Natural Language Processing using fastText

Group N

Abu Talha Khan, Mahendi Jigarkumar Raval, Babari-Isa Blessing Vulasi

1. Brief Introduction

fastText is a code developed at Facebook AI Research for classification of text using machine learning[1]. The references given in the fastText code have been cited in many papers and have appeared in good conferences. It is built on Mac OS and Linux and uses C++11 features and requires good C++11 supported compiler[1]. Amazon AWS Ubuntu virtual machine has been used to implement the code. The goal is to assign documents or sentences to one or more categories, using various techniques like bigrams, hierarchical softmax, increasing learning rate and epochs, etc. and review the scores of the models.

2. Description of Original Dataset

The dataset used in the original project was collected from the stack exchange website on the topic of cooking [2]. Since the replication of this work was carried out in a virtual machine, command lines were used to acquire the dataset onto the workspace. All of these labels were concatenated with a "__label__" prefix. This prefix allows fastText to understand between the classes and the rest of the texts. The dataset is comprised of 15404 observations with 734 different labels (classes) which was then split into 12404 observations for training and 3000 for validation purposes.

3. Replication of original work

Using fastText code in amazon Ubuntu virtual machine for text classification using predefined labels. It can classify documents or text into one or more categories. This is done using machine learning using classified data as learning set[1]. Automatically classified questions from stackexchange on cooking are used with several tags as the classification labels or categories[1]. To build the model the fastText code is downloaded in the virtual machine. This requires a c++ complier with c++11 support. The first step is to build the code in the fastText directory. To download the default dataset, that is cooking questions from stackexchange with associated tags we use the link provided in the fastText tutorial.

Looking at the default data using the "head" function, we observe that each line consists of the tag preceded by "__label__" and followed by a white space and the question text[1]. If the label consists of more than one word they are joined using an hyphen "-" and multiple labels are separated by a white space. The "wc" function in command line gives the number of words and observation in the data. Stackexchange cooking questions data has 15404 questions with the associated labels. The data is roughly divided into test and train by 80-20 split ratio with 12404 examples in training set and 3000 examples in validation set.

Running a simple supervised classifier gives poor results. These results also show that the data has 735 total labels. Using "predict" function to enter desired text to be classified by the above developed model. Example, "Which baking dish is best to bake a banana bread ?" will be classified into the label "baking"[1]. The function "test" used with the validation dataset gives precision and recall of the model. Here, we use these two scores to compare and evaluate all the models. Multiple labels can also be predicted using a specific number with the "predict" function.

Various techniques can be applied to improve the performance of the model. Cleaning the data to remove punctuations and covert all text into lower case is one of them. Increasing the number of epochs to 25 and learning rate to 1.0 helps improve the precision and recall significantly as seen in table 1. Bigrams are used to take two consecutive tokens instead of individuals ones, this method is used by adding "–wordNgrams 2" extension to the supervised model input command[1]. This takes huge memory and may not run on the virtual machine this problem can be solved using hierarchical softmax which is a loss function that helps in achieving faster computation, this is added with option "loss hs"[1]. Multi-label classification model is also built using the “loss one-vs-all” command with the supervised model[1]. All these methods help improve the scores by a great extent the comparisons have been given in table 1 below.

Model	Precision	Recall
Simple supervised model	0.139	0.0603
Cleaned data	0.171	0.074
Epoch = 25	0.517	0.224
Learning rate = 1.0	0.586	0.253
Hierarchical softmax with word bigrams	0.14	0.0607
Multi label model softmax with word bigrams	0.308	0.133
Combined model with lr = 0.5	0.605	0.262

