Active Learning for social media data classification

Progetto di Ingegneria Informatica

Tommaso Crippa

15 June 2024

Personal Code: 10776593
Professor: Barbara Pernici
Supervisor: Carlo Alberto Bono

Contents

1 Introduction

Social media has become a fundamental part of everyday life, serving as a primary means of communication and information sharing worldwide. Platforms like Twitter, Facebook, and Instagram are not only used for personal interactions but also play a crucial role in documenting ongoing events. In the 21st century, social media has revolutionized how we connect, share information, and stay informed, becoming an essential tool for any personal and professional task.

During emergencies, social media platforms provide real-time data about the latest updates, offering a fast open source of immediate news. This capability makes social media an indispensable tool for both individuals and organizations in managing and responding to crises. Specifically, in the context of disasters, these platforms enable fast transmission of critical information, helping to coordinate response efforts and inform the public about safety measures and developments.

However, the vast amount of content posted during such events is often redundant, as it shares the same information from other sources, or is irrelevant, discussing completely different topics. These are some of the main problems that make it challenging to rapidly extract the most critical updates. The numerous amount of posts can overwhelm people from trying to stay informed and respond effectively. Filtering through this data manually is impractical and time-consuming, therefore a new sophisticated method to conduct the process is needed.

Active Learning can become an excellent tool in this context. Active Learning is a subset of Machine Learning, specialized in filtering through massive data streams and prioritizing the most important and relevant posts. By actively selecting the most informative data points for training, these algorithms enhance the model's learning efficiency and effectiveness. Implementing Active Learning in disaster response systems can significantly improve the timeliness and accuracy of information propagation, ensuring that key information is found quickly and accurately and enabling better-coordinated response efforts and more informed decision-making during crises.

1.1 Objective

The objective of this project is to support research in the field of Active Learning by conducting a detailed study on the effectiveness of some of its techniques for collecting and classifying information about an ongoing event. This project aims to examine how these techniques can be used to improve the data collection process during a dynamic and rapidly changing situation, to achieve an accurate and useful information system.

In particular, the "Coreset" technique will be tested, a method that selects a subset of the data that best represents the entire stream; in particular, two specific "Coreset" algorithms will be analyzed, "Farthest-First" and "Minimax", comparing and finding out their strengths and weaknesses. the dataset used will be a comprehensive batch of tweets sent on Twitter during the conflict between Ukraine and Russia at the beginning of 2022.

By applying these techniques to such a complex geopolitical event, the study aims to demonstrate the capability of Active Learning to significantly improve the speed and accuracy of information systems during disasters and crises, exploring the effectiveness of the "Coreset" technique in managing and responding to dynamic situations. Ultimately, the study seeks to contribute to the development of more efficient tools for

crisis management, ensuring that critical information is accessible and usable when it is most needed.

1.2 Related work

The use of social media posts as a source of information to detect and gather details about events has been tested extensively utilizing numerous amounts of different architectures. These range from keyword detection infrastructures employing particle filtering techniques [6] to topic-based modeling approaches [5]. The field has developed a significant number of techniques for effective data retrieval from social media platforms, and the utilization of Active Learning techniques is one of them.

In the field of Machine Learning, Active Learning, while not being the most predominant focus, has been thoroughly studied. Researchers have developed a comprehensive set of techniques aimed at optimizing the learning process from limited labeled data [3]. Specifically, the "Coreset" technique has shown to be a great algorithm for its proven efficiency in handling large-sized datasets [7]. This method strategically selects a representative subset of data to train models more effectively, reducing computational costs while maintaining high accuracy; this is also why one of the main applications of these approaches has been inside the social media data gathering process.

In fact, the combination of Active Learning techniques applied to social media data has been explored in various studies, particularly in the context of natural disasters. For instance, researchers have investigated the application of these techniques to situations such as bushfires and floods, yielding successful results in improving disaster response [1] [4]. However, no study has yet examined the use of these tools in the context of the Russo-Ukrainian conflict. This project aims to demonstrate that the efficiency of these algorithms for data retrieval is effective not only in the context of crisis naturally-caused crises such as hurricanes and earthquakes but also in geopolitical events created by humans, like terrorist attacks or wars such as this one.

2 Implementation

A typical Active Learning pipeline has the structure shown in Figure 1:

Given a stream of posts coming from social media, each one of them passes through an embedding process, where its content is mapped on a high-dimensional plane. The group of embeddings is then processed by an Active Learning algorithm, the main element of the pipeline, where only a small subset of posts (previously chosen by the *budget* parameter) are selected to be validated by a human, which labels each post as "Relevant" or "Not Relevant" to the current event.
The (post, label) tuple is then saved on a database, where its data is used to train a neural network, which is finally used to classify the relevancy of posts.

