# Modelling

```
from octis.evaluation_metrics.diversity_metrics import
WordEmbeddingsInvertedRBOCentroid, TopicDiversity
from octis.evaluation_metrics.coherence_metrics import Coherence,
WECoherenceCentroid
from octis.models.model import load_model_output, save_model_output
from octis.preprocessing.preprocessing import Preprocessing
from octis.optimization.optimizer import Optimizer
from octis.optimization.optimizer_tool import
plot_bayesian_optimization, convergence_res
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from skopt.space import Integer, Real, Categorical
from octis.dataset.dataset import Dataset
from gensim.models import Word2Vec
from IPython.display import Image
from octis.models.LDA import LDA
from octis.models.CTM import CTM
from octis.models.LSI import LSI
from octis.models.ETM import ETM
from gensim.models import Word2Vec
from gensim.models import Word2Vec
import matplotlib.pyplot as plt
from collections import Counter
from joblib import load, dump
import pandas as pd
import numpy as np
import gensim
import string
import spacy
import nltk
import time
import os
import re
```

Three different variants of OCTIS models will be optimized and trained: The standard Latent Dirichlet Allocation (LDA) in its Gensim Implementation, a Contextualized Topic Model (CTM) relying heavily on the use of contextualized transformer models like Bert, as well as a Topic Model with which draws its topics and words from the embedding space of the original coprus (ETM) using word2vec.

```
word2vec_model = Word2Vec.load("./word2vec/word2vec_model")
df_ref=pd.read_csv('./preprocessing/results/df_books_prep.csv')
df_input=pd.read_csv('./preprocessing/results/df_books_chunk.csv')

#df_input['tokenized_text'] =
df_input['preprocessed_text'].apply(lambda text: text.split())
```

In order to improve the salience of the results Mimno et all have argued for the importance of filtering a corpus to remove the most freuqent, as well as the most rare terms in order to make for more meaningful results. As threshhods they have proposed removing the top 5 to 10 percent of most frequent terms, as well as removing everyingth that doens't appear at least 5-10 times in the whole corpus.

```
# # Create a frequency distribution
# word_freq = Counter(word for doc in df_input['tokenized_text'] for
word in doc)

# # Identify words to filter
# total_words = len(word_freq)
# top_5_percent = int(total_words * 0.05)
# most_common_words = set(word for word, freq in
word_freq.most_common(top_5_percent))
# least_frequent_words = set(word for word, freq in word_freq.items()
if freq < 5)

# # Combine the sets of words to remove
# words_to_remove = most_common_words.union(least_frequent_words)

# # Filter words from each document
# df_input['filtered_tokens'] =
df_input['tokenized_text'] .apply(lambda doc: [word for word in doc if
word not in words_to_remove])

# # Optionally, rejoin tokens into a single string
# df_input['filtered_text'] = df_input['filtered_tokens'].apply('
'.join)
```

We start with transforming the corpus in the necessary input format required for the octis model topic modeling implementations.

```
# Writing 'filtered_text' column to 'corpus.txt'
with open('./octis/corpus.txt', 'w', encoding='utf-8') as corpus_file:
    for preprocessed_text in df_input['filtered_text']:
        corpus_file.write(preprocessed_text + '\n')

#creating an input vocabulary file

vocabulary = set()

for text in df_input['filtered_text']:
    vocabulary.update(text.split())

with open('./octis/vocabulary.txt', 'w') as vocab_file:
    for word in sorted(vocabulary):
        vocab_file.write(word + '\n')
```

```
#octis expects a binary, not a model object
#word2vec_model.wv.save_word2vec_format('./word2vec/word2vec_model.bin
', binary=True)
```

Latent Dirichlet Allocation, LDA is yet another transformation from bag-of-words counts into a topic space of lower dimensionality. LDA is a probabilistic extension of LSA (also called multinomial PCA), so LDA's topics can be interpreted as probability distributions over words. These distributions are, just like with LSA, inferred automatically from a training corpus. Documents are in turn interpreted as a (soft) mixture of these topics (again, just like with LSA).

The variant of the dataset for model training and the variant for final later use only differ by the former carrying a split into training, test and validation set.

```
# preprocessor = Preprocessing(vocabulary=None,
save_original_indexes=True, max_features=None,
#                               remove_punctuation=True,
punctuation=string.punctuation,
#                               lemmatize=False,
stopword_list='english',
#                               min_chars=1, min_words_docs=0)
# # preprocess
# dataset = Dataset()
# dataset =
preprocessor.preprocess_dataset(documents_path=r'/Storage/Studium/
DigitalHumanities/Semester5/Thesis/code_notebooks/octis/corpus.txt')

# # # #save the preprocessed dataset

# dump(dataset, './octis/dataset.joblib')
dataset=load('./octis/dataset.joblib')

# preprocessor = Preprocessing(vocabulary=None,
save_original_indexes=True, max_features=None,
#                               remove_punctuation=True,
punctuation=string.punctuation,
#                               lemmatize=False,
stopword_list='english',split=False,
#                               min_chars=1, min_words_docs=0)
# # preprocess
# dataset_final = Dataset()
# dataset_final =
preprocessor.preprocess_dataset(documents_path=r'/Storage/Studium/
DigitalHumanities/Semester5/Thesis/code_notebooks/octis/corpus.txt')

# dump(dataset_final, './octis/dataset_final.joblib')
dataset_final=load('./octis/dataset_final.joblib')
```

# Model Training

## LDA

OCTIS - intrinsic optimization pipeline with skopt. All three models are optimized on the basis of on Word Embedding Coherence Centroid, a normalized metric which draws from an embedding space drawn from the original corpus in order to allow for a more domain specific and closer fit to the texts in use. In addition, as a secondary metric Diversity Centroid was chosen to influence the choice of model parameters towards a broader coverage of content inherent in the texts. The optimization method chosen by OCTIS by default is bayesian, which is a resource intensive stochastic approach well suited for otherwise difficult to optimize or evaluate models, which are taken as a black-box input.