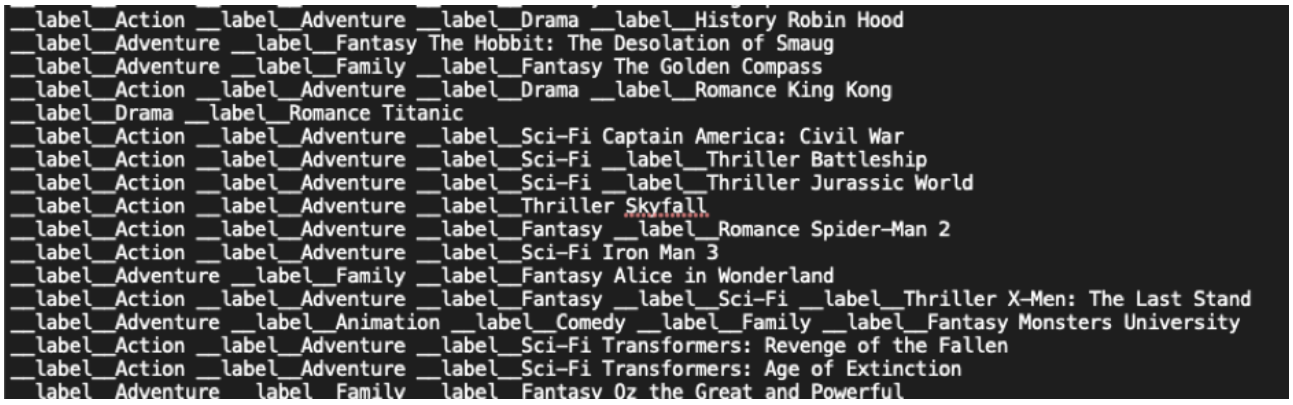
Table1 : precision and recall of all the models trained and validated with original stackexchange cooking data

4. Construction of New Data

All the .txt files created and the python codes used to create the data have been given in the GitHub repository - https://github.com/AbuTalha777/Natural-Language-Processing-Using-fastText

4.1. imdb movies dataset

IMDB API is imported and used in python to generate this data set. IMDB instance is generated and “get_top250_movies()” function is used to get the data of the top rated 250 list on imdb. The genres in the ‘genres’ category with “__label__” prefix are used as the categories followed by the movie title given in the ‘title’ category as the text. Any whitespace in the genre is replaced with “-”. The following image shows the output of the head function for the movie dataset. As this is a small dataset it is used as the validation data. For test data kaggle imdb dataset(https://www.kaggle.com/carolzhangdc/imdb-5000-movie-dataset) is used which contains 5043 observations[6]. In a similar method as above, genres was used for labels and movie titles were used for text. This has 26 total number of labels. This is a multi-labelled data. These dataset are run with all the models as mentioned in table 2.

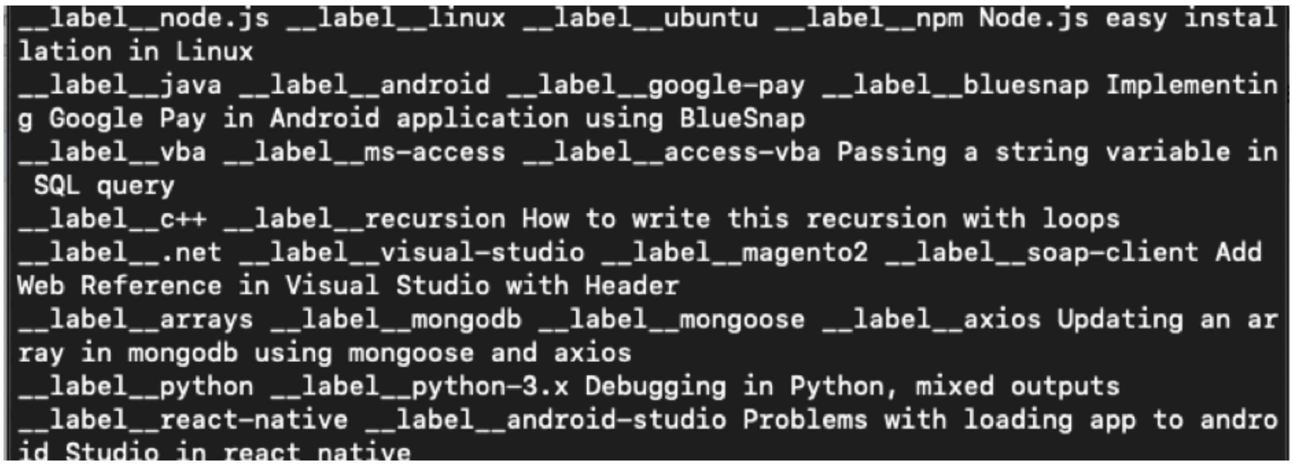