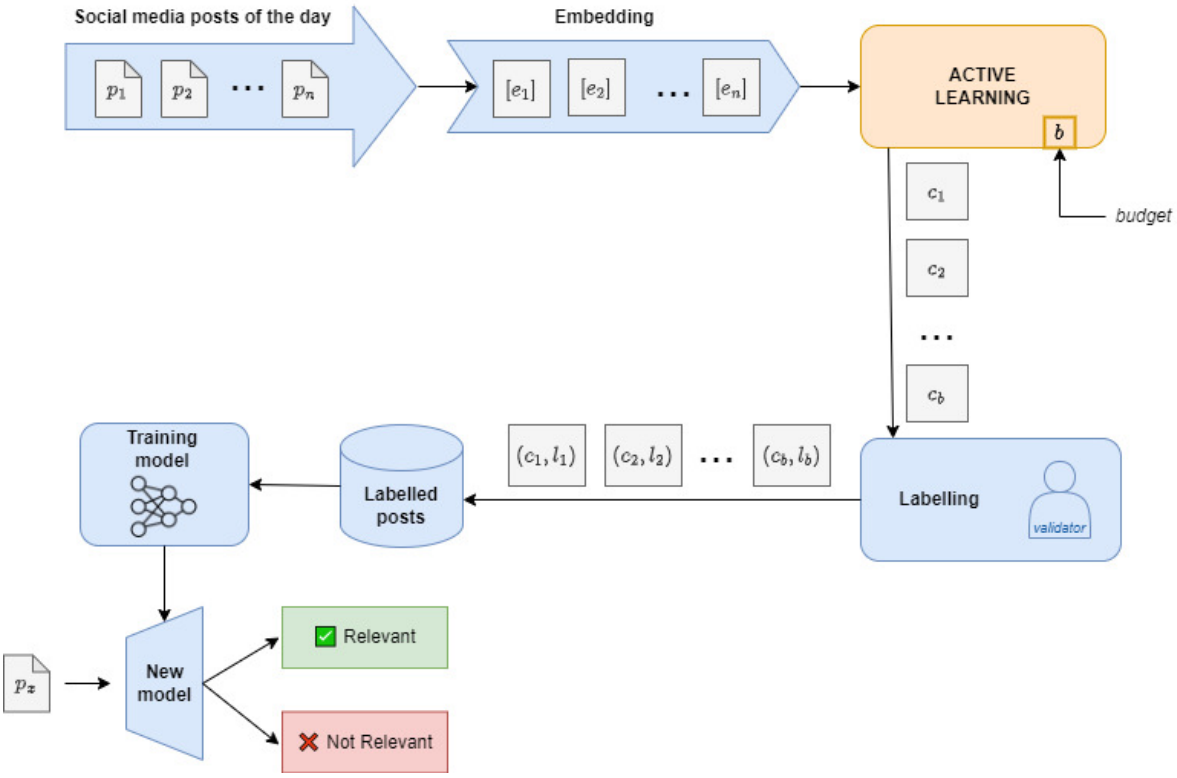


Figure 1: Social Media Active Learning Pipeline

In this project, the main focus will be on the Active Learning part, which is why some other processes will be simplified to test the effectiveness of the algorithm rapidly.
For example, the labeling activity is not done manually by a human operator but it is automatized by using an LLM: the label is not a boolean anymore, but a value between 0 and 1, corresponding to the probability that the tweet content is relevant to the ongoing event; the LLM is asked in 5 different instances to respond "1" if the tweet content contains relevant information about the event, and "0" otherwise; the final value is the average obtained.
Furthermore, the selected embedding model, all-MiniLM-L6-v2, was chosen as a fast but less accurate option, to save up space on the RAM during the computational part of the project.

2.1 Dataset Study

Before approaching the coding phase, I first performed an analysis of the dataset with which I would be working. The dataset consists of 325,000 tweets, each with the following attributes saved:

tid	cid	text	retweet_count	like_count	text_clean	ts	casualty	embedding
...	...	...	...	...	...	...	...	...

- **tid** & **cid** (int): unique identifiers of the tweet

- **retweet_count** & **like_count** (int): tweet impressions
- **text** & **text_clean** (str): textual content of the tweet, the second one is cleaned of all the problematic characters that might not be recognized during the embedding process
- **ts** (datetime): date and time of the tweet's publication
- **casualty** (float): value between 0 and 1, corresponding to the probability that the tweet content talks about war casualties
- **embedding** (tuple<float>): vector composed of 384 floats that represents the tweet's content, used as input for the neural network

The tweets included in the dataset were published during the first two weeks of the conflict between Ukraine and Russia, from 22/02/2022 to 07/03/2022, approximately 23,000 posts per day.
Most of the tweets (70% of them) do not mention casualties, while only about 9% have a high probability of discussing this topic.

Below is an example of a tweet that has been classified as discussing casualties:

> "*@SenStabenow Ukraine needs weapons and humanitarian assistance to defend against #Putin. Stop innocent civilian deaths. @POTUS, provide #SafeAirliftUkraine #StopPutin*"

2.2 Model Construction

To train the classification of tweet labels, a sequential neural network built in TensorFlow was used.

