KTH ROYAL INSTITUTE OF TECHNOLOGY

# Project Proposal

***Student :***
Céline Do Thuan
candt@kth.se

***Teacher :***
Jim Dowling
jim@hopsworks.ai

December 2024
ID2223 - Scalable Machine Learning and Deep Learning

## 1 Project Idea

The project is based on the paper *Fake News Detection on Twitter Using Propagation Structures* from Marion Meyers(B), Gerhard Weiss, and Gerasimos Spanakis. [1]

The idea is to build a Fake News Detector from Tweets using graphs. Indeed, on Twitter, real and fake news propagate differently. They go through different circles and paths, and with different speeds. For example, fake news are 70% more to be retweeted than real news are, and they take about six times less long to reach 1,500 people than real ones. [2]

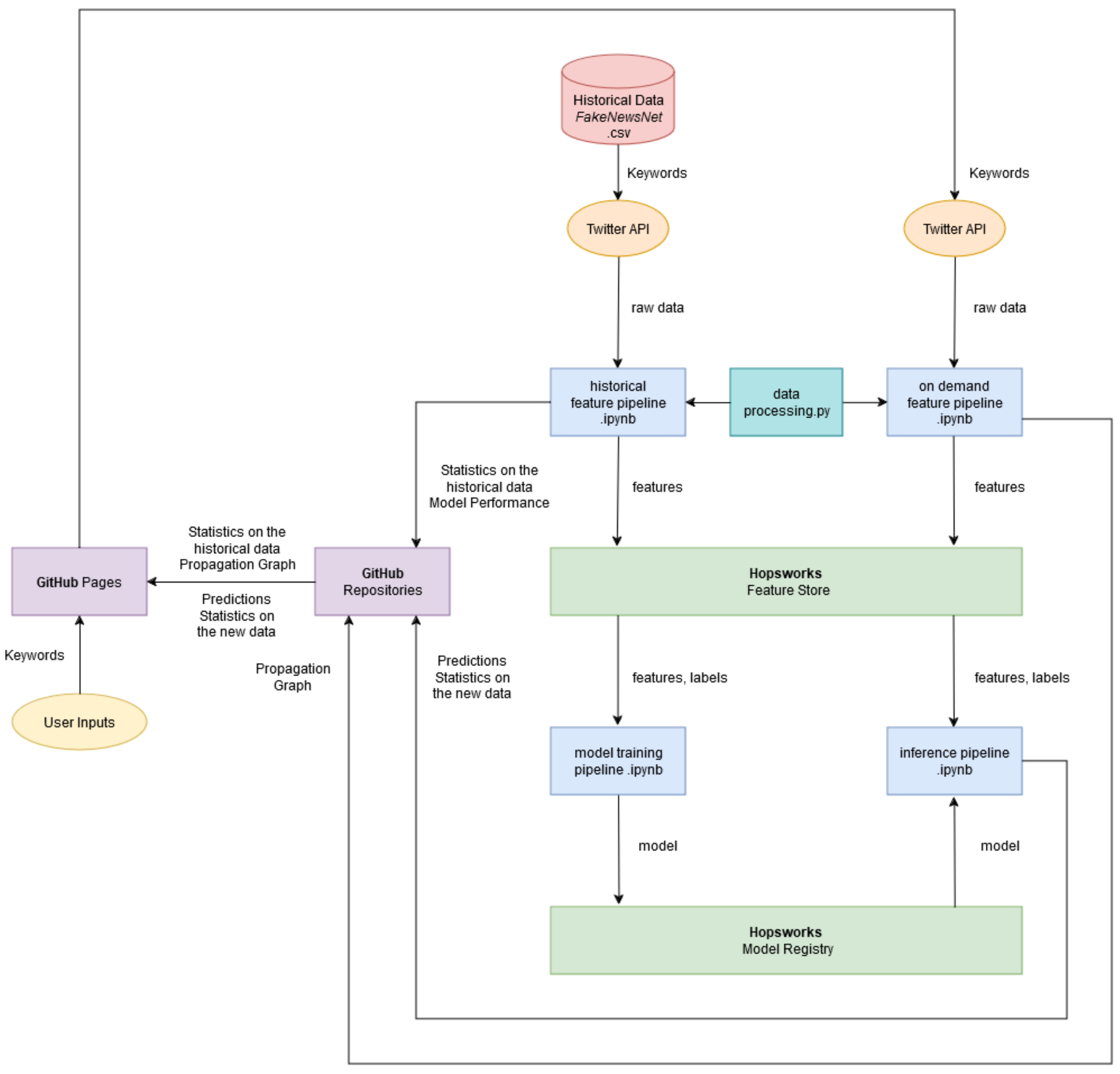
Historical Data
FakeNewsNet
.csv
Keywords
Twitter API
raw data
historical feature pipeline .ipynb
data processing.py
Keywords
Twitter API
raw data
on demand feature pipeline .ipynb
Statistics on the historical data Model Performance
features
features
GitHub Pages
Statistics on the historical data Propagation Graph
Predictions Statistics on the new data
GitHub Repositories
Hopsworks Feature Store
Keywords
User Inputs
Propagation Graph
Predictions Statistics on the new data
features, labels
features, labels
model training pipeline .ipynb
inference pipeline .ipynb
model
model
Hopsworks Model Registry

Figure 1: Project structure

## 2 Method

First, we will use the Twitter API to build our static, historical dataset. We can use the following tool to do that : *https://github.com/KaiDMML/FakeNewsNet*. This allows us to query specific news that are classified as real or fake, and get additional information about the tweet, like the retweets or the user profile.
Then, we will build the propagation graph, in the way described in the paper.