```
__label__Action __label__Adventure __label__Drama __label__History Robin Hood
__label__Adventure __label__Fantasy The Hobbit: The Desolation of Smaug
__label__Adventure __label__Family __label__Fantasy The Golden Compass
__label__Action __label__Adventure __label__Drama __label__Romance King Kong
__label__Drama __label__Romance Titanic
__label__Action __label__Adventure __label__Sci-Fi Captain America: Civil War
__label__Action __label__Adventure __label__Sci-Fi __label__Thriller Battleship
__label__Action __label__Adventure __label__Sci-Fi __label__Thriller Jurassic World
__label__Action __label__Adventure __label__Thriller Skyfall
__label__Action __label__Adventure __label__Fantasy __label__Romance Spider-Man 2
__label__Action __label__Adventure __label__Sci-Fi Iron Man 3
__label__Adventure __label__Family __label__Fantasy Alice in Wonderland
__label__Action __label__Adventure __label__Fantasy __label__Sci-Fi __label__Thriller X-Men: The Last Stand
__label__Adventure __label__Animation __label__Comedy __label__Family __label__Fantasy Monsters University
__label__Action __label__Adventure __label__Sci-Fi Transformers: Revenge of the Fallen
__label__Action __label__Adventure __label__Sci-Fi Transformers: Age of Extinction
__label__Adventure __label__Family __label__Fantasy Oz the Great and Powerful
```