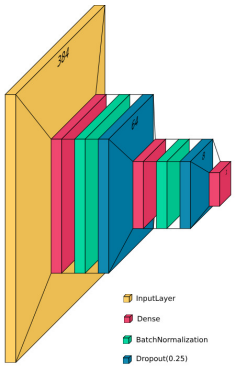


Figure 2: Structure of the neural network

The network takes the tweet's embedding as input, a vector composed of 384 floats, and passes through two dense layers composed of 64 and 8 neurons respectively, with additional normalization and dropout steps to stabilize learning. The final output is a value between 0 and 1, activated via a sigmoid function. The model is then compiled with a loss function specific to binary classification, using an Adam optimizer with a learning rate of 0.001.

To compensate for the high number of non-relevant tweets with respect to the relevant ones observed during the study of the dataset, each class has an associated weight to balance their effect on the neural network by giving more importance to the relevant tweets during the training phase.

During this stage, the labeled elements are divided into a training set and a validation set with an 80/20 split. The model is then trained for 15 epochs, and the best model is selected based on its highest accuracy on the validation set. This model is then tested on the data points from the next timeframe.

The model's structure was designed to achieve significant results despite a limited amount of data, using a simple configuration and elements to prevent overfitting.

2.3 Active Learning Methods

Active learning is a field of machine learning that focuses on optimizing the learning process of neural networks by selectively labeling only the most informative data. Instead of forcing oracles, human annotators that classify the contents of data points, to operate on large amounts of data, an active learning system dynamically chooses which data it wants the oracle to label and learn from that it predicted to be the most beneficial for improving its performance. This approach is particularly valuable when labeled data is scarce or expensive to obtain, as it aims to achieve better model accuracy with fewer labeled examples.

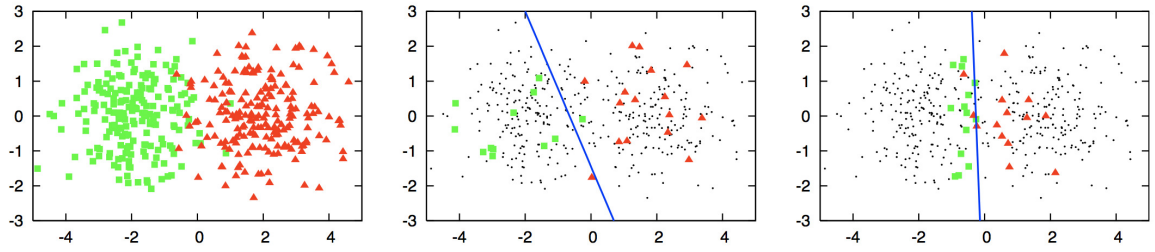


Figure 3: Example of Active Learning Utilization [2]

Figure 3 for example, shows the effectiveness of active learning in improving model accuracy compared to random sampling. In this simple example, given a small budget of points that can be labeled, the model needs to classify the two sets shown in the first image.
The second image shows the model performance as it randomly selected the subset of data points for training the model. This random sampling approach led to a less accurate model, as the points chosen were not adequately chosen to get the best-fit boundary line.
In contrast, the third image demonstrates data selected through an active learning technique. Here, the model has intelligently chosen the most informative data points, which are near to the decision boundary, therefore the most ambiguous ones, and labeling them provides the most significant improvement in the model's understanding of the decision boundary. As a result, the decision boundary in the third image is more accurate and better reflects the true separation between the classes.
This example shows the advantage of active learning: by strategically choosing data points that provide the most information, active learning can enhance the model accuracy and training process with fewer labeled examples.

The field of Active Learning has developed numerous methods for obtaining the most significant data: "Uncertainty Sampling", "Query by Committee," and "Bayesian Active Learning" are some of the most commonly used.
The technique analyzed in this project is "Coreset", an algorithm that aims to select the most informative samples given the limited samples by identifying a smaller subset of data that retains the essential characteristics and diversity of the entire dataset. This way, the learning algorithm can train on this smaller subset, reducing computational costs and improving the model's performance by focusing only on the most critical examples.

Formally, given a dataset D, a set of centers C, and a budget b, "Coreset" aims to find a subset K composed of b new centers from the dataset D such that the maximum euclidean distance between a data point and the nearest center is minimum.

$$max_{i \in D}\ min_{j \in C \cup K}\ \Delta(x_i, x_j)$$

$$|K| = b, K \subseteq D$$

Unfortunately, this problem is NP-Hard, meaning that finding the optimal solution is computationally infeasible when working with large datasets due to the exponential time complexity needed to solve it.
Therefore it is necessary to utilize heuristics or greedy algorithms that can find approximately a similar solution to the real one.
This project focuses on two of these approaches, "Farthest-First" and "Minimax".

2.3.1 Farthest-First Coreset

"Farthest-First Selection" is a greedy algorithm commonly used in coreset selection.
It is a relatively simple algorithm: it selects a randomly labeled data point and it searches for the furthest unlabelled data point from it and it is labelled and added to the list. This process is repeated until it has reached the budget.

