7 hours as seen in Fig 5. In the *URL-wise detection* saturation is achieved at a much higher value of cascade duration (15 hours). This difference can be attributed to shorter cascade life and simpler topological patterns in case of *cascade-wise detection*.

3) Model age-based analysis: Similar to the *URL-wise detection*, analysis was done to interpret the performance of the model when testing is done on tweets published several days

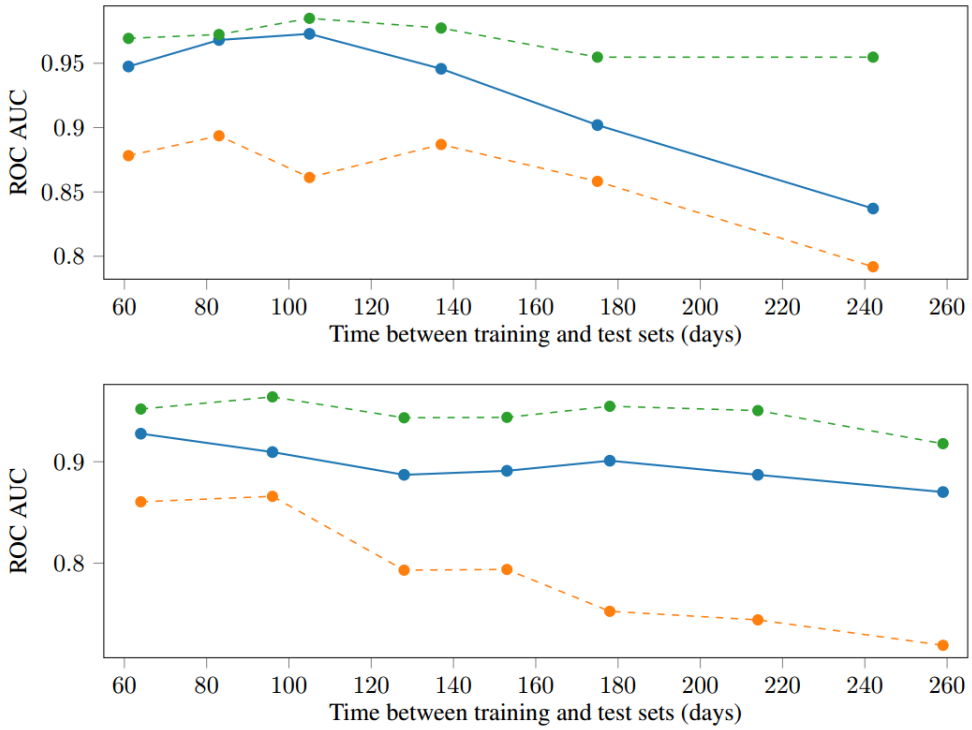


Fig. 6: A plot of URL-wise (top) and cascade-wise (bottom) performance as a function of training set age. The performance obtained for 24 hours diffusion (blue line), performance obtained for 0 hours diffusion (using only the first tweet of each piece of news, orange line) and performance obtained for 24 hour diffusion along with five-fold cross validation (green line) is shown.

or months after the model was trained. Partially overlapping time windows were also created similarly by dividing the test set into subsets. Each of these subsets had 314 $\pm$ 148 cascades on average. The *cascade-wise detection* showed a more robust behaviour in this analysis. The mean ROC AUC decreased by only 4% when classifying tweets published 260 days after training as can be seen in Fig 6. This difference in behaviour can be attributed to two main reasons. First, URLs are represented by more complex graphs than individual cascades. The neural network tends to identify rich structures in the dataset while training. Second, cascades may encompass a variety of user profiles coming from different areas of Twitter.

4) Feature-based analysis: A similar ablation study was done in this detection method as in *URL-wise detection*. A similar rise in performance was seen when user profile and network and spreading were introduced in the input. However, the model's performance showed a 4% decrease after the content was introduced as an input feature. Such contradictory behaviour could be observed mainly due to the model overfitting on the content feature. A large number of cascades (20%) are associated with the 15 largest URLs, which make up only 1.5% of all URLs in the training set. These tweets may represent the same content and hence lead to overfitting.

VI. Our Implementation

In this section, we describe our re-implementation of the approach by Monti *et al*. To evaluate our implementation, we use datasets from the TUDataset Benchmark Datasets [20]. In particular, we select 3 datasets: AIDS [16], PROTEINS [17], [18], and SYNTHETIC [19] for evaluating the classification performance.