Figure 1: Head function output for IMDB movie data

4.2. Stack overflow Questions

Using stackapi, an instance of StackAPI was created with ‘stackoverflow’ as the parameter in python. Questions from 150 pages of size 100 each were stored in a txt file. The associated tags with every question are given under ‘items’ category. Each tag is prefixed with “__label__” followed by whitespace and a question as the text given in ‘title’ column. 15000 questions with associated tags are collected by this method and the generated text file is loaded under the fastText category in the VM. This is a multi-labelled data. This data is divided by an 80-20 ratio into train and validation sets with 12000 questions in training set and 3000 questions in validation set. This data contains 6781 total labels, which is high compared to the default data. The results for all the models applied to this data are given in table 2.



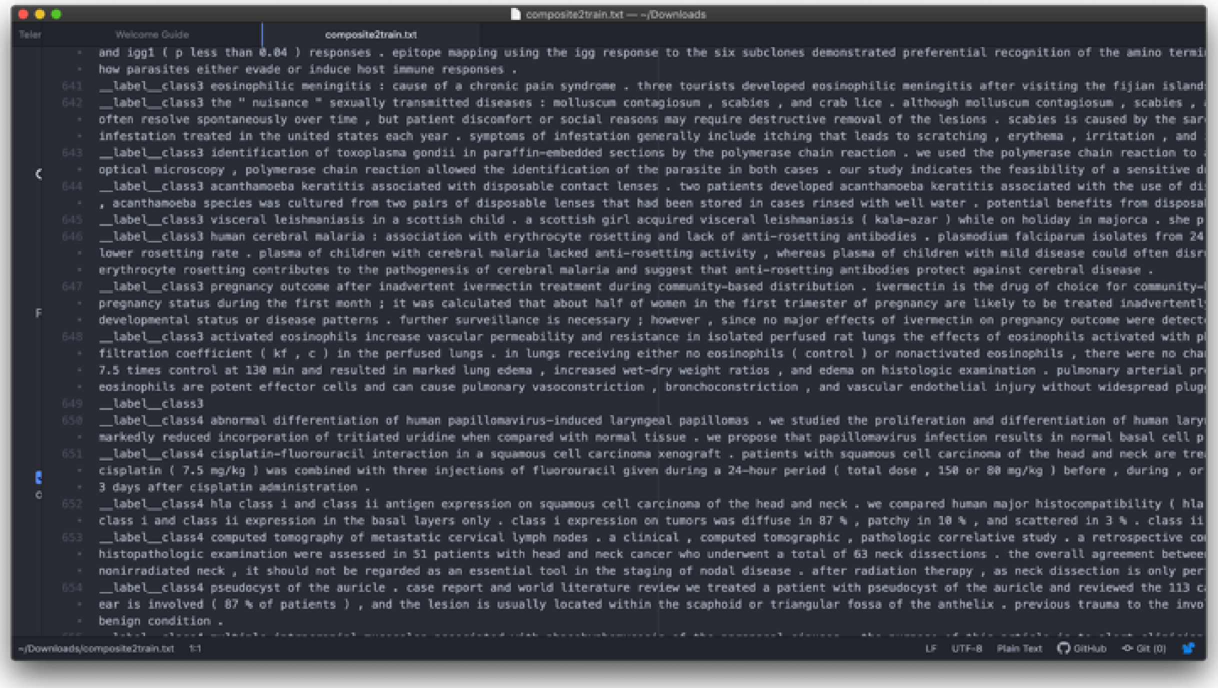
```
__label__node.js __label__linux __label__ubuntu __label__npm Node.js easy instal
lation in Linux
__label__java __label__android __label__google-pay __label__bluesnap Implementin
g Google Pay in Android application using BlueSnap
__label__vba __label__ms-access __label__access-vba Passing a string variable in
 SQL query
__label__c++ __label__recursion How to write this recursion with loops
__label__.net __label__visual-studio __label__magento2 __label__soap-client Add
Web Reference in Visual Studio with Header
__label__arrays __label__mongodb __label__mongoose __label__axios Updating an ar
ray in mongodb using mongoose and axios
__label__python __label__python-3.x Debugging in Python, mixed outputs
__label__react-native __label__android-studio Problems with loading app to andro
id Studio in react native
```