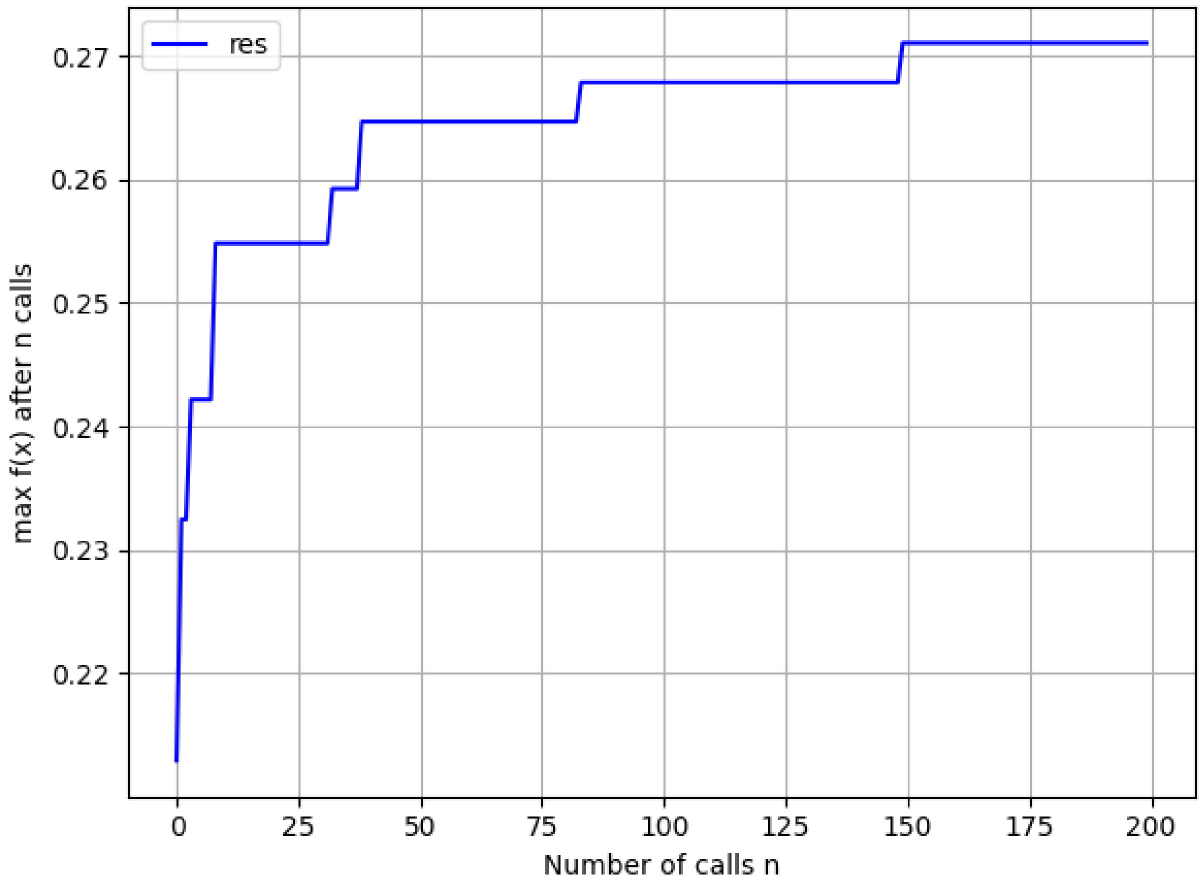
```
# Define the hyperparameter space
parameter_space = {
    'num_topics': Integer(50, 101),  # Range of topics as integer
    'alpha': Real(0.001, 10, prior='uniform'),  # Dirichlet
hyperparameter for document-topic distribution
    'eta': Real(0.001, 10, prior='uniform')  # Dirichlet
hyperparameter for topic-word distribution
}

# Define the model
lda_model = LDA(num_topics=60, alpha=0.09, eta=5.0)

optimization_runs=200
model_runs=3

#npmi = Coherence(texts=dataset.get_corpus())
coherence_centroid = WECoherenceCentroid(topk=20,
word2vec_path='./word2vec/word2vec_model.bin', binary=True)
#topic_diversity = TopicDiversity(topk=10)
DiversityCentroid=WordEmbeddingsInvertedRBOCentroid(topk=20,
weight=0.9, normalize=True,
word2vec_path='./word2vec/word2vec_model.bin', binary=True)
#coherece_score = metric_coherence.score(dataset.get_corpus())
# Run the hyperparameter optimization
start = time.time()

optimizer=Optimizer()
optimizer_results_LDA = optimizer.optimize(model=lda_model,
dataset=dataset, metric=coherence_centroid,
                        search_space=parameter_space,
number_of_call=optimization_runs,
                        n_random_starts=5,surrogate_model='RF',
                        model_runs=model_runs, save_models=True,
topk=20,
extra_metrics=[DiversityCentroid],save_path='./octis/results/',
plot_best_seen=True)

end = time.time()
duration = end - start

# Save the results
optimizer_results_LDA.save_to_csv("./octis/results/Opt_LDAresults.csv"
)

print('Optimizing model took: ' + str(round(duration)) + ' seconds.')

dump(optimizer_results_LDA,
'./octis/models/optimizer_results_LDA_long.joblib', compress=('lzma',
9))
```

```
['./octis/models/optimizer_results_LDA_long.joblib']
```

```
optimizer_results_LDA=load('./octis/models/
optimizer_results_LDA_long.joblib')

# the optimization process here is one of maximization, yet the
provided function for plotting
# assumes the oposite, so we shall invert the result.

values_to_plot_LDA = [-x for x in optimizer_results_LDA.info['f_val']]

plot_bayesian_optimization(values=values_to_plot_LDA,
                           name_plot="LDA_Optimization_Convergence",
                           log_scale=False,
                           conv_max=True)
```

