# BT5151

# Team Project

# Context

- You are working for a news agency that aims to bring the information on disasters to public attention as soon as possible.
- To source the news your department relies on scraping twitter posts algorithmically. A simple approach is to detect key words that may suggest a disaster.
- But this approach can have many false positives. For example, the post: *"The crowd was **on fire** at the concert last night!"* does not imply a real disaster.
- So detection of words/phrases that may indicate disasters is not sufficient.
- The entire context of the tweets needs to be processed to classify the tweet into (1) Disaster or (2) No Disaster

# Goal

- Your Goal is to build an **NLP model** *M* that can detect tweets about disasters by performing two tasks that will be combined in a multi-task model.
- Given the text of a tweet, we will build NLP models to perform:

    1. $T_1$: **Disaster detection:** classify the tweet into one of two categories: “disaster” or “no disaster”
    2. $T_2$: **Sentiment classification:** classify the sentiment of the tweet (next slide) to gauge the severity of the tweet.

We will explore ways to use “signals” from the auxiliary task (sentiment classification) using multi-task learning (MTL). MTL is helpful in many occasions — when there are insufficient labelled samples for the main task, or to improve model performance using auxiliary data in general (even when there are sufficient labelled samples for the main task).

# Datasets

1. $D_1$: The dataset for Task 1 (Disaster classification).
   - Download the dataset from this Kaggle competition.
   - Binary classification task with two labels (1) real disaster and (0) not a real disaster.
2. $D_2$: The labels for Task 2 (Sentiment classification) is not readily available in $D_1$. Therefore, we will obtain the labelled data for Sentiment classification task from another source.
   - Download this Kaggle dataset, which is has sentiment labels for tweets.
   - There are 3 sentiment labels in this dataset: (0) Neutral (1) Negative, (2) Positive
     In our application, this could used to gauge the severity of the tweet.

# 1. Dataset Preprocessing

2 marks

- You have two datasets:
  $D_1 = \{(text_i, label_i^1)\}_{i=1}^{N_1}$ with $N_1$ samples,
  and
  $D_2 = \{(text_j, label_j^2)\}_{j=1}^{N_2}$ with $N_2$ samples,
  where $label^1$ are disaster labels and $label^2$ are sentiment labels.
- Build a corpus by combining the texts from $D_1$ and $D_2$. All corpus-level tasks (such as vocabulary creation) can be done using this combined dataset.
- Tokenize the words and prepare a pipeline for tensor representation of text. The tensor representation may depend on the NLP model you choose to use.
- Split the training data available from Kaggle to use 80% for training and 20% for validation, to obtain $D_1^{train}$ and $D_2^{train}$ for training, and $D_1^{val}$ and $D_2^{val}$ for validation.
- We will use the provided test data $D_1^{test}$ for testing/submission on Kaggle.

# Datasets

- $D_1$: disaster classification data (train.csv, test.csv)
- test.csv: $D_1^{test}$ (no labels available)
- train.csv: $D_1^{train} + D_1^{val}$
- $D_2$: sentiment classification data (tweets.csv): $D_2^{train} + D_2^{val}$

$D_1$

| Text | Disaster Label |
|---|---|
| | |
| | |
| | |
| | |

$D_2$

| Text | Sentiment Label |
|---|---|
| | |
| | |
| | |
| | |

# II. Model Building

- Choose any neural network based NLP model that takes a text as input and produces predictions for the two tasks:
  $M(text) \rightarrow (pred^1, pred^2)$
  $pred^1$ is the predicted class (probability) for the binary classification task $T_1$, and
  $pred^2$ is the predicted class (probability) for the multi-class classification task $T_2$.
- Assume the model is deployed in a server within your news agency. Explain the inputs to the model and predictions that it is expected to output. Also explain how the predictions will be used by reporters.

2 marks

- Note that at this stage, the model is not trained. Training and evaluation steps are described in the following slides.

Note: If you use a pre-trained NLP model (e.g., from hugging face) you may use their vocabulary and tokenization method. Give appropriate citations in your report.