Algorithm 1 Farthest-First Coreset Pseudocode

Input:
e_i: list of Embeddings
l_j: labelled data
b: budget

Implementation:
while b > 0 **do**
    r = random(l_j)
    u = argmax$_i$(dist(r, e_i))
    l_j.add(e_u)
    b = b - 1
end while
return l_j

2.3.2 Minimax Coreset

"Minimax Selection" is another greedy algorithm used to find the coreset of a dataset.
This algorithm's focus is on the unlabelled elements, since in every iteration it searches for the data point that is the furthest from its nearest labeled element, and it is later labeled and added to the set with the others.

Algorithm 2 Minimax Coreset Pseudocode

Input:
e_i: list of Embeddings
l_j: labelled data
b: budget

Implementation:
\# vector containing distance from nearest label of every data points
$min\_dist_i = \min(\text{dist}(e_i, l_j))$
\# matrix of euclidean distances between data points
$d_{ij} = \text{dist}(e_i, e_j)$
while $b > 0$ **do**
    $u = \text{argmax}_i \; min\_dist_i$
    l_j.add(e_u)
    $min\_dist_i = \min(min\_dist_i, d_{iu})$
    $b = b$ - 1
end while
return l_j

2.3.3 Comparison

Both approaches, while both greedy, have a lot of differences:
While the "Farthest-First" algorithm selects the new candidates based on their maximum distance from a singular labeled element, the "Minimax" algorithm, takes into consideration every labeled element, ensuring that all points are as close as possible to the centers.
These characteristics therefore can lead to totally different final sets: usually the first approach, while faster and simpler, leads to less accurate results, and is suitable for smaller datasets. The second one on the other hand, while being computationally more intensive, yields a more balanced and optimal final coreset.
In the end, both selections have their pros and cons, and are suitable in different situations: "Farthest-First" can be used as a rapid approach for smaller sets, while "Minimax" becomes the better tool when scaling as a source of information increases.

3 Results

In the following section, the results of the development process will be discussed:

3.1 Parameters and Generated Models

To test out the performance of my architecture in a flexible manner, I created a set of parameters that can be changed dynamically:

- (str) **AL_METHOD**: Active Learning algorithm used by the model, can be one of the following:
  - *random*: random sampling algorithm
  - *farthest_first*: farthest-first coreset algorithm
  - *minimax*: minimax coreset algorithm
- (str) **AL_EFFICIENCY**: Parameter that decides how the labels are handled, can be one of the following:
  - *normal*: After each timeframe, b new labels are created and memorized with the previous ones.
  - *optimal*: Every data point received in the current timeframe is labeled and memorized with the previous ones, acts as a higher bound to the normal one.
  - *forgetful*: Considers only the labels created during the current timeframe, previous ones are erased, acts as a lower bound to the normal one.
- (int) **BUDGET**: Number of posts that can be labeled in each timeframe.
- (float) **LEARNING_RATE** (LR): Model's learning rate, values between 0 and 1.
- (float) **DECISION_THRESHOLD** (DT): The model's threshold to decide if the content is relevant or not, value between 0 and 1.
- (float) **CLASSIFICATION_THRESHOLD** (CT): Threshold to classify the values in the "casualty" column as relevant or not, value between 0 and 1.
- (int) **TIMEFRAME**: Time interval in hours that splits the dataset into smaller streams that are individually considered during the active learning process.

With this set of parameters I managed to individually test each one by creating models with the following specifications:

MODEL NAME	AL_METHOD	AL_EFFICIENCY	BUDGET	LR	DT	CT	TIMEFRAME
"**my_model**"	"**minimax**"	"**normal**"	**1000**	**1e-5**	**0.5**	**0.1**	**24**
"**farthest_first**"	"farthest_first"	"normal"	1000	1e-5	0.5	0.1	24
"**random**"	"random"	"normal"	1000	1e-5	0.5	0.1	24
"**optimal**"	"minimax"	"optimal"	1000	1e-5	0.5	0.1	24
"**forgetful**"	"minimax"	"forgetful"	1000	1e-5	0.5	0.1	24
"**budget_10**"	"minimax"	"normal"	10	1e-5	0.5	0.1	24
"**budget_100**"	"minimax"	"normal"	100	1e-5	0.5	0.1	24
"**e-3**"	"minimax"	"normal"	1000	1e-3	0.5	0.1	24
"**e-4**"	"minimax"	"normal"	1000	1e-4	0.5	0.1	24
"**decision_0.25**"	"minimax"	"normal"	1000	1e-5	0.25	0.1	24
"**decision_0.75**"	"minimax"	"normal"	1000	1e-5	0.75	0.1	24
"**classification_0.9**"	"minimax"	"normal"	1000	1e-5	0.5	0.9	24
"**0.5_days**"	"minimax"	"normal"	500	1e-5	0.5	0.1	12

3.2 Evaluation Metrics

To evaluate in a complete way all the models, I have utilized the following metrics:

- **train_accuracy**: Accuracy of the model with the training data on the last epoch
- **time_elapsed**: Total amount of time needed to compute the labels and train the neural network at each timeframe
- **precision**: The proportion of correctly predicted positive instances among all predicted positives.

$$precision = \frac{true\_positives}{true\_positives + false\_positives}$$

- **recall**: The proportion of correctly predicted positive instances among all actual positives.

