**Text Classification of Microblog data**

# Executive Summary

Text classification of tweets into 10 different classes based on features such as user description, hashtags and location data were explored. All 4 features are shown to improve the classification, and the order of importance of the different features are ranked as such: (1) Text, (2) User Description, (3) Hashtags, (4) Location information. Early and late fusion methods of integrating the features was also compared, with the late fusion performing better as it allows optimisation when modelling each feature and different weights can be assigned to different features easily. Different machine learning techniques, namely Naïve Bayes, K nearest neighbours and random forest, were also explored. Naïve Bayes tend to do well empirically in this experiment as compared to the rest; possibly due to the assumption of independence of the different term probabilities being largely valid. The final model combining all features in a late fusion model is able to achieve a f1 score of 0.7009 calculated based on a 10-fold cross validation.

# Introduction

Microblogs are user generated content that comes in short segments and can be a rich form of information due to the large number of tweets generated daily. The text classification of microblogs into different classes can create value by diverting the right information to relevant departments for further analysis, or to relevant audiences.

# Methodology

## Program Structure

The programme consists of the following:

1. Pre-processing of text data
2. Feature selection based on Document frequency
3. Classification using machine learning techniques
4. Testing and optimisation based on precision, recall, and f1 scores

Which will be further elaborated in the sections below.

## Data set

The data consist of a set of tweets with 10 classes, 600 tweets per class class, crawled from Twitter. The 10 pre-defined classes are Health, Business, Arts, Sports, Shopping, Politics, Education, Technology, Entertainment, and Travel.

## Pre-processing

The pre-processing section consist of the following steps:

1. Tweets in json format was read
2. Attributes was extracted from tweet object. The following attributes were extracted:
   1. “Text”: The actual text of the status update. This provides the content of the microblog.
   2. “User”, “Description”: The user’s description of their account. This provides information on what type of user and account it is, which influences the type of content they post.
   3. “Entities”, “Hashtag”: Hashtags parsed out of the text of the Tweet. Hashtags are typically keywords that are flagged out in the user-generated tagging system that allows others to easily find messages with a specific theme or content.
   4. Location data such as

“User”, “location”: Location which the user is based in as stated in the profile.

“Place”, “name”: Location name where the Tweet was sent.

“Place”, “country”: Country where the Tweet as sent from

“Place”, “place\_type”: Type of location the Tweet was sent from, e.g. City

Location data was added as a feature as location can help to disambiguate different acronyms and use of words.

1. Tweets were treated with a pre-processing function, which does the following in the order given below:
   1. Convert to lowercase
   2. Remove URLs
   3. Remove Mentions
   4. Remove hashtag symbol
   5. Remove time
   6. Remove punctuations
   7. Word tokenize and stemming
   8. Remove stop words
   9. Remove low frequency words
2. Treated data for each extracted attribute are saved into individual text (.txt) file for further processing.

## Feature selection

Feature selection was performed using document frequency thresholding and was completed as part of the pre-processing step. Document frequency (DF) is the number of documents containing the term. As terms with high DF are preferred during Text classification, terms with DF less than 2 was removed from the corpus.

## Classification methods

### Early fusion

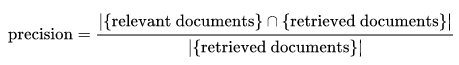
In early fusion, the features are merged into a single corpus before applying different machine learning techniques (Naive Bayes, K Nearest Neighbours, and Random forest) to classify the Tweets.

### Late fusion

In late fusion, each feature is trained using different machine learning models (Naive Bayes, K Nearest Neighbours, and Random forest) optimised using 10-fold cross validation and f1 scores to get a prediction probability for each class. The prediction probabilities of all features are integrated into one prediction using a weighted average ensemble, where the weights are optimised using f1 scores.

## Testing procedure

The text classification model is tested based on Precision, Recall, and f1 score, where







# Results and Discussion

## Early fusion model

Features, text, description, hashtags, location are merged into a single corpus in the pre-processing step. The Corpus is loaded in the classifier and modelled with 3 different machine learning techniques, Naive Bayes, K Nearest Neighbours, and Random forest, to classify the Tweet.

Based on the results in Table 1, the Naïve Bayes model give the highest precision and recall scores. There is significant improvement in the precision and recall scores as compare to the baseline text only classifier. However, as compared to the late fusion model in the section below, the performance of the early fusion model is not as good.

Table 1: Comparison of different machine learning models for early fusion model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Features | Model | Precision | Recall | F1 Score |
| Text (baseline) | Naïve Bayes | 0.6322 | 0.6237 | 0.6237 |
| Text + Description + Hashtags + Location | **Naïve Bayes** | **0.6871** | **0.6765** | **0.6768** |
| K Nearest Neighbour | 0.5903 | 0.5746 | 0.5749 |
| Random Forest | 0.6052 | 0.6010 | 0.5911 |

## Late fusion model

### Optimisation of Machine learning models for each feature

For each feature, 3 different machine learning models (Naive Bayes, K Nearest Neighbours, and Random forest) were fitted and the precision and recall were calculated to determine the model of best fit. Based on the results as shown in Table 2, the Text, Description, and Hashtags are best fitted using the Naïve Bayes model, while the location feature is best fitted with a Random Forest, to give the highest precision and recall. These models are thus selected to be used in the late fusion model and are given different weights in the final optimisation.