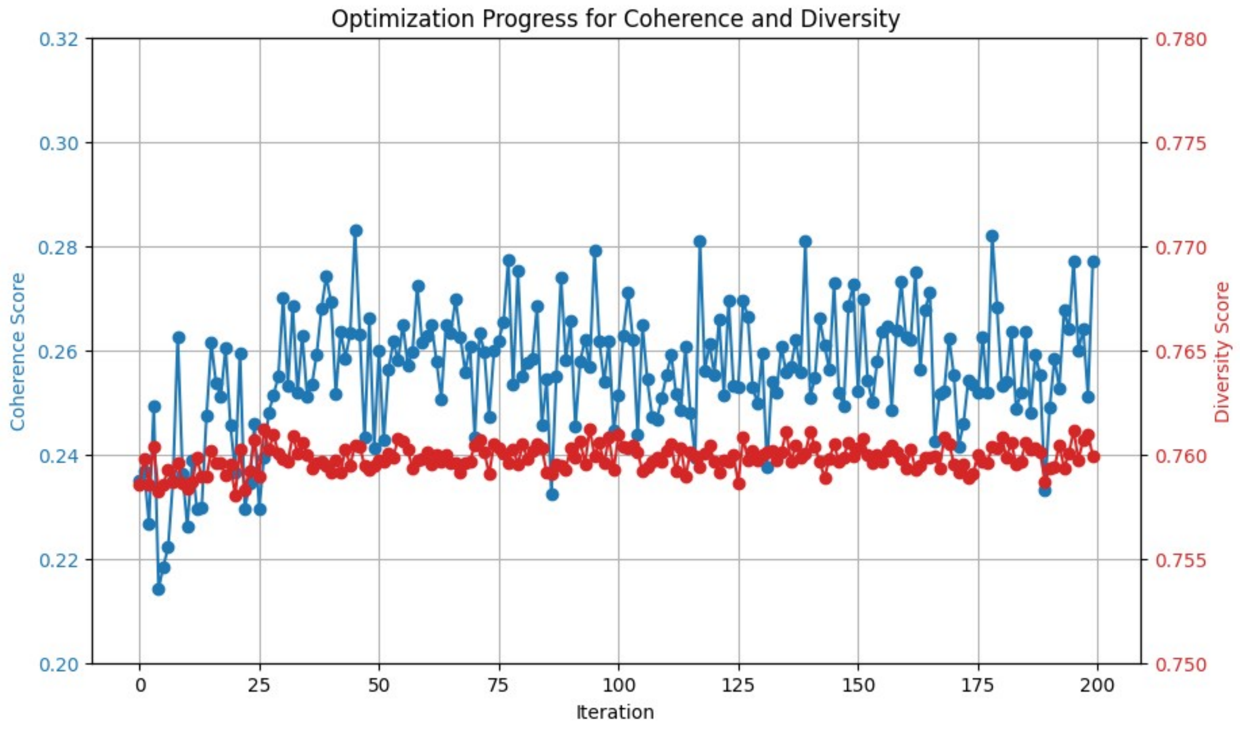
The optimization was successful

```
#The image is just dumped into the working directory, we shall move it
somewhere more fitting
os.replace('./LDA_Optimization_Convergence.png',
'./octis/results/LDA_Optimization_Convergence.png')

Image(filename='./octis/results/LDA_Optimization_Convergence.png')
```



```python
# Plotting WECoherenceCentroid' and
'WordEmbeddingsInvertedRBOCentroid' our two metrics
coherence_scores_LDA = [max(run) for run in
optimizer_results_LDA.info['dict_model_runs']
['WECoherenceCentroid'].values()]
diversity_scores_LDA = [max(run) for run in
optimizer_results_LDA.info['dict_model_runs']
['0_WordEmbeddingsInvertedRBOCentroid'].values()]

# Create a figure with twin y-axis for the second metric
fig, ax1 = plt.subplots(figsize=(10, 6))

ax1.set_xlabel('Iteration')
ax1.set_ylabel('Coherence Score', color='tab:blue')
ax1.plot(coherence_scores_LDA, marker='o', color='tab:blue')
ax1.tick_params(axis='y', labelcolor='tab:blue')
ax1.set_ylim(0.20, 0.32)

ax2 = ax1.twinx()
ax2.set_ylabel('Diversity Score', color='tab:red')
ax2.plot(diversity_scores_LDA, marker='o', color='tab:red')
ax2.tick_params(axis='y', labelcolor='tab:red')
```

```
ax2.set_ylim(0.75, 0.78)

plt.title('Optimization Progress for Coherence and Diversity')
ax1.grid(True)

plt.show()
```



for the choice of the best hyperparameters, we primarily take the coherence values into account, but use diversity as a form of secondary criterion. Of the ten best coherence scores we shall select the parameter set with the highest diversity score.

```
# Extract the best configurations based on coherence scores
coherence_scores_LDA = optimizer_results_LDA.info['dict_model_runs']
['WECoherenceCentroid']
diversity_scores_LDA = optimizer_results_LDA.info['dict_model_runs']
['0_WordEmbeddingsInvertedRBOCentroid']

# Extracting scores and hyperparameter configurations
combined_scores = []
for i, (coherence_run, diversity_run) in
enumerate(zip(coherence_scores_LDA.values(),
diversity_scores_LDA.values())):
    max_coherence = max(coherence_run)
    max_diversity = max(diversity_run)
    alpha = optimizer_results_LDA.info['x_iters']['alpha'][i]
    eta = optimizer_results_LDA.info['x_iters']['eta'][i]
    num_topics = optimizer_results_LDA.info['x_iters']['num_topics']
```

```
[i]
    combined_scores.append((max_coherence, max_diversity, alpha, eta,
num_topics))

# Sort by coherence score first and take the top ten
top_ten_by_coherence = sorted(combined_scores, key=lambda x: x[0],
reverse=True)[:10]

# Select the configuration with the highest diversity score from the
top ten
best_overall_LDA = max(top_ten_by_coherence, key=lambda x: x[1])

# Print the best configuration
print("Best Coherence Score:", best_overall_LDA[0])
print("Corresponding Diversity Score:", best_overall_LDA[1])
print("Best Configuration: num_topics =", best_overall_LDA[4], ",
alpha =", best_overall_LDA[2], ", eta =", best_overall_LDA[3])

Best Coherence Score: 0.27711903966818585
Corresponding Diversity Score: 0.7611531671767668
Best Configuration: num_topics = 73 , alpha = 0.06460320991688963 ,
eta = 2.9415645388839033

best_num_topics = best_overall_LDA[4]
best_alpha = best_overall_LDA[2]
best_eta = best_overall_LDA[3]

# Now instantiate the LDA model with the best hyperparameters
best_model_LDA = LDA(num_topics=best_num_topics,
                     alpha=best_alpha,
                     eta=best_eta)

# Disable partitioning to use the entire dataset
best_model_LDA.partitioning(use_partitions=False)

# Train the model with the dataset
best_model_LDA = best_model_LDA.train_model(dataset_final)

#dump(best_model_LDA,'./octis/models/best_model_LDA.joblib',
compress=('lzma', 9))
best_model_LDA = load('./octis/models/best_model_LDA.joblib')

#dump(best_model_LDA,'./octis/models/best_model_LDA_overall.joblib',
compress=('lzma', 9))
#best_model_LDA = load('./octis/models/best_model_LDA_overall.joblib')
```