# III. Neural Multi-Task Learning

1. Train the model $M$ only for $T_1$ on $D_1^{train}$. Call the trained model $M_{D_1}$. Evaluate $M_{D_1}$ on $D_1^{val}$ and report its performance metric (F1 score) for the first task, $Perf_{T_1}(M_{D_1} | D_1^{val})$. Also print the confusion matrix. **2 marks**

2. Train the model $M$ only for $T_2$ on $D_2^{train}$ . Call the trained model $M_{D_2}$. Evaluate the trained $M_{D_2}$ on $D_2^{val}$ and report its performance metric (Accuracy) $Perf_{T_2}(M_{D_2} | D_2^{val})$. Also print the confusion matrix. **2 marks**

# III. Neural Multi-Task Learning

3. Estimate the sentiment labels for the tweets in $D_1$:
   For all *text* in $D_1$, create $\widehat{label^2}$ from the $pred^2$ output of $M_{D_2}(text)$. This will give you an augmented dataset
   $\widehat{D}_1 = \{(text_i, label_i^1, \widehat{label^2}_i)\}_{i=1}^{N_1}$.

Note that this data consists of $\widehat{D}_1^{train}$ and $\widehat{D}_1^{val}$ corresponding to the train and validation sets respectively.
Analyze and report the predicted sentiment labels for some of the tweets from the disaster dataset. Use at least 1 tweet from each class for your analysis.

$$\widehat{D}_1 = \{(text_i, label_i^1, \widehat{label^2}_i)\}_{i=1}^{N_1}$$

| Text | Disaster Label | Predicted Sentiment Label |
|---|---|---|
| | | |
| | | |
| | | |
| | | |

3 marks

# III. Neural Multi-Task Learning

4. Create another dataset by combining $D_1^{train}$ and $D_2^{train}$:

   $D_{12} = \{(text_i, label_i^1, Null)\}_{i=1}^{N_1^{train}} \cup \{(text_j, Null, label_j^2)\}_{j=1}^{N_2^{train}}$

   where $N_1^{train}$ is the number of training samples in $D_1^{train}$.

1 mark

| Text | Disaster Label | Sentiment Label |
|---|---|---|
| | | Null |
| | | Null |
| | Null | |
| | Null | |

Note, this dataset has missing labels and is different from the augmented dataset $\widehat{D}_1$.

# III. Neural Multi-Task Learning

5. We want to train $M$ for both $T_1$ and $T_2$ on $D_{12}$ by minimizing a weighted loss $\lambda_1 l_1 + \lambda_2 l_2$, where $\lambda$s are positive scalar weights between 0 and 1: higher the $\lambda$ more the emphasis on corresponding task while training; $l_1, l_2$ are task-specific loss functions.

   3 marks

   - When the label of a data sample is *Null*, the corresponding loss is considered as 0. What is the effect of this during network training?
   - Call this model $M_{D_{12}}$.
   - After training with $\lambda_1 = \lambda_2 = 0.5$, evaluate $M_{D_{12}}$ on $\widehat{D}_1^{val}$ and report the performance for both the tasks, i.e., $Perf_{T_1}(M_{D_{12}} | \widehat{D}_1^{val})$ and $Perf_{T_2}(M_{D_{12}} | \widehat{D}_1^{val})$. Print confusion matrices for both.

6. Disaster classification is our primary task. Obtain the best values of the hyper-parameters $\lambda_1, \lambda_2$ using any hyperparameter tuning method to optimise the metric $Perf_{T_1}(M_{D_{12}} | D_1^{val})$ for task 1. Note, $\widehat{D}_1^{val}$ and $D_1^{val}$ have the same labels for task 1.

   2 marks

# III. Neural Multi-Task Learning

7. Train the model $M$ on $D_{12}^{train}$ with the best $\lambda$s obtained in step 6. Call it $M^*_{D_{12}}$. Evaluate it on $D_1^{test}$ for the tasks and report $Perf_{T_1}(M^*_{D_{12}} \mid D_1^{test})$ by participating in the challenge. Submit a screenshot of your leaderboard score and position. **2 marks**

8. In step 7, you performed multi-task learning, where data for the second task is externally sourced. Is the best value of hyper-parameter $\lambda_2$ from step 6 zero or positive, i.e., is $\lambda_2 = 0$ or $\lambda_2 > 0$? What does the value of $\lambda_2$ convey? Does the externally sourced sentiment data improve the model accuracy for our primary task of disaster classification? **2 marks**

9. For multi-task learning why is the augmented dataset $\widehat{D}_1$ used only for testing but not for training? **2 marks**

# IV. Random Forest

In this part we will use a Random Forest model to for disaster classification and try to improve its result by utilizing the sentiment classification dataset. The preprocessing steps for this model may or may not be what was used in part III.

