



School of
Computing Science

Coursework of Text of Data

Junwen Yang, 2746688y

School of Computing Science

University of Glasgow

G12 8QQ

Degree of Master of Science at the University of Glasgow

Catalogue

Part 1 Dataset	3
Glance of Data.....	3
Q & A (Data Preparation)	3
Part 2 Clustering	6
Vector and Method Preparation.....	6
K-means Implementation.....	6
Analysis and Evaluation (Q & A)	7
Part 3 Comparing Classifiers	14
Vector Preparation.....	14
Training Models.....	14
Evaluation of the five methods.....	14
RandomForest Classifier	17
Part 4 Parameter Tuning	19
Tuning Process with ParameterGrid.....	19
Evaluation and Analysis.....	19
Part 5 Context Vectors using BERT	21
Encoding text && LogisticRegression Classification	21
Trainer of HuggingFace.....	21
Tuning Process.....	22
Analysis and Evaluation.....	23
Part 6 Conclusion and Future Work	24
Part 7 Research Paper Report	28

In this coursework, there are totally seven sections. A dataset of dialogs in various movies is used to do clustering, classification and more. The last part is a review of the selected paper. The tool name convokit designed by Cornell group is used to operate dataset in the beginning.

Part 1 Dataset

Glance of Data

Using the tool, the data can be shown as pandas dataframes:

```
corpus.get_utterances_dataframe().head()
```

	timestamp	text	speaker	reply_to	conversation_id	meta.movie_id	meta.parsed	vectors
id								
L1045	None	They do not!	u0	L1044	L1044	m0	[{'rt': 1, 'toks': [{'tok': 'They', 'tag': 'PR...	[]
L1044	None	They do to!	u2	None	L1044	m0	[{'rt': 1, 'toks': [{'tok': 'They', 'tag': 'PR...	[]
L985	None	I hope so.	u0	L984	L984	m0	[{'rt': 1, 'toks': [{'tok': 'I', 'tag': 'PRP',...	[]
L984	None	She okay?	u2	None	L984	m0	[{'rt': 1, 'toks': [{'tok': 'She', 'tag': 'PRP...	[]
L925	None	Let's go.	u0	L924	L924	m0	[{'rt': 0, 'toks': [{'tok': 'Let', 'tag': 'VB'...	[]


```
corpus.get_conversations_dataframe().head()
```

	vectors	meta.movie_idx	meta.movie_name	meta.release_year	meta.rating	meta.votes	meta.genre
id							
L1044	[]	m0	10 things i hate about you	1999	6.90	62847	['comedy', 'romance']
L984	[]	m0	10 things i hate about you	1999	6.90	62847	['comedy', 'romance']
L924	[]	m0	10 things i hate about you	1999	6.90	62847	['comedy', 'romance']
L870	[]	m0	10 things i hate about you	1999	6.90	62847	['comedy', 'romance']
L866	[]	m0	10 things i hate about you	1999	6.90	62847	['comedy', 'romance']

Conversations with genre are the data objects that will be used.

Q & A (Data Preparation)

(a) *What is your dataset, and why did you select it? How might automatic classification/labelling of your dataset be used in practice? [2 marks]*

The dataset will be used in this coursework is 'Cornell Movie-Dialogs Corpus', a large collection of fictional conversations extracted from raw movie scripts. The size of this dataset can be seen:

```
corpus.print_summary_stats()
```

Number of Speakers: 9035

Number of Utterances: 304713

Number of Conversations: 83097

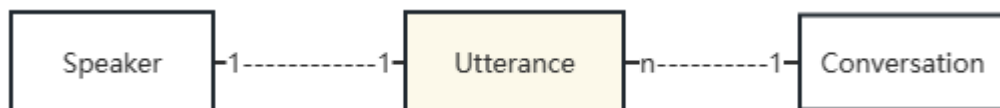
Since the conversation objects as samples that will be used in the trial are over

10000, 7000 conversation objects will be selected randomly.

```
conver_data = []
for i in range(6000):
    convs = corpus.random_conversation()
    conver_data.append(convs)
print(len(conver_data))
```

6000

This dataset was selected because of good structure of data, like documents can be conversations which are the combinations of utterance and there is a well-designed tool to process data, like using `get_id` function to search utterances of a conversation, plus, my interest. In this dataset, sentences and information of dialog were collected as three objects: Speakers, Utterances, Conversations. Each kind of objects maintains a column named 'meta-data' (shown in dataframes above). The relationship among three objects is like this:



The idea is to classify genre of conversations using utterances and genre

(b) Provide a summary of the labels and input text to be used. What labels are to be predicted? Did you need to do any preprocessing to create the labels or to make sure the number of labels was > 3 and <= 10? What is the text that will be used for classification? [4 marks]

For each conversation the genre is a string with shape of list, in the trial a multiclass classification will conduct, so the processes below extract the first item of the genre list, just assuming the first one is the most correlated and suitable one.

There are totally 24 genres and only the ten most frequent ones are selected: **['drama', 'thriller', 'comedy', 'action', 'romance', 'crime', 'sci-fi', 'adventure', 'mystery', 'horror']** (respective count:[('drama', 3212), ('thriller', 2543), ('comedy', 1725), ('action', 1574), ('crime', 1457), ('romance', 1434), ('sci-fi', 1067), ('adventure', 1065), ('mystery', 1047), ('horror', 811)]) And the text that will be used to classify are utterances being included in a conversation, they are combined as a string text relating to one conversation's label. Conversations with empty genre values are removed.

```
length of label-list: 5453
length of texts-list: 5453
one label example: drama
one text example: She looks thirty-five, forty. She didn't look Italian to me.
What, Ma?
She's more than twenty-nine years old, Marty. That's what she tells you.
```

(c) Is the dataset already split into a training, validation and test set? If not, use a 60/20/20% split to create the training, validation and test set.