Table 2: Comparison of different machine learning models for each feature

|  |  |  |  |
| --- | --- | --- | --- |
| Feature | Machine learning Model | Precision | Recall |
| Text | **Naïve Bayes** | **0.6322** | **0.6237** |
| K Nearest Neighbour | 0.6553 | 0.2127 |
| Random Forest | 0.5709 | 0.5637 |
| Description | **Naïve Bayes** | **0.4888** | **0.4663** |
| K Nearest Neighbour | 0.2933 | 0.2675 |
| Random Forest | 0.4884 | 0.4660 |
| Hashtag | **Naïve Bayes** | **0.6037** | **0.4673** |
| K Nearest Neighbour | 0.5027 | 0.3611 |
| Random Forest | 0.5944 | 0.4491 |
| Location | Naïve Bayes | 0.2649 | 0.2302 |
| K Nearest Neighbour | 0.2703 | 0.2093 |
| **Random Forest** | **0.2840** | **0.2470** |

### Results of adding different features to the basic classifier

The predicted probability of the different classes is given different weights and optimised to find the weight giving the best F1 score. Description, hashtag and location features are added onto the basic text classifier individually and combined in total.

Based on the results in Table 3, all features, description, hashtag and location, improve the F1 score of the text classifier. Description provides the most improvement, followed by hashtags and finally location, as observed from the F1 score and the optimised relative weights, where the feature which provides a more improvement is given a higher weight.

In the model that combines all features, the **F1 score of 0.7009** is the best of all the models discussed in this paper and thus is chosen to be the final model.

Table 3: Comparison of Late fusion results based on features added

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Features Used | Weights | Precision | Recall | F1 score |
| Text (baseline) | [1] | 0.6322 | 0.6237 | 0.6237 |
| Text + Description | [0.6 0.4] | 0.6948 | 0.6850 | 0.6875 |
| Text + Hashtag | [0.7 0.3] | 0.6502 | 0.6433 | 0.6450 |
| Text + Location | [0.9 0.1] | 0.6438 | 0.6362 | 0.6381 |
| Text + Description + Hashtag + Location | **[0.55 0.25 0.15 0.05]** | **0.7076** | **0.6990** | **0.7009** |

# Discussion

## Features

The 4 features explored are ranked in the following order in terms of importance, as seen in the relative weights of each feature in the final model:

1. Text
2. User Description
3. Hashtag
4. Location

A possible explanation for this is due to the richness of information each of the features provide. Text is the content of the tweet and has most direct influence on the tweet classification.

User description tells us about the type of account and user this tweet is from. This is a good supplement to the text content as the description is usually quite accurate and descriptive of the purpose of the account, for example a news channel will mention the word “news” and a travel channel is likely to mention the word “travel”. Certain channels, official accounts are also dedicated only to one type of tweet, making the description more relevant to tweet classification. The drawback is that the common users are likely to tweet multiple types of tweet from the same account.

Hashtags are keywords flagged out by the user themselves in the user-generated tagging system that allows others to easily find messages with a specific theme or content. Although hashtags are also in the text content of the tweet, calling hashtags out separately as a feature to be trained separately improves the classification as it gives emphasis to these keywords.

Location data was also added as a supplement to the classifier as it can help to disambiguate different acronyms and use of words. A hypothetical example is ERP, which will stand for “Electronic Road Pricing” in Singapore, making the tweet more likely to be news or traffic related, where else in other parts of the world people know it as “Enterprise resource planning” which will likely be a Technology related tweet. However, as location tagged tweets and user account location information is sparse, location alone has poor precision and recall (see table 2) but makes for a good supplement to the other features.

Overall, all features improve the performance of the text classifier and the combined classifier that uses all features has the best precision, recall and f1 scores. Future work for further improvement of the classifier can consider using comments, social relations between users, images or URLs associated with tweets.

## Comparison of Early Fusion vs Late Fusion

The method of combining features before or after learning concepts has significant impact on the final F1 score. Comparing the early and late fusion, Late fusion performance classifier has a much better f1 score (late fusion f1 = 0.7009 vs early fusion f1 =0.6768) holding the features constant. This is likely because late fusion allows the different features to be weighted differently, where else the early fusion applies the same weight to all text terms in the combined corpus used for the training of the final model.

## Comparison Machine learning techniques

3 machine learning techniques (1) Naive Bayes, (2) K Nearest Neighbours, and (3) Random forest are used to optimise the different classifiers in both the early and late fusion models. The Naïve Bayes model tend to do well in terms of classification as compared to the other 2 based on empirical data. Naïve Bayes assumes independence of the different term probabilities, which largely valid in this case. As the data set contains a long tail of infrequent words, the data can be considered sparse and the Naïve Bayes model tend to do well with sparse data as well.

# Conclusion

Text classification of tweets can be improved by including more features, such as user description, hashtags and location data. All features are shown to improve the classification, and the order of importance of the different features are ranked as such: (1) Text, (2) User Description, (3) Hashtags, (4) Location information. Early and late fusion methods of integrating the features was also compared, late fusion performs better as it allows optimisation when modelling each feature and different weights can be assigned to different features easily. Naïve Bayes tend to do well empirically in this experiment as compared to the other machine learning techniques like K nearest neighbour and random forest. Future work for further improvement of the classifier can consider adding more features by using comments, social relations between users, images or URLs associated with tweets.