The topic output for the optimal model

```
top_words_per_topic_LDA = best_model_LDA['topics']
```

```python
# Display the top words for each topic
for topic_id, words in enumerate(top_words_per_topic_LDA):
    print("Topic", topic_id + 1, ":", words)
```

```
Topic 1 : ['relation', 'sex', 'pressure', 'driver', 'delicacy',
'arranged', 'ruins', 'curtains', 'impatient', 'willing']
Topic 2 : ['plague', 'louder', 'motives', 'foreign', 'travel',
'examined', 'disappointment', 'prince', 'authority', 'prisoner']
Topic 3 : ['toll', 'winds', 'roar', 'tide', 'stage', 'lovers', 'er',
'sits', 'represented', 'wheels']
Topic 4 : ['milk', 'outline', 'fountain', 'staff', 'sunset', 'yonder',
'obtained', 'splendid', 'entire', 'fruit']
Topic 5 : ['abbot', 'assist', 'virtues', 'revenge', 'sons', 'labour',
'agitation', 'harm', 'delivered', 'amiable']
Topic 6 : ['abode', 'examined', 'obtained', 'gallery',
'disappointment', 'travel', 'ice', 'mile', 'arranged', 'that']
Topic 7 : ['beast', 'purple', 'lid', 'rare', 'camp', 'monster', 'the',
'marvellous', 'fashionable', 'merry']
Topic 8 : ['prophet', 'inspired', 'cavern', 'kings', 'ridiculous',
'yonder', 'crimson', 'wave', 'glare', 'lamps']
Topic 9 : ['the', 'quality', 'park', 'prince', 'guilt',
'disappointment', 'magnificent', 'entire', 'willing', 'rapid']
Topic 10 : ['kingdom', 'maiden', 'darling', 'willows', 'tent',
'angels', 'stronger', 'camp', 'older', 'reward']
Topic 11 : ['cigar', 'understanding', 'pleasures', 'severe',
'impatient', 'agitation', 'prisoner', 'chapel', 'housekeeper',
'willing']
Topic 12 : ['paw', 'earliest', 'major', 'silk', 'knock', 'harm',
'warning', 'nephew', 'ing', 'examined']
Topic 13 : ['the', 'paw', 'major', 'afterward', 'endeavour',
'sergeant', 'that', 'mistake', 'distinguish', 'fairy']
Topic 14 : ['er', 'lord', 'elder', 'cavern', 'staff', 'lightly',
'rival', 'terrific', 'faintly', 'milk']
Topic 15 : ['housekeeper', 'ocean', 'wave', 'prophet', 'flying',
'prisoner', 'clearly', 'noon', 'ice', 'date']
Topic 16 : ['lid', 'destined', 'contained', 'respectable', 'bushes',
'inside', 'mistake', 'flood', 'sisters', 'quality']
Topic 17 : ['prince', 'jealous', 'furnished', 'picked', 'chimney',
'savage', 'the', 'hint', 'italian', 'college']
Topic 18 : ['louder', 'honourable', 'cavern', 'rope', 'landlady',
'stillness', 'occasioned', 'bones', 'level', 'gods']
Topic 19 : ['abbot', 'yard', 'er', 'picked', 'echoed', 'crimson',
'lightly', 'visits', 'curtain', 'pattern']
Topic 20 : ['merry', 'captain', 'named', 'camp', 'wagon', 'showman',
'advance', 'bade', 'relations', 'earliest']
Topic 21 : ['pattern', 'gods', 'careful', 'ice', 'stir', 'rope',
'yard', 'queen', 'dressing', 'nice']
Topic 22 : ['conducted', 'endeavour', 'gallery', 'singer', 'absorbed',
'turns', 'prince', 'principal', 'officers', 'ceiling']
Topic 23 : ['student', 'honourable', 'paw', 'accents', 'earliest',
'merit', 'attendants', 'mischief', 'beard', 'disappointment']
Topic 24 : ['milk', 'traveller', 'relations', 'chimney', 'staff',
'nice', 'dogs', 'amiable', 'consent', 'pitcher']
Topic 25 : ['priests', 'honourable', 'foreign', 'system', 'temple',
'student', 'agitation', 'italian', 'revenge', 'attendants']
Topic 26 : ['abbot', 'monster', 'rope', 'obscurity', 'harbor', 'rat',
'beating', 'camp', 'severe', 'frequent']
Topic 27 : ['abbot', 'basket', 'exposed', 'yonder', 'sisters',
'bushes', 'afterward', 'amazement', 'cavern', 'burned']
Topic 28 : ['painter', 'galley', 'wounded', 'maiden', 'pardon',
'student', 'points', 'passionate', 'er', 'lord']
Topic 29 : ['dismal', 'milk', 'traveller', 'staff', 'hearth',
'stately', 'nice', 'marvellous', 'vampire', 'elder']
Topic 30 : ['enemies', 'milk', 'parent', 'abode', 'destruction',
'pattern', 'the', 'minister', 'prisoner', 'abbot']
Topic 31 : ['toll', 'verse', 'sex', 'forgive', 'remote', 'poem',
'maiden', 'warmth', 'torn', 'priests']
Topic 32 : ['guilty', 'louder', 'impatient', 'thrust', 'wounded',
'mistake', 'beating', 'agreed', 'endeavour', 'contempt']
Topic 33 : ['vampire', 'fruit', 'stately', 'suspected', 'muttered',
'recollect', 'ruined', 'astonished', 'hide', 'stricken']
Topic 34 : ['fountain', 'wedding', 'wave', 'hearth', 'heap', 'noon',
'breathed', 'housekeeper', 'flat', 'pressure']
Topic 35 : ['abbot', 'student', 'pattern', 'dwelling', 'blessed',
'onward', 'gallery', 'hearth', 'gates', 'advance']
Topic 36 : ['onward', 'poem', 'surrounding', 'yonder', 'glided',