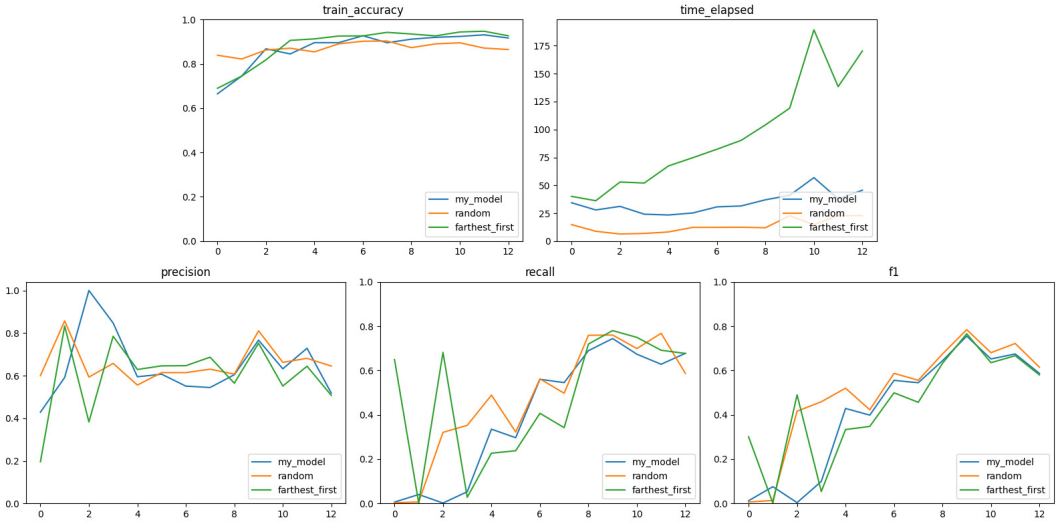
$$recall = \frac{true\_positives}{true\_positives + false\_negatives}$$

- **f1**: The harmonic mean of precision and recall, balancing the results from both metrics.

$$f1 = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

3.3 Analysis

3.3.1 Active Learning Techniques

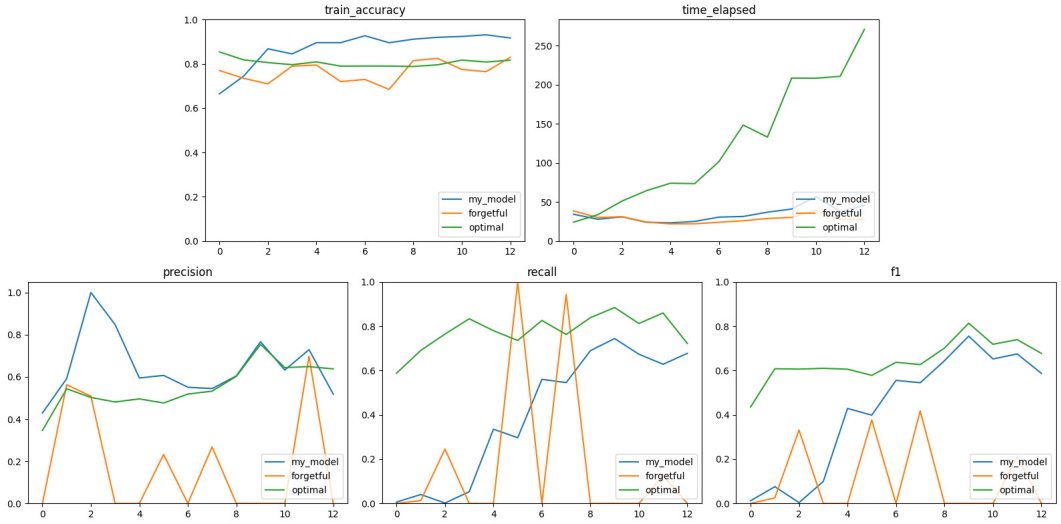


When comparing the results between the two coreset methods they appear quite similar: both of them approach higher train accuracy as the number of data points are added to their dataset, this improvement can

be explained due to their refinement in correctly identifying non-relevant data points overtime, as shown in the recall graph. However, the use of random sampling yields different outcomes: the training accuracy doesn't keep up with the active learning methods as the model trains on more labels.

Yet, when evaluating the results against the test set, with unseen data points, the differences in results between the active learning methods and the random one seem mostly negligible. The Farthest-First version, while during the initial development was the fastest version, has become the slowest one, and will probably need refactoring, as it scales to the number of labeled elements linearly fast.