Provide a table with the label counts for each split of the dataset and comment on the distribution across labels and across data splits. [2 marks]

```
from sklearn.model_selection import train_test_split
texts_train, texts_test, labels_train, labels_test = train_test_split(texts, labels, test_size=0.2, random_state=42)
texts_train, texts_val, labels_train, labels_val = train_test_split(texts_train, labels_train, test_size=0.25, random_state=42)
```

The result of splitting the dataset is the table below:

	Training set	Validation set	Testing set
Texts	3271	1091	1091
Labels	3271	1091	1091

Part 2 Clustering

In this part, the conversations stored in texts list will be used to do clustering. First, tokenization is conducted with spacy tools to remove stop words, whitespace and punctuation, also, to calculate similarity (the replacement of distance for k-means algorithm) between two conversations, lemma is used as the root of words. And then sparse TF-IDF vectors will be obtained from these lemmas.

Vector and Method Preparation

To calculate the sparse TF-IDF vectors, vocabulary of this corpus is needed:

K-means Implementation

- Step 1: Pick k random "centroids"
- Step 2: Assign each vector to its closest centroid
- Step 3: Recalculate the centroids based on the closest vectors
- [Repeat Steps 2 and 3 until the model converges]

There methodology of K-means algorithm is implemented by codes below:

```
def calculate_centroid(centroids, current_clusters, k):
    new_centroids = []
    # for cluster in current_clusters:
    for i in range(k):
        similarities = {}
        for data_point in current_clusters[i]:
            similarities[current_clusters[i].index(data_point)] = sparse_cosine_similarity(centroids[i], data_point)
        average = sum(similarities.values()) / len(similarities)
        a = min(similarities.keys(), key=lambda x: (similarities[x] - average))
        new = current_clusters[i][a]
        new_centroids.append(new)
    return new_centroids

def assign_data_points(data_points, centroids):
    #here we use similarity between data_points and centroids
    assigned_clusters = [ [] for _ in range(len(centroids)) ]
    for item in data_points:
        max_sim = -math.inf
        center_index = 0
        for cent in centroids:
            similarity = sparse_cosine_similarity(item, cent)
            if similarity > max_sim:
                max_sim = similarity
```

```

        center_index = centroids.index(cent)
        #通过上面的 for 循环找到了 similarity 最大的, 以及对应的 center
        assigned_clusters[center_index].append(item)
    return assigned_clusters

def isConvergence(last_cen, cen, k):
    change = []
    for i in range(k):
        dis = sparse_cosine_similarity(last_cen[i], cen[i])
        change.append(dis)
    mean = sum(change) / k
    if mean < 2E-3:
        return True
    else:
        return False

max_run = 2000
k = 5
data_points = nor_vec
last_centroids = None
centroids = []
result = None
# cur_assigned_clusters = []
for i in range(max_run):
    if len(centroids) == 0:
        centroids = random.sample(data_points, k)
    #assignment of data_points
    last_centroids = centroids
    cur_assigned_clusters = assign_data_points(data_points, centroids)
    #calculate new centeroids
    centroids = calculate_centroid(centroids, cur_assigned_clusters, k)
    if isConvergence(last_centroids, centroids, k):
        result = cur_assigned_clusters
    elif i - max_run == 1:
        result = cur_assigned_clusters

print("length of five clusters: ", len(result[0]), len(result[1]), len(result[2]), len(result[3]), len(result[4]))
length of five clusters: 1222 1210 743 781 1548

```

Analysis and Evaluation (Q & A)

(a) When using $k=5$ clusters, give a few examples of the documents assigned to each cluster, and the top 5 tokens with the highest magnitude in the corresponding centroid. [2 marks]

Some examples of clustering result in each cluster:

=====Four conversations of cluster 1 =====

I don't know.

You see that's what bothers me. No other bottle of nasal spray was found in the house. The police looked. There was only the one bottle. But you say you didn't arrive until after Mr. Marsh was dead -- yet we know he was using the nasal spray prior to his death. How do you think it got there?

P.M.

A.M -- or P.M.?

Three fifteen.

Would you read for us the time of the purchase?

Dristan nasal spray.

There's an item you picked up that's marked. Will you read it?

Yes.

Well - I'm a little confused. This is a charge receipt from Rosen's Drug Store where Mr. Marsh had an account. It's dated the day of the murder.

Is this your signature?

Yes.

Gulp!

Yes.

Are they following us?

They are with me!

Who are they? Lieutenant, get those people away from there.

Really? Maybe I can get you on my screen and see you at last!

...Not far, now.

=====Four conversations of cluster 2 =====

I don't think so.

That you come with me.

What do you suggest?

Don't go home. And don't go to work. Either one could be bad.

I'm going to find him. Because he'd find me.

No idea. Honest. What are you going to do?

Where do you think Jerry is?

Jonas builds assassins for a living. Several of whom may be in place already. We'd like to kill a few birds with one stone.

He's shown himself. Why haven't you arrested him or killed him or done whatever it is you do?

He's why I watch Jerry. Jerry's the bait for Jonas.

And Jonas?

I'm, it really doesn't matter. Think C.I.A. and exponentiate. I'm a government employee and I've been watching Jerry for awhile.

I'm the artist. Like my mother.
You the architect?

Please.

Ice?

Fine.

Jack Daniel's okay? It's gonna have to be.

I need the money to smooth my way, you understand? Now, have you got that sort of cash here or do we need to meet in the morning?

Excuse me?

Your brother got mixed up with child procurers and tried to make this world a better place, Mrs De Moraes. And having rescued one little life he unwisely set out to repeat the exercise. You dont mess around with child procurers. Right now my guess is hes either on the run, held captive, or dead. I understand your misgivings, Mrs De Moraes. But Ive seen the boy and made telephone contact with the man Leon bought him from. If anyone knows what happened to your brother it will be that man. Which leads me to why Im here at such a late hour. I need 20,000, in cash, by 11 this morning.