'forwards', 'porch', 'verse', 'angels', 'numerous']
Topic 37 : ['tent', 'plague', 'bag', 'poem', 'lift', 'relations',
'wept', 'profound', 'trifle', 'sole']
Topic 38 : ['housekeeper', 'distinguish', 'wounded', 'endeavour',
'reflections', 'labour', 'folly', 'er', 'poverty', 'assist']
Topic 39 : ['prophet', 'lord', 'fruit', 'beating', 'prisoner',
'swiftly', 'agreed', 'dressing', 'fatigue', 'terrace']
Topic 40 : ['guilt', 'widow', 'dagger', 'poured', 'haired', 'reward',
'impatient', 'yard', 'assumed', 'mirth']
Topic 41 : ['merry', 'lord', 'the', 'that', 'abbot', 'funeral',
'fairy', 'poem', 'authority', 'smooth']
Topic 42 : ['milk', 'staff', 'chill', 'stir', 'torn', 'quality',
'bowed', 'pattern', 'knife', 'yonder']
Topic 43 : ['abbot', 'er', 'reputation', 'mistake', 'yield', 'guilt',
'maiden', 'fatigue', 'chimney', 'poured']
Topic 44 : ['louder', 'beating', 'lantern', 'groan', 'ray', 'bade',
'sexton', 'ocean', 'sum', 'cautiously']
Topic 45 : ['wave', 'dwelling', 'er', 'innocence', 'hearth', 'mode',
'elder', 'mummy', 'driver', 'ruined']
Topic 46 : ['knife', 'yes', 'stair', 'forgive', 'jumped', 'rode',
'didn', 'goblins', 'officers', 'rapid']
Topic 47 : ['monster', 'painter', 'sum', 'wave', 'captain',
'religion', 'employment', 'expectation', 'prince', 'rang']
```

```
Topic 48 : ['abbot', 'science', 'bade', 'ice', 'nobleman', 'deadly',
'guide', 'signs', 'ashamed', 'current']
Topic 49 : ['abbot', 'paw', 'major', 'sergeant', 'rat', 'rope',
'motionless', 'knock', 'starting', 'captain']
Topic 50 : ['sergeant', 'paw', 'examined', 'driver', 'afterward',
'plague', 'match', 'major', 'knock', 'chill']
Topic 51 : ['tent', 'bushes', 'beings', 'expectation', 'willows',
'flood', 'yes', 'revenge', 'canoe', 'behaviour']
Topic 52 : ['throne', 'th', 'st', 'hide', 'clergyman', 'wedding',
'yonder', 'roar', 'kings', 'guide']
Topic 53 : ['plague', 'artist', 'date', 'stage', 'abode',
'represented', 'absent', 'prophet', 'destined', 'military']
Topic 54 : ['the', 'beds', 'scarce', 'gun', 'candles', 'destruction',
'hath', 'warning', 'torn', 'queen']
Topic 55 : ['student', 'captain', 'hanging', 'accordingly', 'parent',
'foreign', 'coach', 'seeking', 'lord', 'examined']
Topic 56 : ['lid', 'dismal', 'lovers', 'housekeeper', 'slender',
'camp', 'enemies', 'prisoner', 'thrust', 'lamps']
Topic 57 : ['nice', 'abbot', 'carrying', 'didn', 'suspect', 'crying',
'fruit', 'prisoner', 'merry', 'monster']
Topic 58 : ['abbot', 'milk', 'staff', 'elder', 'sons', 'housekeeper',
'priests', 'marry', 'bones', 'th']
Topic 59 : ['er', 'nice', 'didn', 'stair', 'wave', 'monster', 'isn',
'yes', 'guide', 'that']
Topic 60 : ['milk', 'nice', 'traveller', 'tent', 'mischief',
'priests', 'fruit', 'prince', 'obtained', 'request']
Topic 61 : ['abbot', 'nice', 'sum', 'hath', 'afterward', 'mistake',
'earlier', 'heir', 'date', 'paw']
Topic 62 : ['relate', 'remote', 'bushes', 'dressing', 'furnished',
'monster', 'guilty', 'invited', 'reward', 'splendid']
Topic 63 : ['beating', 'enthusiasm', 'reputation', 'affections',
'guide', 'stage', 'stole', 'gods', 'louder', 'breathed']
Topic 64 : ['student', 'didn', 'lid', 'that', 'driver', 'yes', 'knot',
'heavily', 'goblins', 'naughty']
Topic 65 : ['haired', 'obscurity', 'guilty', 'monimia', 'degrees',
'frequent', 'remorse', 'heir', 'galley', 'trifling']
Topic 66 : ['poem', 'the', 'college', 'attachment', 'contained',
'older', 'principal', 'hearth', 'rare', 'misfortunes']
Topic 67 : ['pattern', 'purple', 'crying', 'coroner', 'clad',
'savage', 'cease', 'reputation', 'clearly', 'faintly']
Topic 68 : ['student', 'dwelling', 'enthusiasm', 'pointing',
'disappointment', 'rat', 'rope', 'marchioness', 'honourable',
'delicacy']
Topic 69 : ['wedding', 'hath', 'tent', 'ice', 'rope', 'minister',
'rat', 'energy', 'noon', 'bushes']
Topic 70 : ['earthly', 'fairy', 'dismal', 'lovers', 'gallery', 'didn',
'lightly', 'hide', 'burden', 'funeral']
Topic 71 : ['ocean', 'bones', 'hearth', 'funeral', 'beds', 'minister',
'fish', 'sexton', 'milk', 'beach']
```

```
Topic 72 : ['the', 'ice', 'mistake', 'afterward', 'serve', 'appeal',
'gates', 'chill', 'splendid', 'gift']
Topic 73 : ['the', 'mistake', 'dressing', 'understanding',
'housekeeper', 'angels', 'religion', 'chimney', 'hearth', 'relations']
```

```
dump(top_words_per_topic_LDA,'./analysis/
top_words_per_topic_LDA.joblib')
```

```
['./analysis/top_words_per_topic_LDA.joblib']
```