3.3.2 Efficiency of Active Learning

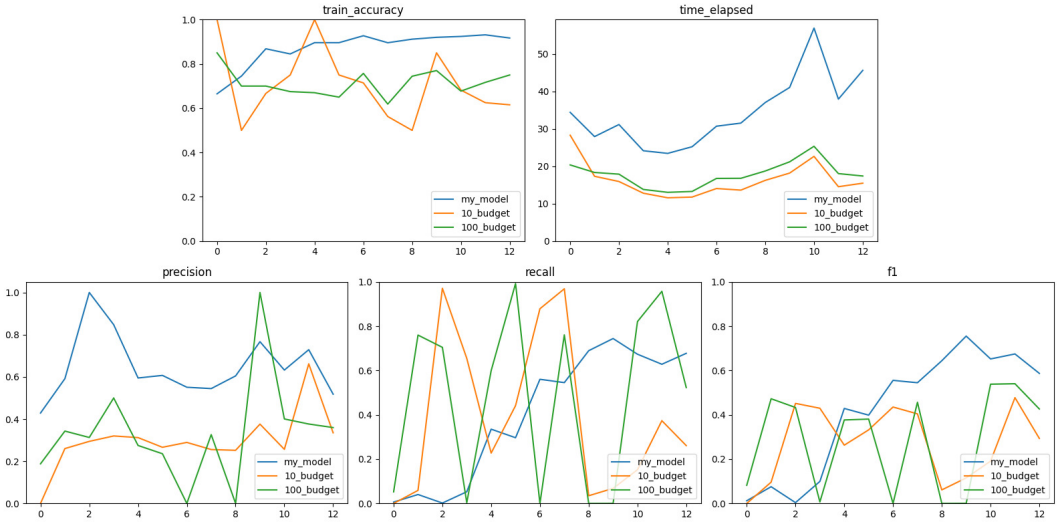


These plots illustrate the performance of the current implementation compared to the optimal approach, where the model is trained on every label of the current timeframe, and the forgetful approach, which only considers the budgeted labels of the current timeframe and serves as a lower bound.

The forgetful option isn't comparable to the other versions, as its lack of data in each timeframe leads to models with unsatisfying results. When comparing the optimal model with the normal one, it's evident that the former surpasses the other with a higher recall, avoiding most false negatives and therefore better results.

It needs to be addressed that the active learning approach, while yielding slightly lower results than the optimal one, it is much less time-consuming and less computationally intensive.

3.3.3 Budget Used

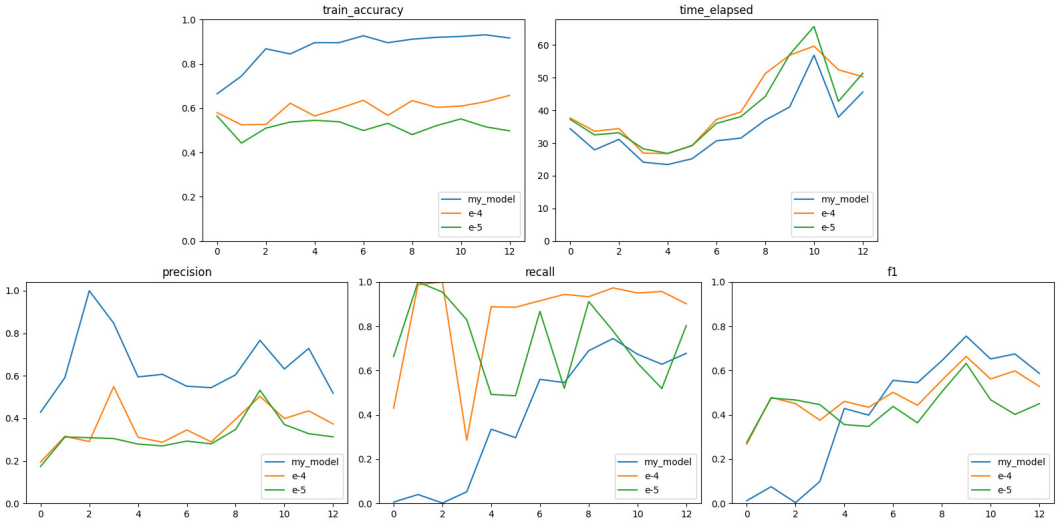


The quality of a neural network is proportional to the size of the training data it can analyze, as evidenced by the results. The model utilizing 1000 labels outperformed the others significantly, demonstrating improved performance with each timeframe.

Conversely, models with 10 and 100 labels per timeframe suffered from data scarcity, leading to instances of indiscriminate decision-making, such as classifying everything positively or negatively.

These graphs show that there is a crucial tradeoff during the selection of the budget, as a higher budget will lead to better performance but also higher computation times. The decision of which budget to use, depending also on the number of human validators available at the time, can make-or-break the successfulness of the operation.