What... What are you talking about?

You seem surprised. Could it be you dont think that badly of him after all? You neednt worry. It seems his motives were pure. From what I can make out he bought the boy to rescue him from further abuse.

=====Four conversations of cluster 3 =====

Who are you?

No. Don't ever want to go out without telling us.

Which is it?

NOOO!

Then he dies. Right now.

No...

I wanna play a game.

No.

Did you ever tie him up?

Exactly what do you have in mind, Mr. Corrigan.

Did you ever engage in sado- masochistic activity with him?

I'll request the scientists from Pacific-Tech to monitor the drop. We'll clear the area all around. After that we'll hit them all over the world. I'll have long-range bombers alerted, loaded and standing by. Then our first target will be the initial landing place outside Los Angeles.

There's only one thing that will stop the Martians! We've held back previously because of the danger of radiation to civilians. Now there's no choice. The United Nations has voted authority to the United States. The White House will confirm an order to use the Atom bomb.

=====Four conversations of cluster 4=====

Every time I said it, it was. I never really thought I was going to make it.

It wasn't a lie.

Every day for the last five years, I told myself someday I would be out here again. No more bars. No more guards. No more fights just to stay alive. Every day for the last five years, I told myself that lie.

I say you should get a shrink.

We should get ourselves a lawyer.

Not my head, buddy. Not me. I'm gettin' a headache just listenin' to you. Maybe that's the only way to get through. Besides, six heads'll be better than one.

Come on, Professor. The army's not gonna give you any answers. You'll be buttin' your head against a stone wall.

I heard stories. I don't think you wanna know.

I wish they'd tell us what they're going to do with us.

I sure wish Superman was around. He wouldn't let any of this go on. Not for one minute.

I heard stories. I don't think you wanna know.

I wish they'd tell us what they're going to do with us.

I sure wish Superman was around. He wouldn't let any of this go on. Not for one minute.

=====Four conversations of cluster 5=====

We talked.

What?... What'd you do?

He came by.

He called you?

He told me last night.

How'd you find that out?

It's a lot of money. About a half-a- million dollars. All of it in Cabo in safe deposit boxes and more comin' in.

Well, if the A.T.F. guy is the one who wants you, that'll only interest him up to a point.

I want to talk to them first. I know more now about Ordell's money.

What I meant was have a lawyer do the negotiating for you.

I don't know yet. I'm going to talk with Dargus and Nicolet today. Do what you suggested. Offer to help and see what happens.

Are you?

I called in sick this morning. As far as the airline knows, I'm still available.

It's not strictly legal.

...Well, if the pay's right and it's legal I'll do it.

Got a job for you.

...That's the test, ain't it? Test of true love--

Sure, Lana, I'd love to.


I can't Clark already asked me. Didn't you? He used to love this song.

Didn't you? So he just said -- "Would you dance with me?"

Where you been, Harry?

Jesus, if I have a heart attack, I hope you know what to do.

5 tokens with the highest magnitude in the corresponding centroid:

 =====top 5 tokens estimated by TF in centroid 1 =====
lieutenant, people, away,
=====top 5 tokens estimated by TF in centroid 2 =====
force, get, drug, trouble, got,
=====top 5 tokens estimated by TF in centroid 3 =====
think, come, oughta, kind, warning,
=====top 5 tokens estimated by TF in centroid 4 =====
try, store, sense, think, close,
=====top 5 tokens estimated by TF in centroid 5 =====
know oh goodbye lion right

(b) Based on (a), do the clusters make sense? Are there certain topics that appear in some but not others? [2 marks]

Actually, the result of clustering doesn't make sense. It can be seen in the conversations of one cluster that there is no pattern. There neither any obvious topic among conversations in one cluster, but it seems that in the second cluster, conversations are a little bit related to jobs or careers.

(c) Construct a confusion matrix between the $k=5$ clusters and your target labels.

Since only label the five cluster with one, two...five have none sense, the most representative label of each cluster was found by calculating the most common label of the assigned data point in each cluster, but the result show that they are all 'action'.

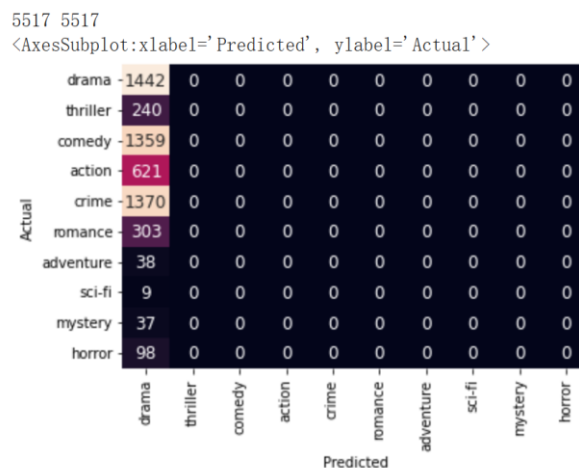
```
[('action', 332)], [('action', 318)], [('action', 200)], [('action', 197)],
[('action', 408)]]
```

```
def setClusterLabel(label_indexes):
    all_labels = []
    for i in label_indexes:
        all_labels.append(cluster_labels[i])
    count = Counter(all_labels).most_common(1)

    return count

[ ] search_index = []
cluster_result_labels = []
centroids_index = []
for i in range(5):
    list = []
    centroids_index.append(nor_vec.index(centroids[i]))
    for item in result[i]:
        index = nor_vec.index(item)
        list.append(index)
    search_index.append(list)
for x in search_index:
    cluster_result_labels.append(setClusterLabel(x))
```