Preparing the results for further analysis

```
def get_document_topic_percentages(topic_document_matrix):

    num_docs = topic_document_matrix.shape[1]
    num_topics = topic_document_matrix.shape[0]
    data = []

    # Iterate over each document
    for doc_index in range(topic_document_matrix.shape[1]):
        # Get topic distribution for the document
        topic_distribution = topic_document_matrix[:, doc_index]

        data.append([round(percentage * 100, 2) for percentage in
topic_distribution])

    column_names = [f"Topic {i+1}" for i in range(num_topics)]
    df = pd.DataFrame(data, columns=column_names)
    df.index.name = "Document ID"
    return df

topic_document_matrix_LDA = best_model_LDA["topic-document-matrix"]
topic_distribution_df_LDA =
get_document_topic_percentages(topic_document_matrix_LDA)
topic_distribution_df_LDA.head(10)
```

```
             Topic 1  Topic 2  Topic 3  Topic 4  Topic 5  Topic 6
Topic 7  \
Document ID

0               0.03     0.03     0.03     0.03     0.03     0.03
0.03
1               0.19     2.68     0.36     1.12     2.18     1.21
0.34
2               0.25     0.49     0.91     1.05     7.00     1.43
0.54
3               0.00     0.43     0.00     0.18     8.61     0.83
0.00
4               0.21     0.01     0.01     2.16    11.04     4.61
3.06
5               0.39     0.01     0.39     0.26     2.53     8.84
```

```
1.24
6                 0.00     0.71     0.87     0.68    10.97     4.35
0.34
7                 0.01     0.01     1.24     1.06     2.34     0.07
0.45
8                 0.07     0.45     1.86     2.02    16.63     2.75
1.95
9                 0.37     0.29     0.85     1.55     7.01     0.78
1.69

             Topic 8  Topic 9  Topic 10  ...  Topic 64  Topic 65
Topic 66  \
Document ID                              ...

0                 0.03     0.03      0.03  ...      0.03      0.03
0.03
1                 1.00     0.50      0.67  ...      0.31      1.17
0.00
2                 1.57     1.35      2.61  ...      0.65      2.13
0.54
3                 1.36     0.01      0.43  ...      0.23      0.37
0.00
4                 0.33     3.96      1.46  ...      0.82      6.64
0.56
5                 0.89     1.15      2.87  ...      0.11      8.06
0.59
6                 0.82     3.14      1.12  ...      0.15      8.92
1.84
7                 1.49     0.01      5.21  ...      0.01      1.88
2.63
8                 0.93     1.02      0.58  ...      1.07      2.36
1.64
9                 2.14     2.56      0.45  ...      0.41      1.16
1.68

             Topic 67  Topic 68  Topic 69  Topic 70  Topic 71  Topic
72  \
Document ID

0                  0.03      0.03      0.03     36.64      0.03
0.03
1                  0.00      0.35      0.89      2.27      0.24
0.00
2                  0.43      0.18      2.91     12.59      0.84
0.00
3                  0.00      0.00      0.23      0.22      0.11
0.00
4                  0.01      1.11      3.25      3.79      0.50
0.01
5                  0.13      0.00      0.25      1.97      1.10
```

```
0.08
6                 0.17      0.85      4.13      2.10      0.29
0.00
7                 0.01      0.01      1.35     11.11      0.12
0.40
8                 0.02      0.17      1.51      3.10      1.34
0.00
9                 0.00      0.95      0.61      1.45      0.83
0.39

             Topic 73
Document ID
0                 0.03
1                 0.85
2                 0.90
3                 0.00
4                 0.80
5                 0.00
6                 1.26
7                 0.01
8                 2.48
9                 0.00

[10 rows x 73 columns]

original_indexes = dataset_final._Dataset__original_indexes
topic_distribution_df_LDA['Original Document Index'] =
original_indexes
```