1. How can the MTL strategy of loss combination in question 5 (part III) be used with a Random Forest? Explain. **2 marks**

2. Try the following approach:
   Use the augmented dataset $\widehat{D}_1^{train}$ obtained in question 3 (part III) to train a Random Forest for prediction of $label^1$. **2 marks**

   Use the $\widehat{label}^2$ for weighting the samples while fitting the Random Forest.
   The weight $w_i$ for a sample $i$ can be defined as

$$w_i = \begin{cases} \lambda_p & \text{if } \widehat{label}_i^2 = \text{positive} \\ \lambda_n & \text{if } \widehat{label}_i^2 = \text{negative} \\ \lambda_0 & \text{if } \widehat{label}_i^2 = \text{neutral} \end{cases} \quad \text{where } \lambda_p, \lambda_n, \lambda_0 \in [0,1]$$

   - Evaluate the model on $\widehat{D}_1^{val}$ and report the performance for task 1. Print the confusion matrix.

# IV. Random Forest

3. Use any hyper parameter tuning method to find the best values of the $\lambda$s to maximize $Perf_{T_1}(M_{\widehat{D}_1^{train}} \mid \widehat{D}_1^{val})$. Is the optimal value result in $\lambda_p = \lambda_n = \lambda_0$? If not what is the ordering of the $\lambda$s, and what can we infer from this ordering? **2 marks**

4. Train the model with the best $\lambda$s obtained in step 3. Evaluate it on $\widehat{D}_1^{test}$ for the tasks and report $Perf_{T_1}(M_{\widehat{D}_1^{train}} \mid D_1^{test})$ by participating in the challenge. Submit a screenshot of your leaderboard score and position. **2 marks**

# V. Further Exploration

Think of “X” other strategies to train a model for disaster detection which uses data/models for sentiment classification. They may be similar to or different from the approaches in parts III and IV.

For each strategy:

i. Explain your strategy in words and why you think *it is expected* to work (before implementing and observing its performance). — 3 marks

ii. Implement your strategy and participate in the challenge. Submit a screenshot of your leaderboard score and position. — 6 marks

X = (<number of team members> - 1). E.g., if there are 3 team members, X = 2. Marks are for all X strategies together.

# Rules (1)

- It is important to explain the design choices in each step you perform (in preprocessing, feature engineering, model training…).
- Ask yourself why you are performing each step and write the reason
- Explain what inference you draw or what you observe after each relevant step
- Your explanation should be in your own words (plagiarism will be penalized)
- Your submission should be readable like a report

# Rules (2)

- You can use any online resource including code: cite them
- No restrictions on choice of models used within the project, except for what is explicitly stated.
- The submitted code should execute without any errors. Jupyter files that are submitted should have all the outputs after code execution. If there are any errors in any question, no marks will be given for that part.
- You can use a subset of the data if computational resources are insufficient for model training.
- Note that your leaderboard position/score will NOT be used for evaluating your project. It is only to obtain test scores to compare your own model variants and, for your reference, to see the model's "global" standing.

# Submission

- Per team:
  1. One (or more) Jupyter Notebook(s)
  2. Also submit the same notebook(s) converted to HTML
  3. Document (filename: main, format: ppt/word/pdf…) containing
     - Names and IDs of all team members
     - Leaderboard screenshots
     - [If there are multiple notebooks] sequence in which to view files
  - Upload 1 zipped folder containing all files
- Submission Deadline: April 9, 2023 (11:59 PM)

# Consultation

- You are encouraged to discuss your project ideas with the TAs before submission and get their feedback on your solutions.
- You may consult them during tutorials and in additional consultation slots (over Zoom):
    - Rishav: Tuesdays 11 am - 12 noon
    - Aishwarya: Thursdays 2-3 pm
    - Debabrata: Fridays 4-5 pm
    - Meng Ziwei: Wednesdays 4-5 pm
    - He Ziyang: Thursdays 3-4 pm
- Please use the discussion forum to ask any general questions about the project