With the setting that all clusters correlate to label 'action', the result of confusion matrix is shown below:



(d) What trends, if any, do you notice from the confusion matrix? Does it look like some clusters are able to pick up on a single label? When a cluster includes multiple labels, are they related? [2 marks]

Because of the operation in (c), the confusion matrix doesn't work well. But with exploration in five clusters, it is found that the most frequent labels of data point assigned in each clusters are the same (comedy, drama, action, crime):

```
[('comedy', 647), ('drama', 614), ('action', 592), ('crime', 298)],
[('action', 97), ('drama', 92), ('comedy', 80), ('crime', 38)], [('drama',
300), ('action', 271), ('comedy', 267), ('crime', 112)], [('comedy', 157),
('drama', 141), ('action', 135), ('crime', 74)], [('comedy', 210), ('drama',
198), ('action', 190), ('crime', 107)]]
```

And list below shows the distribution of all labels in each cluster:

```
[Counter({'comedy': 647, 'drama': 614, 'action': 592, 'crime': 298, 'horror': 147, 'adventure': 112, 'thriller': 58, 'mystery': 27, 'sci-fi': 17, 'romance': 16}),
Counter({'action': 97, 'drama': 92, 'comedy': 80, 'crime': 38, 'adventure': 15, 'horror': 13, 'thriller': 13, 'romance': 2, 'mystery': 1, 'sci-fi': 1}),
Counter({'drama': 300, 'action': 271, 'comedy': 267, 'crime': 112, 'horror': 64, 'adventure': 44, 'thriller': 18, 'romance': 10, 'sci-fi': 9, 'mystery': 7}),
Counter({'comedy': 157, 'drama': 141, 'action': 135, 'crime': 74, 'horror': 32, 'adventure': 25, 'thriller': 16, 'romance': 5, 'mystery': 4, 'sci-fi': 2}),
Counter({'comedy': 210, 'drama': 198, 'action': 190, 'crime': 107, 'horror': 50, 'adventure': 33, 'thriller': 14, 'mystery': 8, 'sci-fi': 7, 'romance': 3})]
```

The reason for the highlight clustered result may be caused by the imbalance of data set, which have much more drama, action and comedy conversations than others.

Part 3 Comparing Classifiers

Simple introduction of models

logistic regression: a widely used classifier that can deal with dependencies between features and incorporate arbitrary features easily, but it will overfit on very sparse data like the one-hot vectors in this trial.

SVC: a model with a selected kernel which reflects data to higher dimension and then aims to find the largest margin among groups with a separation. It is effective in high dimensional spaces even there are more dimensions than the samples. It is designed for binary question, so the multiclass classification is handled in one-vs-one scheme.

Vector Preparation

The detail tools and pipelines of this process can be seen in the same section of .ipynb file.

In the process of making one-hot vectors, because the training set and validation set have different texts, leading to different vocabulary list, in order to have the same features, the tokens only in training vocabulary list are ignored.

Training Models

The detail of training process are in the same section of .ipynb file.

Evaluation of the five methods

Accuracy, precision, recall and f1 scores are used to evaluate the performance of the models trained in this section.

The heatmap below shows the four scores both in validation set and training set of five models:

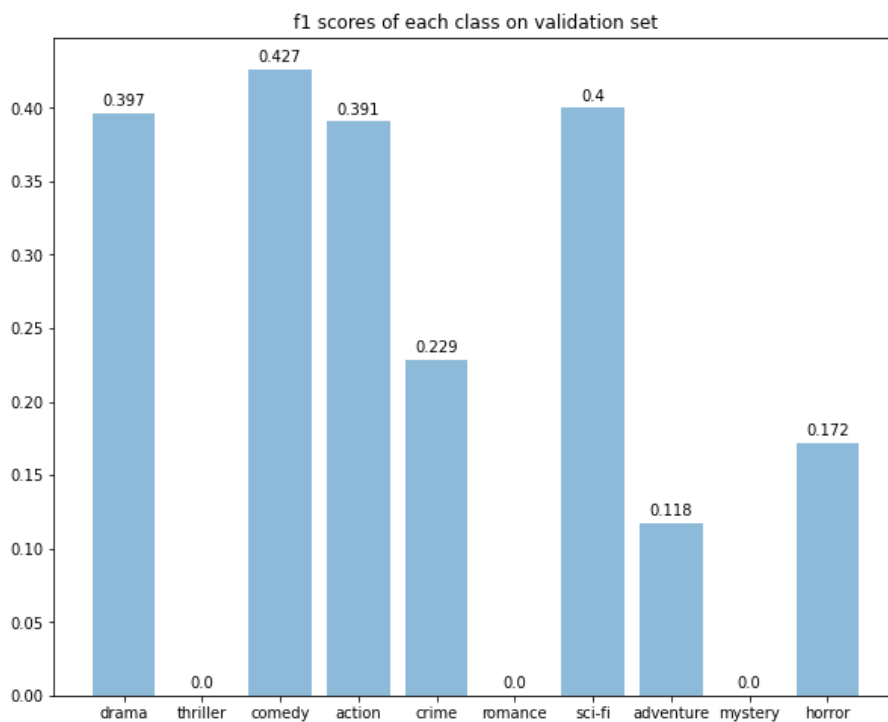
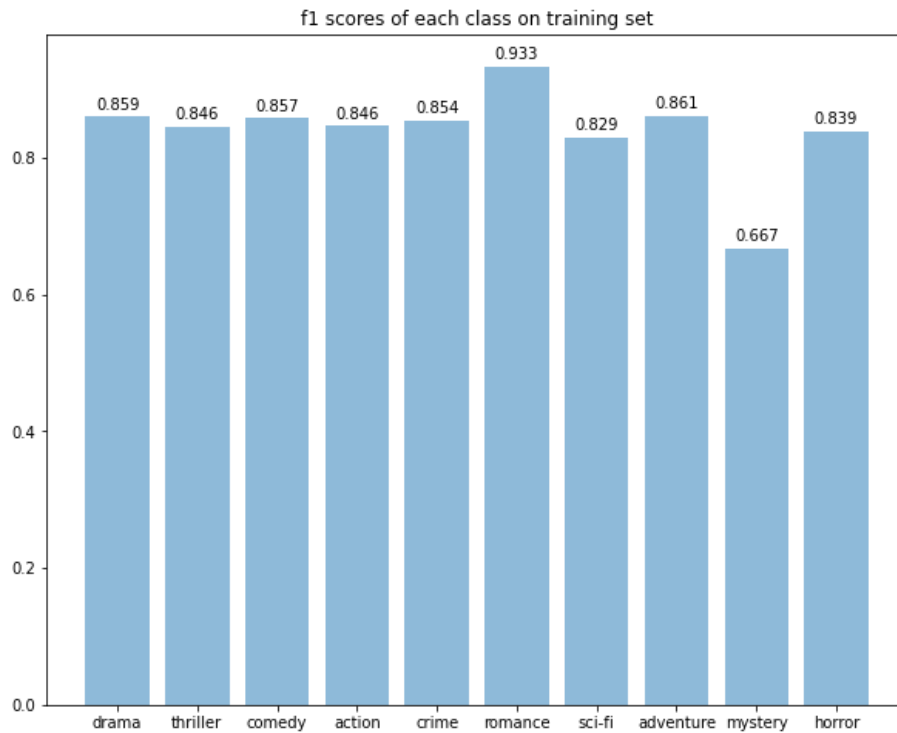