The rows of the original document are sampled in a random order when passed into octis as a dataset in case the default split=True prameter is set. A choice whihc is necessary to allow for hold out date in the parameter optization. Thus the document order is scrambled. For the sake of retraining a model with optimal paramters on the full corpus, the parameter is set to False, the order of documents is maintained and can be easily rejoined. For safeties sake, we will join according to order of the indices function nonetheless.

In order to further enrich the data, we will add the sentiment to the dataframe using the defacto sentiment analysis standard VADER.

```
nltk.download('vader_lexicon')

def add_sentiment_scores(df, text_column):

    sia = SentimentIntensityAnalyzer()

    # Calculate sentiment scores
    df['sentiment'] = df[text_column].apply(lambda text:
sia.polarity_scores(text)['compound'])

    return df
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     /home/florian/nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
```

```
add_sentiment_scores(df_input, 'preprocessed_text')
```

```
                                       preprocessed_text  \
0    fragment adventure turned steed hoping cross d...
1    plague portion ensuing relating street manner ...
2    whatsoever away terms included language charac...
3    doll wangos leaving justice skill witches spea...
4    note text little work finished year intended i...
..                                                 ...
217  friend worth letters intimacies acquaintances ...
218  happened carriage crowd leaving ball begged go...
219  distinct reached ears henceforward distinctly ...
220  laugh came wonder wise man hippogriffs dragons...
221  forehead consultation left precisely walked gr...

                  reference  sentiment
0        Aikin_SirBertran_1    -0.9201
1    Ainsworth_OldSaintPa_1     0.9773
2      Ainsworth_Rookwood_1     0.9989
3    Ainsworth_TheLancash_1    -0.9998
4       Austen_Northanger_1     1.0000
..                      ...        ...
217     LeFanu_InaGlassDa_5     1.0000
218     LeFanu_InaGlassDa_6     0.9998
219     LeFanu_InaGlassDa_7     0.9999
220     LeFanu_InaGlassDa_8     1.0000
221     LeFanu_InaGlassDa_9    -0.9736

[222 rows x 3 columns]
```

```
df_input['input_index'] = df_input.index
df_input['ref'] = df_input['reference'].apply(lambda x: x.rsplit('_',
1)[0])
df_merge = df_input.merge(df_ref, left_on='ref', right_on='reference',
how='left')
df_merge['date'] = df_merge['date'].astype('Int64')
df_merge['birthdate'] = df_merge['birthdate'].astype('Int64')
df_merge.head()
```

```
                                 preprocessed_text_x
reference_x  \
0  fragment adventure turned steed hoping cross d...
Aikin_SirBertran_1
1  plague portion ensuing relating street manner ...
Ainsworth_OldSaintPa_1
2  whatsoever away terms included language charac...
```

```
Ainsworth_Rookwood_1
3  doll wangos leaving justice skill witches spea...
Ainsworth_TheLancash_1
4  note text little work finished year intended i...
Austen_Northanger_1

   sentiment  input_index                   ref  index
reference_y  \
0    -0.9201            0      Aikin_SirBertran      0
Aikin_SirBertran
1     0.9773            1  Ainsworth_OldSaintPa      1
Ainsworth_OldSaintPa
2     0.9989            2    Ainsworth_Rookwood      2
Ainsworth_Rookwood
3    -0.9998            3  Ainsworth_TheLancash      3
Ainsworth_TheLancash
4     1.0000            4     Austen_Northanger      5
Austen_Northanger

                                               title  \
0                            Sir Bertrand, A Fragment
1  Old Saint Paul's: A Tale of the Plague and the...
2                                           Rookwood
3  The Lancashire Witches: A Romance of Pendle Fo...
4                                   Northanger Abbey

                          author  date  ...      mode   genre
gender  \
0  Aikin, John and Anna Laetitia  1773  ...  Fragment  Gothic    NaN

1    Ainsworth, William Harrison  1841  ...       NaN     NaN      m

2    Ainsworth, William Harrison  1834  ...     Novel  Gothic      m

3    Ainsworth, William Harrison  1848  ...       NaN     NaN      m

4                   Austen, Jane  1817  ...       NaN     NaN      f


  birthdate  nationality role (central/peripheral/influence)  \
0      <NA>      English                             Central
1      1805      English                                 NaN
2      1805      English                             Central
3      1805      English                                 NaN
4      1775      English                                 NaN

                                                text     source  \
0  SIR BERTRAND, A FRAGMENT:\n\nAFTER this advent...     colors
1  OLD SAINT PAUL\'S\n\n  _A TALE OF THE PLAGUE\n...  pb-manual
2  \nThe Project Gutenberg EBook of Rookwood, by ...     colors
```

```
3  Proofreading Team.\n\n\n\n\n\n\n[Illustration:...  pb-manual
4  Northanger Abbey\n\n\nby Jane Austen\n\n(1803)...  gutenberg

                                 preprocessed_text_y  \
0  fragment adventure turned steed hoping cross d...
1  plague portion ensuing relating street manner ...
2  whatsoever away terms included language charac...
3  doll wangos leaving justice skill witches spea...
4  note text little work finished year intended i...

                                      tokenized_text
0  ['fragment', 'adventure', 'turned', 'steed', '...
1  ['plague', 'portion', 'ensuing', 'relating', '...
2  ['whatsoever', 'away', 'terms', 'included', 'l...
3  ['doll', 'wangos', 'leaving', 'justice', 'skil...
4  ['note', 'text', 'little', 'work', 'finished',...

[5 rows x 21 columns]
```

```
# merging of topic distribution with features
# reorganizing the order of columns and clean up
df_txt_features_LDA = df_merge.merge(topic_distribution_df_LDA,
right_on='Original Document Index', left_on='input_index')
df_txt_features_LDA=df_txt_features_LDA.drop(['text',
'preprocessed_text_y','tokenized_text','preprocessed_text_x', 'index',
'ref', 'Original Document Index'], axis=1)
df_txt_features_LDA.rename(columns={'reference_x':
'reference','reference_y': 'text_key'}, inplace=True)
df_txt_features_LDA = df_txt_features_LDA[['input_index'] + [col for
col in df_txt_features_LDA.columns if col != 'input_index']]
df_txt_features_LDA.rename(columns={'role
(central/peripheral/influence)': 'role'}, inplace=True)
df_txt_features_LDA.head()
```

```
   input_index               reference  sentiment
text_key  \
0            0      Aikin_SirBertran_1    -0.9201
Aikin_SirBertran
1            1  Ainsworth_OldSaintPa_1     0.9773
Ainsworth_OldSaintPa
2            2    Ainsworth_Rookwood_1     0.9989
Ainsworth_Rookwood
3            3  Ainsworth_TheLancash_1    -0.9998
Ainsworth_TheLancash
4            4     Austen_Northanger_1     1.0000
Austen_Northanger

                                               title  \
0                            Sir Bertrand, A Fragment
1  Old Saint Paul's: A Tale of the Plague and the...
```

```
2                                                   Rookwood
3  The Lancashire Witches: A Romance of Pendle Fo...
4                                           Northanger Abbey

                          author  date    period      mode
genre  ...  \
0  Aikin, John and Anna Laetitia  1773  Romantic  Fragment
Gothic  ...
1    Ainsworth, William Harrison  1841       NaN       NaN
NaN  ...
2    Ainsworth, William Harrison  1834       NaN     Novel
Gothic  ...
3    Ainsworth, William Harrison  1848       NaN       NaN
NaN  ...
4                   Austen, Jane  1817       NaN       NaN
NaN  ...

  Topic 64  Topic 65 Topic 66 Topic 67 Topic 68  Topic 69  Topic 70
Topic 71  \
0     0.03      0.03     0.03     0.03     0.03      0.03     36.64
0.03
1     0.31      1.17     0.00     0.00     0.35      0.89      2.27
0.24
2     0.65      2.13     0.54     0.43     0.18      2.91     12.59
0.84
3     0.23      0.37     0.00     0.00     0.00      0.23      0.22
0.11
4     0.82      6.64     0.56     0.01     1.11      3.25      3.79
0.50

   Topic 72  Topic 73
0      0.03      0.03
1      0.00      0.85
2      0.00      0.90
3      0.00      0.00
4      0.01      0.80

[5 rows x 88 columns]

df_txt_features_LDA.to_csv('./analysis/df_txt_features_LDA.csv',
index=False)
```

creating exportable model elements for use in pyLDAdavis in a alter step

```
topic_term_dists_LDA= best_model_LDA["topic-word-matrix"]
doc_topic_dists_LDA = best_model_LDA["topic-document-matrix"]
doc_topic_dists_LDA = doc_topic_dists_LDA.T
vocab = dataset_final.get_vocabulary()
doc_lengths = [len(doc) for doc in dataset_final.get_corpus()]
```

```python
corpus = dataset_final.get_corpus()

# Initialize term frequency dictionary
term_frequency_dict = {term: 0 for term in vocab}

# Count the frequency of each term in the corpus
for document in corpus:
    for word in document:
        if word in term_frequency_dict:
            term_frequency_dict[word] += 1

# Convert term frequencies to a list in the order of the vocabulary
term_frequency= [term_frequency_dict[word] for word in vocab]
```