Figure 2: Head function output for stack overflow questions data

4.3. Ohsumed Data

The Ohsumed Dataset is an online medical database consisting of numerous articles. In this project, we used a dataset of Ohsumed articles from a novel project's GitHub repository []. The training and validation datasets were provided exclusively. They were readily classified into 23 categories separated in text files inside 23 different folders.

With the aid of the Python programming language, all these data were put together in a single composite file along with the __label__ tag for each observation. The __label__ tag is vital or else the fastText program will fail to read the categories. This dataset was run with all the models as mentioned in table 1. Figure 3 shows a small snippet of the training dataset to better understand how the categories are labelled.



```
and igg1 ( p less than 0.04 ) responses . epitope mapping using the igg response to the six subclones demonstrated preferential recognition of the amino termi
how parasites either evade or induce host immune responses .
__label__class3 eosinophilic meningitis : cause of a chronic pain syndrome . three tourists developed eosinophilic meningitis after visiting the fijian island
__label__class3 the " nuisance " sexually transmitted diseases : molluscum contagiosum , scabies , and crab lice . although molluscum contagiosum , scabies ,
often resolve spontaneously over time , but patient discomfort or social reasons may require destructive removal of the lesions . scabies is caused by the sar
infestation treated in the united states each year . symptoms of infestation generally include itching that leads to scratching , erythema , irritation , and
__label__class3 identification of toxoplasma gondii in paraffin-embedded sections by the polymerase chain reaction . we used the polymerase chain reaction to
optical microscopy , polymerase chain reaction allowed the identification of the parasite in both cases . our study indicates the feasibility of a sensitive d
__label__class3 acanthamoeba keratitis associated with disposable contact lenses . two patients developed acanthamoeba keratitis associated with the use of di
, acanthamoeba species was cultured from two pairs of disposable lenses that had been stored in cases rinsed with well water . potential benefits from disposa
__label__class3 visceral leishmaniasis in a scottish child . a scottish girl acquired visceral leishmaniasis ( kala-azar ) while on holiday in majorca . she p
__label__class3 human cerebral malaria : association with erythrocyte rosetting and lack of anti-rosetting antibodies . plasmodium falciparum isolates from 24
lower rosetting rate . plasma of children with cerebral malaria lacked anti-rosetting activity , whereas plasma of children with mild disease could often disr
erythrocyte rosetting contributes to the pathogenesis of cerebral malaria and suggest that anti-rosetting antibodies protect against cerebral disease .
__label__class3 pregnancy outcome after inadvertent ivermectin treatment during community-based distribution . ivermectin is the drug of choice for community-
pregnancy status during the first month ; it was calculated that about half of women in the first trimester of pregnancy are likely to be treated inadvertentl
developmental status or disease patterns . further surveillance is necessary ; however , since no major effects of ivermectin on pregnancy outcome were detect
__label__class3 activated eosinophils increase vascular permeability and resistance in isolated perfused rat lungs the effects of eosinophils activated with p
filtration coefficient ( kf , c ) in the perfused lungs . in lungs receiving either no eosinophils ( control ) or nonactivated eosinophils , there were no cha
7.5 times control at 130 min and resulted in marked lung edema , increased wet-dry weight ratios , and edema on histologic examination . pulmonary arterial pr
eosinophils are potent effector cells and can cause pulmonary vasoconstriction , bronchoconstriction , and vascular endothelial injury without widespread plug
__label__class3
__label__class4 abnormal differentiation of human papillomavirus-induced laryngeal papillomas . we studied the proliferation and differentiation of human lary
markedly reduced incorporation of tritiated uridine when compared with normal tissue . we propose that papillomavirus infection results in normal basal cell p
__label__class4 cisplatin-fluorouracil interaction in a squamous cell carcinoma xenograft . patients with squamous cell carcinoma of the head and neck are tre
cisplatin ( 7.5 mg/kg ) was combined with three injections of fluorouracil given during a 24-hour period ( total dose , 150 or 80 mg/kg ) before , during , or
3 days after cisplatin administration .
__label__class4 hla class i and class ii antigen expression on squamous cell carcinoma of the head and neck . we compared human major histocompatibility ( hla
class i and class ii expression in the basal layers only . class i expression on tumors was diffuse in 87 % , patchy in 10 % , and scattered in 3 % . class ii
__label__class4 computed tomography of metastatic cervical lymph nodes . a clinical , computed tomographic , pathologic correlative study . a retrospective co
histopathologic examination were assessed in 51 patients with head and neck cancer who underwent a total of 63 neck dissections . the overall agreement betwee
nonirradiated neck , it should not be regarded as an essential tool in the staging of nodal disease . after radiation therapy , as neck dissection is only per
__label__class4 pseudocyst of the auricle . case report and world literature review we treated a patient with pseudocyst of the auricle and reviewed the 113 c
ear is involved ( 87 % of patients ) , and the lesion is usually located within the scaphoid or triangular fossa of the anthelix . previous trauma to the invo
benign condition .
```