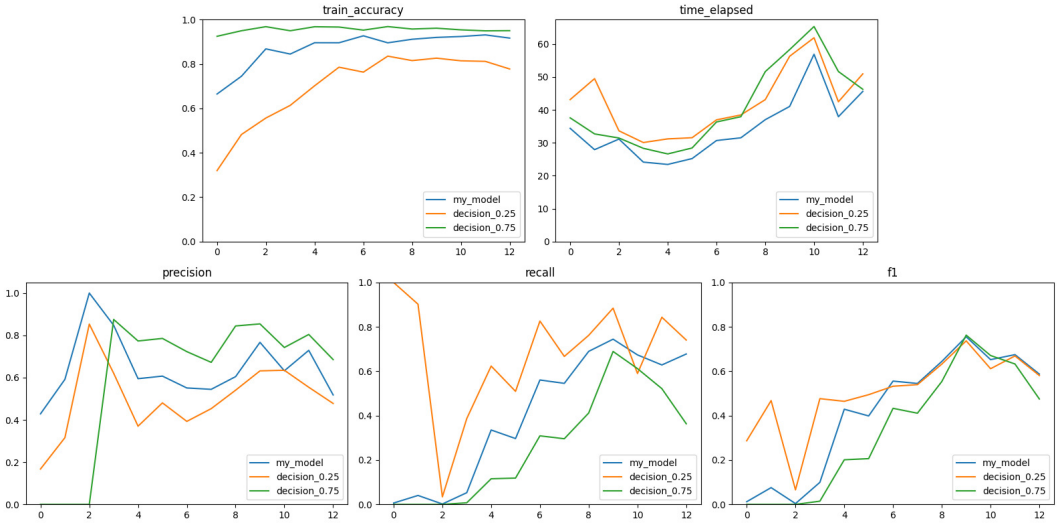
3.3.4 Learning Rate Used



The results here are clear: a higher learning rate leads to better results.
Because of the scarcity of data and the limited number of epochs, models with lower learning rates lack the resources to effectively optimize their weights, particularly impacting their precision performance.

Given the dynamic nature of the crisis under study, it is preferable to train the model with a higher learning rate to achieve better results more quickly.

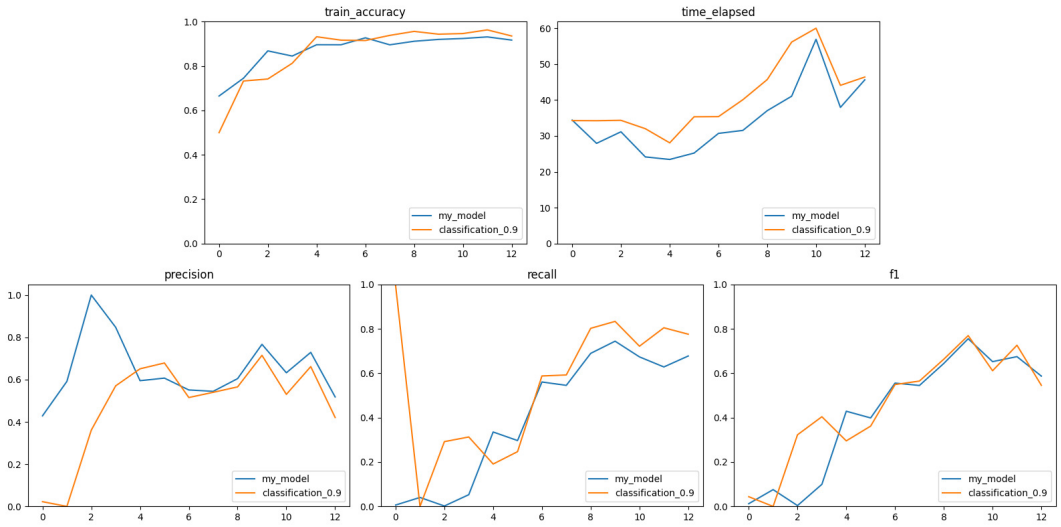
3.3.5 Decision Threshold Used



As expected, the position of the decision threshold is directly proportional to the model's precision and inversely proportional to its recall.
Ultimately, the F1 values for each decision threshold are mostly similar.

It is up to the user of the information system to select the preferred threshold: by having a higher precision with a higher decision boundary, conversely losing out on more false negatives that could contain crucial information; on the other hand allowing a lower limit, most relevant data points will be collected, but they will be combined with a higher amount of irrelevant posts.