A. Dataset description

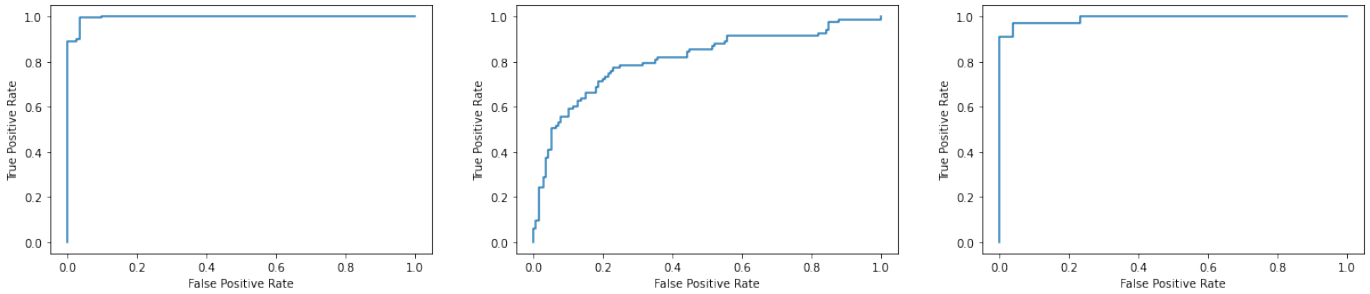
Since the fake news classification dataset used in [1] is not publicly available, we perform classification on other standard datasets. We use 3 binary graph classification datasets from the TUDataset Benchmark Dataset suite. We specifically choose these datasets as they involve binary classification and contain node attributes, similar to the original paper. The AIDS dataset consists of 2000 graphs, with an objective to classify molecules as AIDS or non-AIDS. The PROTEINS dataset consists of 1113 graphs to classify the protein graphs as enzymes or non-enzymes. The SYNTHETIC dataset consists of 300 synthetically generated continuous attributed graphs with an aim of binary classification.

B. Implementation Details

We use PyTorch for our implementation. We train our model for 500 epochs on the AIDS dataset, and for 1000 epochs

Dataset	Accuracy	Precision	Recall	F1 Score
AIDS [16]	99.02%	99.25%	99.52%	98.51%
PROTEINS [17], [18]	77.89%	68.91%	71.57%	76.21%
SYNTHETIC [19]	95%	96.97%	94.12%	94.93%

TABLE I: Results on various benchmarks on the test set.



(a) AIDS (AUC = 99.57%) (b) PROTEINS (AUC = 81.1%) (c) SYNTHETIC (AUC = 99.1%)

Fig. 7: ROC curves computed across various benchmarks.

on PROTEINS and SYNTHETIC datasets. We use a batch size of 128. We use the Hinge loss for training our network, with Adam optimizer and a learning rate of $1e-3$. Since the benchmarks do not specifically consist of test data, we sample the training and test set by performing an 80-20 split on the dataset.

We use accuracy, precision, recall, and F1 score (with macro averaging) as the evaluation metrics. These metrics are computed as follows:

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

$$P = \frac{TP}{TP + FP} \tag{2}$$

$$R = \frac{TP}{TP + FN} \tag{3}$$

$$F1 = \frac{2 * P * R}{P + R} \tag{4}$$

where TP, TN, FP, and FN denote True Positives, True Negatives, False Positives, and False Negatives respectively. Acc denotes Accuracy, P denotes Precision, R denotes the Recall, and $F1$ denotes the F1-score. We report these evaluation metrics on the test set.

C. Results

As seen in Table I, we obtain competitive performance on all 3 benchmarks. Specifically, on the AIDS and SYNTHETIC dataset, we report extremely strong performance on all evaluation metrics.

In order to further examine the classification performance, we plot the ROC curve for each benchmark in Fig 7 and compute the area under each curve (ROC AUC). For the AIDS dataset, we achieve an ROC AUC of 99.57%, for the PROTEINS dataset, we achieve an ROC AUC of 81.1%, and for the SYNTHETIC dataset, we achieve an ROC AUC of 99.1%.

We observe good performance on evaluation metrics and a very high ROC AUC, which shows the robustness of our implementation on the classification benchmark and the strength of the Geometric Deep Learning approach.

VII. Future work

Several observations have been made that leave scope for future research. We have validated the efficacy of the authors' model across competitive benchmarks, but not across languages and geographies. If proven so, then it would validate the theory that the architecture is mainly dependent on graph-based connections and the spacing of features.