Using dummy results as baselines, firstly discussing the situation on the training set (LRC1, LRC2, SVC correlating to LogisticRegression with One-hot vectorization, LogisticRegression with TF-IDF vectorization, SVC respectively), it shows that all of them performed well by taking accuracy into consideration that they all classified over half of the conversations into the right genres, LCR1 did especially well in this perspective. When it comes to precision, LCR1 and SVC both achieved the goal that most of the conversations assigned to one genre are truly tagged with this genre, whereas LRC2 only did a little bit less than half of that. And for the recall score, only LRC1 took over 70% that conversations in one genre are correctly classified, neither LCR2 nor SVC just achieved nearly 30%. This is the same situation on f1 score. And all this trend occurred in validation set but with much lower scores in four measurements, which may be related to overfitting problem.

More specifically, accuracy is a kind of global measurement on all data which may be useless on unbalanced dataset, so precision and recall which explore the situation in each classification are more useful. Based on this, it can be seen from the figure above that for three models, overfitting occurred in LRC1 and SVC to some extent, since the scores of precision, recall and f1 on the training set displayed higher than those on the validation set, especially on model LRC1. For model LRC1, there are over 4000 features, which may be one reason for its definitely overfitting. However, on LRC2, both precision and recall scores of training set and validation set are under 50%, implying the underfitting of this model.

In the section on the exploration of the dataset, to get a more balanced dataset, only the 10 most frequent genres were maintained, but showing their counts: ('drama', 3240), ('thriller', 2522), ('comedy', 1776), ('action', 1535), ('crime', 1479), ('romance', 1395), ('mystery', 1086), ('sci-fi', 1078), ('adventure', 1066), ('horror', 836), there are over 3000 conversations in the most genre class which is three or four times more than some of the others, so it cannot be treated as a balanced dataset. And after vectorization, it can be seen that an apparent noise occurred: like 'didn't', '--name' and ' ', they are stored as didn, t , didn't, --name, indicating an optimizing requirement on tokenization and vectorization.

Summarizing, the best performance came from LogisticRegression with One-hot vectorization, f1-scores on each class are shown in bar charts.



RandomForest Classifier

RandomForest is an integration of many separated decision trees (bagging method). When a sample is inputted, the class of it is decided by the count of classification result made by trees in this forest. And each inner node of the decision tree is a decision based on attribution. After going through the decided branches, the leaf node is the decision result. Here selected the parameter `n_estimators=500` as the number of trees in this forest.



```
=====on training set=====on validation set
accuracy_gnbt=          1.0 accuracy_gnbv=      0.373
precision_gnbt=          1.0 precision_gnbv=     0.575
recall_gnbt=           1.0 recall_gnbv=       0.184
f1_gnbt=              1.0 f1_gnbv=          0.195
```

Using spacy to do tokenization, the result shows that on the training set, all scores are 1 higher than the scores of five other models, indicating this model can classify all conversations correctly into their genres. But on the validation set, the scores are much lower than those in training set even if the precision score is over 50%. This may be due to overfitting.

Part 4 Parameter Tuning

Tuning Process with ParameterGrid

The tool of sklearn named ParameterGrid is used to find a nice combination of parameters which can improve the performance of model LogisticRegression with Tfidf vectors. Parameter Grid is set as below:

```
param_grid = {  
    "class_weight" : [None, "balanced"],  
    'C': [0.01, 0.1, 1, 10, 100],  
    'max_df' : [0.6, 0.7, 0.8, 0.9],  
    "sublinear_tf" : [True, False],  
    "max_features": [None, 1000, 3000, 5000, 6000, 7000]}
```

And the result is:

```
➞ best_params={'C': 100, 'class_weight': 'balanced', 'max_df': 0.7, 'max_features': None, 'sublinear_tf': False}  
best_f1=0.263
```

Evaluation and Analysis

With the ParameterGrid, here are the parameters being tuned:

'C': [0.01, 0.1, 1, 10, 100]

'class_weight': [None, 'balanced']

"max_df": [0.6, 0.7, 0.8, 0.9]

"sublinear_tf": [True, False]

"max_features": [None, 1000, 3000, 5000, 6000, 7000]

C and class_weight are the parameters of LogisticRegression: C is the parameter representing the strength of regularization, the smaller it is, the stronger the L2 (regularization selected) is. Class_weight is the parameter to control the weight of each class. If not given, all classes are supposed to have weight one. The "balanced" mode uses the values of y to automatically adjust weights inversely proportional to class frequencies, which is useful to unbalanced dataset.