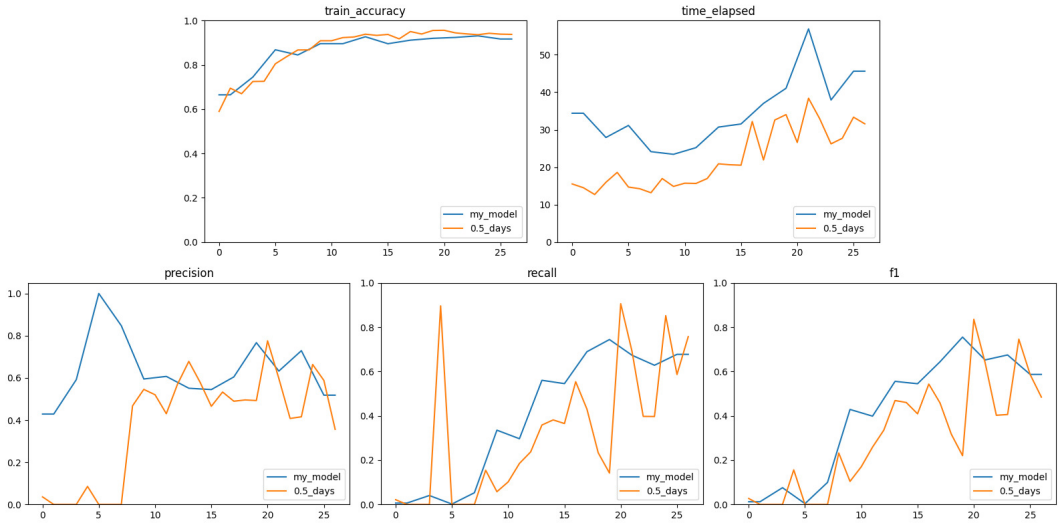
3.3.6 Classification Threshold Used



By utilizing a higher classification threshold, only the most certain positive labels are included, eliminating ambiguity. While this strategy reduces false negatives, it also compromises the model's precision.
Analogous to the "Decision Threshold Used" example, the F1 values for each classification threshold are predominantly similar.

Also similarly to that experiment, when a human validator is being more lenient when classifying relevant posts, the precision of the model will drop, likewise the amount of false negatives not recognized will also decrease. Alternatively, a strict validator will passively provoke the creation of a model with higher precision.

3.3.7 Timeframe Used



When using a smaller timeframe (while having the same throughput), the first models have a very bad performance, but as time goes on its values start getting more similar to the ones with a higher time gap. The final test data is also seemingly less stable than the my_model one, but it takes less to compute.

The selection of the timeframe in a real-life scenario depends mostly on the nature of such an event. In the case of instantaneous natural disasters, such as earthquakes, a smaller timeframe is definitely needed to rapidly get details about its damage, for conflicts like the Russo-Ukrainian one however, where the news can be handled at a slower pace, a bigger timeframe can be utilized to get a more coherent model.

4 Conclusions

When compared to random sampling, the utilization of active learning methods has had significantly better results during training. However, when tested with other unseen data, all techniques exhibited similar performance.

The productivity of active learning is clearly demonstrated during the analysis of the "AL_EFFICIENCY" parameter. It shows that training on a modest batch of the training data can gradually approach the same results as the optimal approach, utilizing just a proportionally small portion of it.

The selection of the budget is a crucial part of the development of an active learning infrastructure. Insufficient data results in ineffective models, whereas an excess of data reduces the usefulness of active learning tools.

Finally, the assignment of values to minor parameters such as the "DECISION_THRESHOLD" and "CLASSIFICATION_THRESHOLD", can be used to fine-tune the model to the user's needs. When they are low, more false positives, therefore irrelevant content, will be considered; on the other hand high values will avoid

this problem, but could be missing some other crucial information about the event.

To conclude, the successful implementation of an active learning architecture relies on identifying the most effective algorithms, appropriately allocating the budget, and fine-tuning the parameters. This project demonstrates that, when these elements are carefully optimized, active learning can significantly enhance the data collection and classification processes from social media during dynamic events.

4.1 Possible Future Developments

In summary, the successful implementation of an active learning architecture relies on identifying the most effective algorithms, appropriately allocating the budget, and fine-tuning the parameters. This project demonstrates that, when these elements are carefully optimized, active learning can significantly enhance the data collection and classification processes from social media during dynamic situations.
To enhance the findings of this project, future research could explore other Active Learning techniques, such as "Uncertainty Sampling" and "Query by Committee," to study their effectiveness with this particular dataset.

On a broader scale, it would be valuable to test the importance of other steps in the pipeline. For example, future studies could focus on the initial filtering of data using specific keywords or topics. Otherwise, another possible track could be the development of an Online Learning algorithm that adapts to incoming data, potentially adjusting parameters like the active learning technique in real time.

There are numerous opportunities to expand this project and advance research toward creating a state-of-the-art application capable of detecting and summarizing the main events during an ongoing disaster that could prove to be crucial in informing and protecting citizens.
By continuing to refine these information systems, we move closer to a future where timely, accurate data can help crisis response, ultimately making the world a safer place.

References

[1] Carlo Bono and Mehmet Mulayim. *Learning Early Detection of Emergencies from Word Usage Patterns on Social Media*, pages 308–323. 05 2023.

[2] DataCamp. Active learning: Curious ai algorithms, 2018. https://www.datacamp.com/tutorial/active-learning.

[3] Ren Pengzhen, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij Gupta, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *ACM Computing Surveys*, 54:1–40, 10 2021.

[4] Daniela Pohl, Abdelhamid Bouchachia, and Hermann Hellwagner. Active online learning for social media analysis to support crisis management. *IEEE Transactions on Knowledge and Data Engineering*, 32(8):1445–1458, 2020.

[5] Alan Ritter, Mausam Mausam, Oren Etzioni, and Sam Clark. Open domain event extraction from twitter. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1104-1112, 08 2012.

[6] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Tweet analysis for real-time event detection and earthquake reporting system development. *IEEE Transactions on Knowledge and Data Engineering*, 99, 11 2012.

[7] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach, 2018.