Adversarial attacks are another interesting avenue of research. They could potentially be used to understand the shortcomings of model architectures and test the stability of attacks. In addition, it is possible that adversarial attacks can reduce the black-box nature of decision-making in graph-based networks. This leads to an increase in the overall interpretability of the model and increased application in sensitive industries such as the medical domain.

Finally, this approach could make significant headway in social media network analysis for fake news detection, which in today's age of information, is a critical task. In cases such as the Sushant Singh Rajput death case and Donald Trump's Speech Analysis, these systems could improve transparency in the public domain.

VIII. Conclusion

In this report, we summarize the approach followed in the paper by Monti *el al.*, titled 'Fake News Detection on Social Media using Geometric Deep Learning'. We also reimplement the authors' model on various graph classification benchmarks, achieving competitive results across various evaluation metrics. Finally, we also present several future directions which can be explored.

REFERENCES

[1] F. Monti, F. Frasca, D. Eynard, D. Mannion, and M. M. Bronstein, "Fake news detection on social media using geometric deep learning," *arXiv preprint arXiv:1902.06673*, 2019.

[2] O. Etzioni, M. Banko, S. Soderland, and D. S. Weld, "Open information extraction from the web," *Communications of the ACM*, vol. 51, no. 12, pp. 68–74, 2008.

[3] A. Magdy and N. Wanas, "Web-based statistical fact checking of textual documents," in *Proceedings of the 2nd international workshop on Search and mining user-generated contents*, 2010, pp. 103–110.

[4] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu, "Fake news detection on social media: A data mining perspective," *ACM SIGKDD explorations newsletter*, vol. 19, no. 1, pp. 22–36, 2017.

[5] S. Feng, R. Banerjee, and Y. Choi, "Syntactic stylometry for deception detection," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2012, pp. 171–175.

[6] V. L. Rubin and T. Lukoianova, "Truth and deception at the rhetorical structure level," *Journal of the Association for Information Science and Technology*, vol. 66, no. 5, pp. 905–917, 2015.

[7] S. M. Mohammad, P. Sobhani, and S. Kiritchenko, "Stance and sentiment in tweets," *ACM Transactions on Internet Technology (TOIT)*, vol. 17, no. 3, pp. 1–23, 2017.

[8] Z. Jin, J. Cao, Y. Zhang, and J. Luo, "News verification by exploiting conflicting social viewpoints in microblogs," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.

[9] A. Kucharski, "Study epidemiology of fake news," *Nature*, vol. 540, no. 7634, pp. 525–525, 2016.

[10] S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," *Science*, vol. 359, no. 6380, pp. 1146–1151, 2018.

[11] S. Kwon, M. Cha, K. Jung, W. Chen, and Y. Wang, "Prominent features of rumor propagation in online social media," in *2013 IEEE 13th international conference on data mining*. IEEE, 2013, pp. 1103–1108.

[12] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, and M. Sun, "Graph neural networks: A review of methods and applications," *ArXiv*, vol. abs/1812.08434, 2018.

[13] L. Guangfeng, J. Wang, K. Liao, Z. Fan, and W. Chen, "Structure fusion based on graph convolutional networks for node classification in citation networks," *Electronics*, vol. 9, p. 432, 03 2020.

[14] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.

[15] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio', and Y. Bengio, "Graph attention networks," *ArXiv*, vol. abs/1710.10903, 2018.

[16] K. Riesen and H. Bunke, "Iam graph database repository for graph based pattern recognition and machine learning," in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Springer, 2008, pp. 287–297.

[17] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. Vishwanathan, A. J. Smola, and H.-P. Kriegel, "Protein function prediction via graph kernels," *Bioinformatics*, vol. 21, no. suppl_1, pp. i47–i56, 2005.

[18] P. D. Dobson and A. J. Doig, "Distinguishing enzyme structures from non-enzymes without alignments," *Journal of molecular biology*, vol. 330, no. 4, pp. 771–783, 2003.

[19] A. Feragen, N. Kasenburg, J. Petersen, M. de Bruijne, and K. M. Borgwardt, "Scalable kernels for graphs with continuous attributes." in *NIPS*, 2013, pp. 216–224.

[20] C. Morris, N. M. Kriege, F. Bause, K. Kersting, P. Mutzel, and M. Neumann, "Tudataset: A collection of benchmark datasets for learning with graphs," in *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020)*, 2020. [Online]. Available: www.graphlearning.io