"max_features", "max_df" and "sublinear_tf" are parameters of TfidfVectorizer. Sublinear_tf is the scaling of applying tf sublinear. Term frequency will be changed with form of $1 + \log(\text{tf})$. And max_df is useful when there is no stop-word removal process, terms that have a document frequency strictly higher than the given threshold (*corpus-specific stop words*) are ignored. If float in range [0.0, 1.0], the parameter represents a proportion of documents, integer absolute counts. And "max_features" is defaulted as None, meaning all tokens in vocabulary are used, if not none, a vocabulary will be built with max_features ordered by tf.

For TfidfVectorizer, using `max_df = 0.7` may ignore some words in stop-words list with high document frequency. For LogisticRegression model, `C` is 10, higher than the default value 1, meaning the strength of regularization has decreased, and `class_weight` is set as balanced, due to the unbalanced class of genres. Training with the parameters given by Tuning result, the LogisticRegression finally accesses the performance below:

☞	accuracy	precision	recall	f1
LRC2 TRAIN	0.648	0.493	0.306	0.299
Tuned_LRC2 TRAIN	0.959	0.964	0.982	0.973
LRC2 VALIDATION	0.341	0.123	0.14	0.123
Tuned_LRC2 VALIDATION	0.342	0.323	0.229	0.250

As for the result, it can be seen that after tuning, using this best combination of parameters, the performance got the best result on training set all over 0.9, also an improvement is on validation set, especially precision increased nearly 20% which is obvious compared to the 10% increase of other three measurement scores. But the result illustrates that the problem of overfitting still exists.

Part 5 Context Vectors using BERT

In this section, an exploration will be conducted on a deep learning-based approach, to see whether it can improve the performance compared to the earlier more traditional approaches

Encoding text && LogisticRegression Classification

The pipeline of the RobertaTokenizer is used to encoding text.

```
4 nlp_features = pipeline('feature-extraction', model="roberta-base")
```

Training set and validation set are processed separately, with the parameter truncation and max_length set with the length limitation of Roberta model

```
[ ] bert_vecs_val = []

for text in texts_val:
    output = nlp_features(text, truncation=True, max_length=514)
    output = np.array(output)
    vec = output[0]
    bert_vecs_val.append(vec[0]) # store the vecs
```

And only the first vector is selected to comprise the matrix to train a LogisticRegression classification model. The result on validation is shown in the left picture with accuracy, precision, recall and f1 scores.



accuracy=0.389
precision=0.242
recall=0.164
f1=0.148



accuracy=0.426
precision=0.265
recall=0.175
f1=0.158

Since it is even worse than the normal methods, the evaluation matrix of training set is computed showing in the right picture. It can be seen it is not because of overfitting, it seems that the model hasn't been trained entirely with underfitting problem.

Trainer of HuggingFace

Some process like making label string to integer and transferring data into Dataset object are conducted.

Based on the given notebook and HuggingFace instruction of text classification, the tokenizer and model are imported and loaded:

```
from transformers import AutoModelForSequenceClassification
from transformers import AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained("distilbert-base-uncased")
model = AutoModelForSequenceClassification.from_pretrained("roberta-
base", num_labels=10, id2label=id2label, label2id=label2id)
```

Also, for evaluation, the module “evaluation” is imported and loaded the four evaluation score which are inserted into the computing matrix:

```
import evaluate
accuracy = evaluate.load("accuracy")
f1 = evaluate.load("f1")
precision = evaluate.load("precision")
recall = evaluate.load("recall")

import numpy as np
def compute_metrics(eval_pred):
    predictions, labels = eval_pred
    predictions = np.argmax(predictions, axis=1)
    accuracy_score = accuracy.compute(predictions=predictions, references=
labels)
    precision_score = precision.compute(predictions=predictions, reference
s=labels, average="macro")
    recall_score = recall.compute(predictions=predictions, references=labe
ls, average="macro")
    f1_score = f1.compute(predictions=predictions, references=labels, avera
ge="macro")
    result_metrix = {"accuracy_score": accuracy_score['accuracy'],
                    "precision_score": precision_score['precision'],
                    "recall_score": recall_score['recall'],
                    "f1_score": f1_score['f1']}
    return result_metrix
```

Using “trainer” of HuggingFace, after setting some of the training arguments below:

```
learning_rate = 1e-4
batch_size = 16
epochs = 1 # This is
```

The model is trained and the result shows in the table:

Loss	Accuracy	Precision	Recall	F1
1.777101	0.269406	0.026940	0.1	0.042446

Extremely bad, the main reason is supposed to be the lack of training data, because it is a 10-class classification task using neural network.

The complete codes can be seen in the same section of this part of .ipynb file.