Let $G = (V, E)$ be the news graph.

- $V$ is the set of nodes of the graph. A node can be of two types:
    1. A tweet node: the node stores the tweet and its associated user. A tweet belongs to a news graph if it contains the keywords extracted from the headline of the news article.
    2. A retweet node: the node stores the retweet and its associated user. All retweets of a tweet node are present in the graph.
- $E$ is the set of edges of the graph. Edges are drawn between a tweet and its retweets. Edges contain a time weight that corresponds to the time difference between the tweet and retweet publish times.

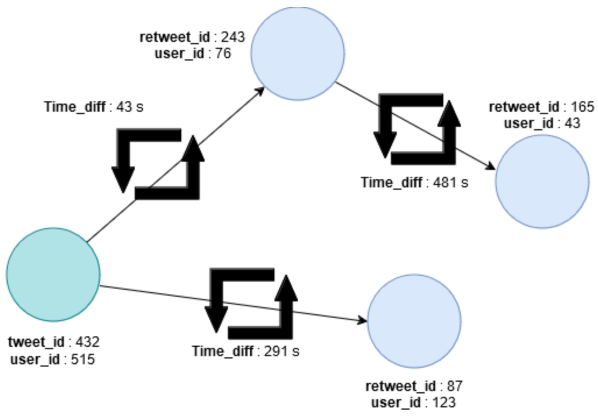

Figure 2: Example of a propagation graph

We will use the propagation graph to build the classifier. The paper uses two different ways to build the classification model.

The first way is to manually extract features from the propagation graph, like the average time between a tweet and a corresponding retweet, along with the previous features queried from the Twitter API. Then, they fed those features to different classifiers to test their performances over the full and balanced dataset.
We will use the Random Forest Classifier as it yielded the best result in the paper (86% of accuracy), and will allow us to extract the importance of the different features in the classification.

The second way is to use geometric graph learning, not to classify nodes in the graph, but to classify the graph itself. As this method is harder to implement and yielded worse result than with the traditional classifier (73% of accuracy), we will only implement it if time allows us to.

Finally, our end-to-end Machine Learning system will be on-demand for users, through the UI. They will be able to fact-check pieces of news that they will input into the UI.

## 3 Dataset

We will use 2 types of datasets :

- a static, historical dataset : the FakeNewsNet dataset.
- a non-static data stream from the Twitter API, to get the news that we want to classify.

As mentioned before, the static dataset comes from the FakeNewsNet dataset, which is a news articles dataset that have been fact-checked and labeled by two organizations : Politifact and Gos sipcop. The headline from each news article is then extracted and used as keywords for a query to the Twitter API.

The on-demand dataset will also come from queries to the Twitter API, but the keywords will be selected by the user, through the UI.

The Twitter API queries allow us to get the following features :

- news content: the body of the article, images, publish date.
- tweets: the list of tweets containing the article headline keywords
- retweets: the list of retweets of all tweets previously retrieved
- user information: the profile information (user id, creation date, 200 most recent published tweets, list of followers and friends) of all users that have posted a tweet or retweet related to the news article.

Both datasets have the same features, manually extracted from the graphs. Here is the list of features in the dataset :

- User/Social Context Features
  - *Average number of followers* : For each user that has either posted a tweet or a retweet in the graph, his amount of followers is retrieved. Those counts are then averaged over all users involved in the news graph
  - *Average number of following* : For each user that has either posted a tweet or a retweet in the graph, his amount of following (friends) is retrieved. Those counts are then averaged over all users involved in the news graph

- Network Features
  - *Retweet Percentage* : This is measured through the following equation:

$$\frac{\text{number of retweets}}{\text{number of tweets} + \text{number of retweets}}$$

  - *Average Time Difference* : This measures the average time between a tweet and a corresponding retweet. Since each edge of the graph has a time weight on it, it is computed by making the average of all the edge weights of the graph
  - *Number of tweets*
  - *Number of retweets*
  - *Time first last or News lifetime* : This measure is obtained by computing the time difference between the first and last recorded publish dates of tweets (or retweets) in the graph
  - *Average favorite count* : For each node, its number of favourites is retrieved. Those counts are then averaged over all nodes in the graph
  - *Average retweet Count* : For each tweet, its number of retweets is retrieved. Those counts are then averaged over all tweets in the graph
  - *Users Touched in 10h* : Starting from the first post recorded in the graph, all posts that happened in the first 10 hours of the diffusion are retrieved. From those posts, the amount of unique users involved in the spread is then calculated
  - *Performance of the post in 1 hour* : This feature is calculated by the following equation:

$$\frac{\text{number of tweets and retweets in the first hour}}{\text{total number of tweets and retweets in the graph}}$$

## 4 Tools

The data and the model will be stored on Hopsworks.
The UI will be hosted by GitHub Pages.
The on-demand pipeline will be handled by GitHub Actions.

## 5 UI Design

The user will have an input box, where they will be able to input keywords that describe the news article they want to fact check. The website will return the label "Fake" or "Real" to the user.

The UI could also display the graph from the queried tweet and the statistics (features) that we extracted from that tweet, compared to the average statistics for fake and real news on Twitter.

## References

[1] M. Meyers, G. Weiss, and G. Spanakis. *Fake News Detection on Twitter Using Propagation Structures*, pages 138–158. 10 2020. ISBN 978-3-030-61840-7. doi: 10.1007/978-3-030-61841-4_10.

[2] S. Vosoughi, D. Roy, and S. Aral. The spread of true and false news online. *Science*, 359 (6380):1146–1151, 2018. doi: 10.1126/science.aap9559. URL `https://www.science.org/doi/abs/10.1126/science.aap9559`.