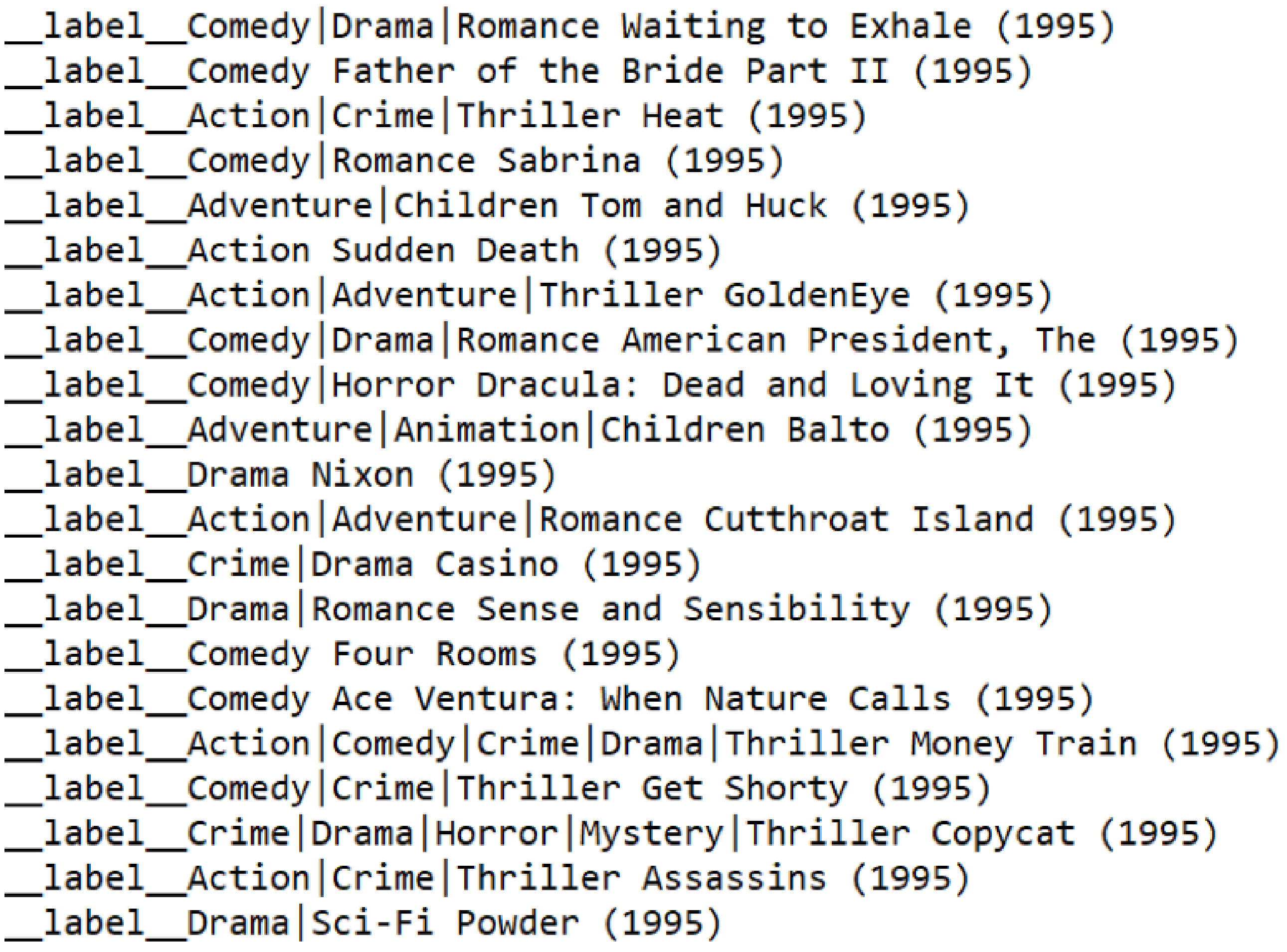
composite2train.txt — ~/Downloads
Teler
Welcome Guide
composite2train.txt
~/Downloads/composite2train.txt 1:1
LF UTF-8 Plain Text GitHub Git (0)

Figure 3: Example of the Ohsumed dataset

4.4. Online movies review Dataset

The fourth dataset came from a movies review website. The dataset had 9742 observations with 771 categories, the dataset comprised of movies from different genres, some movies were multi-label movies. The dataset was split into 2000 Train Data and 200 Test Data, "__label__" prefix was added to each category and the corresponding headline was appended to the string. This dataset was run with all the models as mentioned in table

1. Figure 4 bellow shows a snippet of the dataset with the appended labels.



```
__label__Comedy|Drama|Romance Waiting to Exhale (1995)
__label__Comedy Father of the Bride Part II (1995)
__label__Action|Crime|Thriller Heat (1995)
__label__Comedy|Romance Sabrina (1995)
__label__Adventure|Children Tom and Huck (1995)
__label__Action Sudden Death (1995)
__label__Action|Adventure|Thriller GoldenEye (1995)
__label__Comedy|Drama|Romance American President, The (1995)
__label__Comedy|Horror Dracula: Dead and Loving It (1995)
__label__Adventure|Animation|Children Balto (1995)
__label__Drama Nixon (1995)
__label__Action|Adventure|Romance Cutthroat Island (1995)
__label__Crime|Drama Casino (1995)
__label__Drama|Romance Sense and Sensibility (1995)
__label__Comedy Four Rooms (1995)
__label__Comedy Ace Ventura: When Nature Calls (1995)
__label__Action|Comedy|Crime|Drama|Thriller Money Train (1995)
__label__Comedy|Crime|Thriller Get Shorty (1995)
__label__Crime|Drama|Horror|Mystery|Thriller Copycat (1995)
__label__Action|Crime|Thriller Assassins (1995)
__label__Drama|Sci-Fi Powder (1995)
```