Tuning Process

Codes are not shown here, due to the device limitation, it is a manual process. Here is a part of the result on the best set of three hyperparameters given by trainer:

Epoch	Training Loss	Validation Loss	Accuracy Score	Precision Score	Recall Score	F1 Score
1	No log	2.314281	0.085243	0.033028	0.104530	0.028411
2	2.337900	2.149974	0.246563	0.063335	0.100749	0.046165
3	2.173000	1.894027	0.246563	0.076020	0.100244	0.059923
4	1.909700	1.843886	0.252979	0.082876	0.103036	0.066043
5	1.848400	1.825598	0.255729	0.083719	0.104321	0.069686
6	1.848400	1.815032	0.252979	0.078255	0.103094	0.072206
7	1.818400	1.808667	0.245646	0.074501	0.100278	0.072294
8	1.804800	1.804289	0.245646	0.074367	0.100125	0.072888
9	1.806800	1.800732	0.246563	0.074210	0.100423	0.075789
10	1.802600	1.798096	0.255729	0.078432	0.103968	0.079556

Analysis and Evaluation

Three hyperparameters were selected to tune the model: learning-rate, batch-size and epoch. In case of lacking of computing capacity and training time, batch size was set between 8 and 16, there are totally three sets of combination with respective result:

	Learning rate	Batch size	Epoch
Set1	1e-6	8	10
Set2	1e-6	16	30
Set3	1e-7	8	20

	Loss	Accuracy	Precision	Recall	F1
Set1	1.781956	0.263927	0.051369	0.098977	0.061419
Set2	1.826101	0.259361	0.078472	0.100843	0.080318
Set3	1.789955	0.254812	0.076659	0.103085	0.079310

It can be seen that the comprehensively best performance occurred in set3 among these three. The loss in this parameter set finally accessed 1.790 with f1 score at 0.079, lower than the loss of 1.834 on set2 and the f1 score of 0.061 on set1. And for the trend, the loss keeps decreasing and accuracy, f1 score keep increasing implying with higher epoch, loss can get convergence with other evaluation score rising. But the problem is that in total experiment of BERT, the performance is extreme poor, I guess it is because of the amount of data, 3000 data points are not adequate to train a deep learning neural network.

The reason can be the nice combination of learning rate and batch size. Learning rate is the step size to do gradient decrease in deep learning and batch size the amount of data trained each time, whereas the larger batch size is, the fewer times each epoch is updated, where a bigger learning rate can be more suitable.

**In this part, the tuning result were lost. I bought a colab plus in another google account and shared the notebook because the origin account have already run out of the computing units. The problem is that without privacy to save changes and accidentally close of colab, all code-running-result are lost.*

Part 6 Conclusion and Future Work

(a) Take the best approach from the prior questions (which should be trained on the training set) and evaluate it with the test set. Report the evaluation metrics. Provide a single confusion matrix of the classifications on the test set. [2 marks]

Overall, the best model is the tuned LogisticRegression model in Part 4.

The prediction result on test dataset is below:

Accuracy	Precision	Recall	F1
0.313	0.338	0.196	0.209

The confusion matrix of the classifications on the test set is:

Actual	drama	119	2	52	55	29	0	11	0	0	14
	thriller	2	2	0	3	1	0	1	0	1	0
	comedy	70	3	99	48	30	0	11	0	0	12
	action	65	2	51	111	26	0	9	0	2	12
	crime	33	1	28	16	30	0	4	0	0	6
	romance	2	1	1	1	0	1	0	0	0	0
	adventure	13	3	12	13	5	0	8	0	0	2
	sci-fi	2	0	1	2	2	0	0	0	0	1
	mystery	2	0	0	2	2	0	0	0	0	1
	horror	10	1	15	19	5	0	3	0	0	7
		drama	thriller	comedy	action	crime	romance	adventure	sci-fi	mystery	horror
		Predicted									

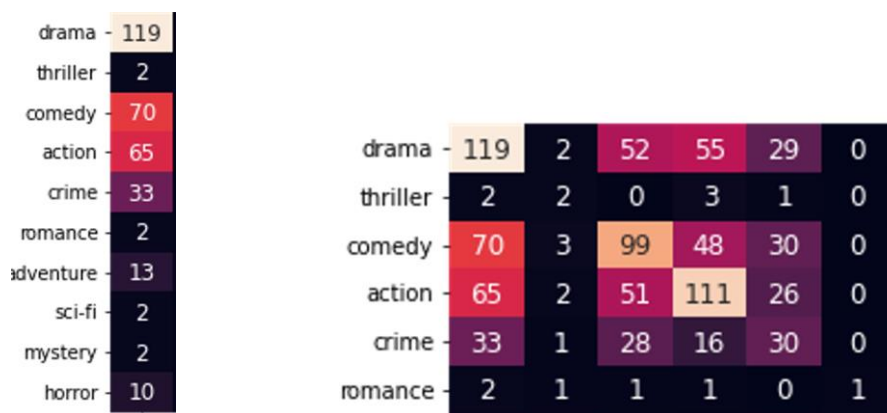
(b) Manually examine the predictions on the test set. Analyze the results for patterns and trends using the confusion matrix. Hypothesize why common classification errors are made. Report on your error analysis process and summarize your findings. Be sure to give specific examples of the types of errors you observe. [5 marks]

	True count	Predicted count
Drama	282	318
Thriller	10	15
Comedy	273	259

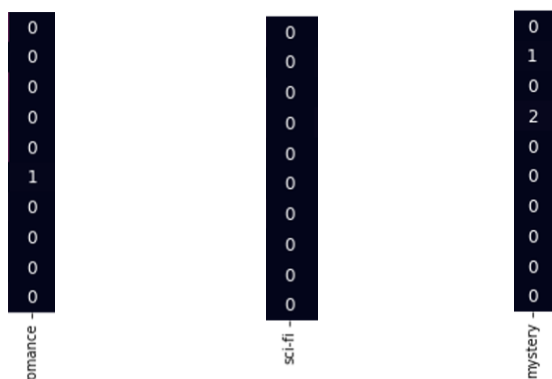
Action	278	270
Crime	118	130
Romance	6	1
Adventure	56	47
Sci-fi	8	1
Mystery	7	3
Horror	60	55

In the table above, the first column is the sum of each row representing the true number of conversations in this genre, and the second column is the sum of each column representing the conversation classified into this genre.

In the testing set, only the class with a number of objects (the count of true data points) like “drama”, “comedy”, “action” can achieve a better classification result than the others with few objects, “better” here means there are more conversations being classified as “True Positive” than the ones incorrectly classified into other genres (there are only these three diagonal values which are higher than the value in the same column). And on the side of “classification error”, the probability that a conversation is predicted as genres with high frequency is much higher than others, just like the part of matrix showing below, drama conversations are more likely to be classified as drama, comedy, action and crime. This is the same trend of other classes.



And from confusion matrix, it can be seen that conversation is hardly being predicted as romance, sci-fi, mystery.



The reason why error occurred may be complex. The test data is not balanced,

especially in class “romance”, “sci-fi”, “mystery”, and this is the same problem in training set, without down-sampling process which can due to the bad classification result. And since there are 10 classes, the result of classification indicates more data should be taken to train the model.

However, multilabel classification should work better in labeling movie genres, since a movie is allocated to several type in daily life, and “triller” and “horror”, “crime” and “sci-fi”, these are too similar to classify by texts even for human.

(c) Discuss whether the final performance is sufficiently high for the stated purpose of this classifier. What impact would false positives and false negatives have on the deployment of this machine learning pipeline? Refer to the confusion matrix from (a) to support your claims. [3 marks]

The final performance is poor. False positives and false negatives may be big problems if this system deploys. For example, in the first column of matrix, except the first element means these are truly drama conversations, the others are all false positives, and if the classification is used, people might find the expecting drama dialogs are crime ones even can be thriller. And some of the drama conversations are labeled as comedy or action, this kind of false negatives may be confused to users.

(d) Could the deployment of this system have any negative societal effects? [2 marks]

Since the poor result showing in test dataset, If the system deployments, there will be a problem that the movie may be classified into a wrong genre. Imagining a case that a young child would like to enjoy a comedy by selecting a movie in the genre group but actually played a horror one, which would be a mental injury. There may be a lot of effects like this arousing by this kind of mis-classification. But if it is a well-performed classification model, it can be a good tool.

(e) Propose further steps that could be taken to improve the classification effectiveness of the system. [2 marks]

To improve the performance, three approaches should be addressed:

1. Enlarge the size of training set, it can be seen that even in the regular machine learning method, overfitting occurred, since the dataset is split with sklearn tool, the feature distribution between training set and validation set should be the same, and it is a classification with 10 labels, it is supposed that the size of training set should be enlarge. And this will also suit for the neural network training.
2. The model should be multilabel one since it is more common in daily life that a movie is typed with several genres. And labels taken into

classification should be more isolate, not like horror and thriller.

3. The tokenization process should be improved. Exploring the tokenization result of spacy and self-set pipeline, there are some potential noises like didn't --> [didn, 't], and <>sentence, --sentence --> ['—sentence'], the model might use the feature of noise to make classification. A better and more accurate sentence process pipeline should be designed.
4. More computing units are needed. BERT cannot get good performance without high GPUs; even only fine-tuning approach runs with pre-trained models.
5. A more comprehensive parameter tuning approach should be taken.

Part 7 Research Paper Report

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding:

<https://aclanthology.org/N19-1423.pdf>

Background and context

Pre-training of a number of unlabeled texts can improve performance of many language models effectively by tuning obtained language representations to adjust specific tasks, just as asking a native speaker knowing grammars and semantics to writing a story rather than an English-learner. After the exploration in context-free techniques, context-aware techniques were introduced like ELMo and PGT, which are two approaches applied with feature-based and fine-tuning methodologies. But limitation here is that only unidirectional language models can be used like left-to-right self-attention layers in PGT, context on the right of a token is not introduced. To take context into consideration completely, models like Bi-LSTM using two LSTM models processing input sequences forwardly and reversely and concatenate results together, but without making Bi-Transformer, BERT applying ‘Bi-direction’ in another way.

Contributions

A new language representation model is introduced in this paper, which improves the state-of-the-art performance on eleven NLP tasks especially natural understanding systems. Based on the strategy of fine-tuning, there are two steps to generate a deep bidirectional representation: pre-training and fine-tuning. And instead of using left-to-right architecture like GPT or the countenance of two unidirectional models, in the first step BERT using Masks LM to implement considering bi-direction of context. In the second step, in order to adjust to downstream tasks, the pre-trained parameters are used as initial input and train by correlated labeled data for specific task with self-attention mechanism of Transformer, which is the reason why BERT gives unified model to various tasks. And using GPU, BERT is also a time-saving model especially in step two.

Critique

In order to prove BERT's promotion of NLP process, the author conducted a series of experiments. In the first experiment, five groups of models completed the tasks provided by CLUE respectively, among which two models had different parameter configurations of BERT, and one of them had the same parameter configuration as PGT in the group. This serves as a nice contrast

showing how BERT overperform others because before BERT, the best model was PGT. For the evaluation of results, the experiment uses a standard systematic test evaluator, among which QQP and MRPC is an indicator based on f1 and Spearman is an indicator based on STS, which makes the comparison of results more convincing. But there is a problem in this part that for the instability on small datasets of BERT Large configuration group, the author only randomly restarted to solve it, but didn't go further to give the reason and the scale of small dataset that causes instability of model. And in the second experiment of Question and Answering and SWAG, human performance was recorded, the more the model accesses to human, the more successful it is, so this is a brilliant comparison to show the power of BERT. Moreover, a bidirectional model BiLSTM was tested to show that using MLM technique, BERT is the best even comparing to feature strategy and other bidirectional implementation.

Another positive point is that effectiveness on different hyperparameters were tested, providing a good guidance for user to tune their own models.

Writing quality

The report is organized in a good structure, from normal abstract, introduction, method, experiment and conclusion. In the beginning, the introduction of BERT also detailed as input/output representations, two steps and techniques in each step, giving a comprehensive illustration of what is BERT. And charts and figures are used respectively in experiment and theory section respectively, increasing readability.