Figure 3: Example of movie review data

5. Results on New Data

5.1. IMDB movie titles and stack overflow questions data

Model	Precision1	Recall1	Precision2	Recall2
Simple supervised model	0.732	0.257	0.14	0.053
Cleaned data	0.732	0.257	0.18	0.0677
Epoch = 25	0.76	0.267	0.402	0.152
Learning rate = 1.0	0.748	0.263	0.461	0.174
Hierarchical softmax with word bigrams	0.564	0.198	0.155	0.0585
Multi label model softmax with word bigrams	0.688	0.242	0.32	0.121
Combined model with lr = 0.5	0.804	0.282	0.212	0.0798

Table 2 : precision and recall of all the models trained and validated with imdb movie data and stack overflow questions data (Precision1 and Recall1 for IMDB data, Precision2 and Recall2 for stack overflow data)

The recall and precision of each model is used as the evaluation technique for all the datasets. The original data gives 0.605 precision on a model with all the above discussed techniques combined. From the precision and recall it is evident that by tweaking the supervised text categorizing model with few added functionalities the results are improved drastically. The IMDB movie data performs better than the default data and gives precision as high as 80%, one of the reasons can be the small length of text, that is movie titles have less word count compared to cooking questions, also the total number of labels is very less compared to the original data and the number of observation are also less compared to the original data. Cleaning the IMDB data does not give any difference in the results as it does not have significant number of punctuation to begin with. The stack overflow questions do not perform well with the fastText model. The poor results can be due to the high number of labels in this data. Applying bigrams to the models does not help in improving the models here. All the models give best results with all the techniques combined in one model.

5.2. Ohsumed Data

As mentioned earlier, the training and validation datasets were provided separately. There was a total of 10664 observations for training and 12938 observations for validation. Table 3 shows the results obtained from creating the model with Ohsumed data compared to the Original data.

Model	Precision	Recall
Learning rate = 1.0	0.17	0.17
Learning rate = 0.5	0.0989	0.495

Table 3 : precision and recall of all the models trained and validated with Ohsumed data

5.3. Movies Review Data

Using the same information as in the construction of new data, the results of the movies review.

Model	Precision	Recall
Learning rate = 1.0	0.732	0.257
Learning rate = 5.0	0.0626	0.313

Table 4 : precision and recall of all the models trained and validated with kaggle movie data

6. Conclusion

The fastText model seems to perform better with data containing more observations and less total number of labels. By cleaning the data and increasing the number of epochs and increasing the learning rate the results can be significantly improved.

In this project, we have successfully been able to perform natural language processing using fastText. We have analyzed the classification performances by trying out new datasets from different sources. For future, natural language preprocessing techniques such as tokenization, removing stop words and lower casing may be used to achieve better classification results. This methodology can be further utilized with datasets such as social media feeds and its polarity as class.

References

[1]"Text classification · fastText", Fasttext.cc, 2019. Available: https://fasttext.cc/docs/en/supervised-tutorial.html.

[2]"Seasoned Advice", Seasoned Advice. Available: https://cooking.stackexchange.com/.

[3]"AbuTalha777/Natural-Language-Processing-Using-fastText", GitHub, 2019. Available: https://github.com/AbuTalha7 Language-Processing-Using-fastText.

[4]"StackAPI: A Python wrapper for the Stack Exchange API — StackAPI 0.1.12 documentation", Stackapi.readthedocs.io, 2019. Available: https://stackapi.readthedocs.io/en/latest/.

[5]"IMDbPY", PyPI, 2019. Available: https://pypi.org/project/IMDbPY/.

[6]"IMDB 5000 Movie Dataset", Kaggle.com, 2019. Available: https://www.kaggle.com/carolzhangdc/imdb-5000-movie-dataset.

[7]"MovieLens Latest Datasets", GroupLens, 2019. Available: https://grouplens.org/datasets/movielens/latest/.