Export for the analysis

```python
dump(topic_term_dists_LDA, './analysis/topic_term_dists_LDA.joblib')
dump(doc_topic_dists_LDA, './analysis/doc_topic_dists_LDA.joblib')
dump(vocab, './analysis/vocab.joblib')
dump(doc_lengths, './analysis/doc_lengths.joblib')
dump(term_frequency, './analysis/term_frequency.joblib')
```

```
['./analysis/term_frequency.joblib']
```

## CTM

```python
# Initialize the CTM model with some default parameters
ctm_model = CTM(batch_size=128, num_epochs=30,
inference_type='zeroshot', bert_model="bert-base-nli-mean-tokens")

# Define the hyperparameter space for CTM
parameter_space = {
    'num_topics': Integer(50, 100),
    'num_layers': Categorical([1, 2, 3]),
    'num_neurons': Categorical([100, 200, 300, 500, 750, 1000]),
    'learn_rate':Real(0.001, 0.1),
    'optimizer': Categorical(['adam', 'sgd', 'msprop']),
    'dropout': Real(0.0, 0.9, prior='uniform')
}

# Define the evaluation metric
coherence_centroid = WECoherenceCentroid(topk=20,
word2vec_path='./word2vec/word2vec_model.bin', binary=True)
DiversityCentroid = WordEmbeddingsInvertedRBOCentroid(topk=20,
weight=0.9, normalize=True,
word2vec_path='./word2vec/word2vec_model.bin', binary=True)

# Run the hyperparameter optimization
start = time.time()
```

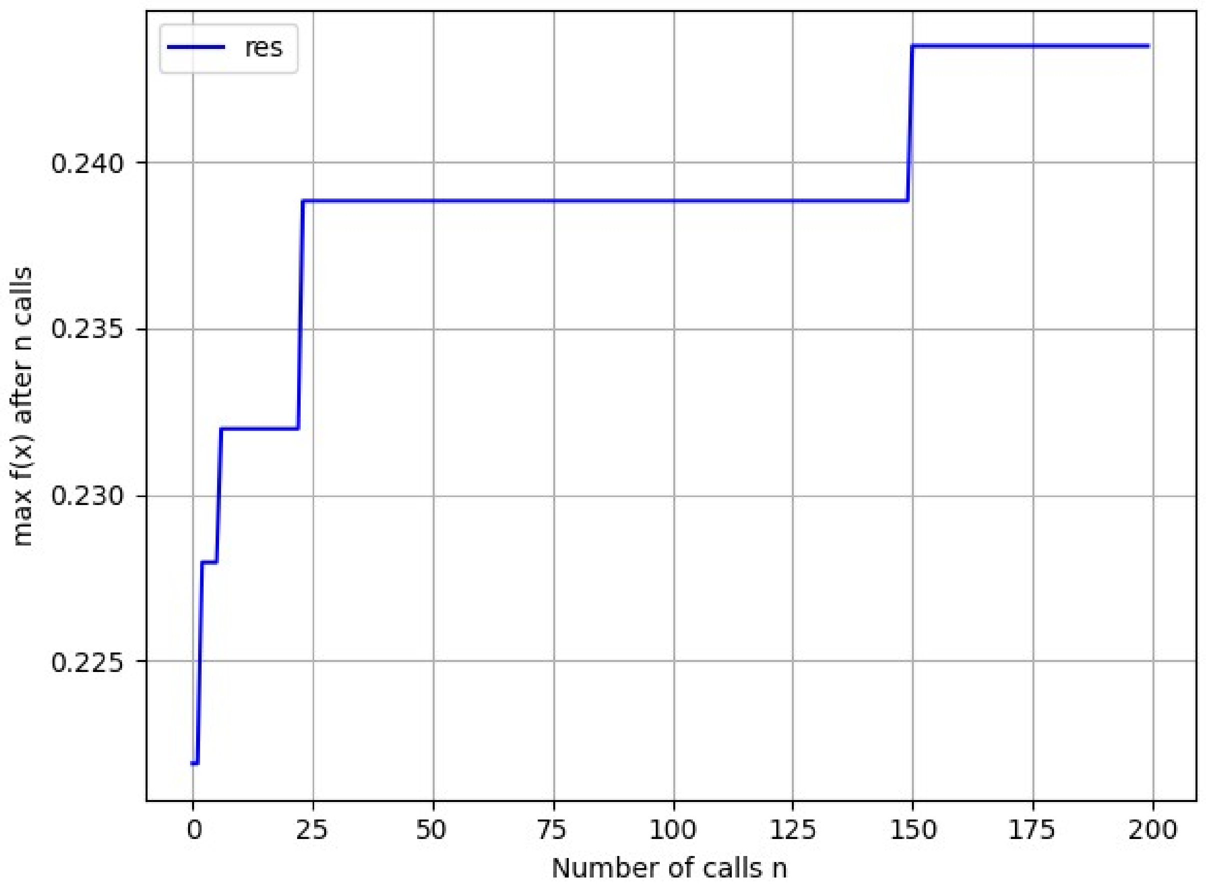
```
optimizer = Optimizer()
optimizer_results_CTM = optimizer.optimize(model=ctm_model,
dataset=dataset, metric=coherence_centroid,
                        search_space=parameter_space,
number_of_call=200,
                        n_random_starts=5, surrogate_model='RF',
                        model_runs=3, save_models=True, topk=20,
                        extra_metrics=[DiversityCentroid],
save_path='./octis/results/', plot_best_seen=True)

end = time.time()
duration = end - start

# Save the results
optimizer_results_CTM.save_to_csv("./octis/results/Opt_CTMresults.csv"
)

print('Optimizing model took: ' + str(round(duration)) + ' seconds.')

dump(optimizer_results_CTM,
'./octis/models/optimizer_results_CTM.joblib', compress=('lzma', 9))
```

```
['./octis/models/optimizer_results_CTM.joblib']
```

```
optimizer_results_CTM =
load('./octis/models/optimizer_results_CTM.joblib')

# the optimization process here is one of maximization, yet the
provided function for plotting
# assumes the oposite, so we shall invert the result.

values_to_plot_CTM = [-x for x in optimizer_results_CTM.info['f_val']]

plot_bayesian_optimization(values=values_to_plot_CTM,
                          name_plot="CTM_Optimization_Convergence",
                          log_scale=False,
                          conv_max=True)
```

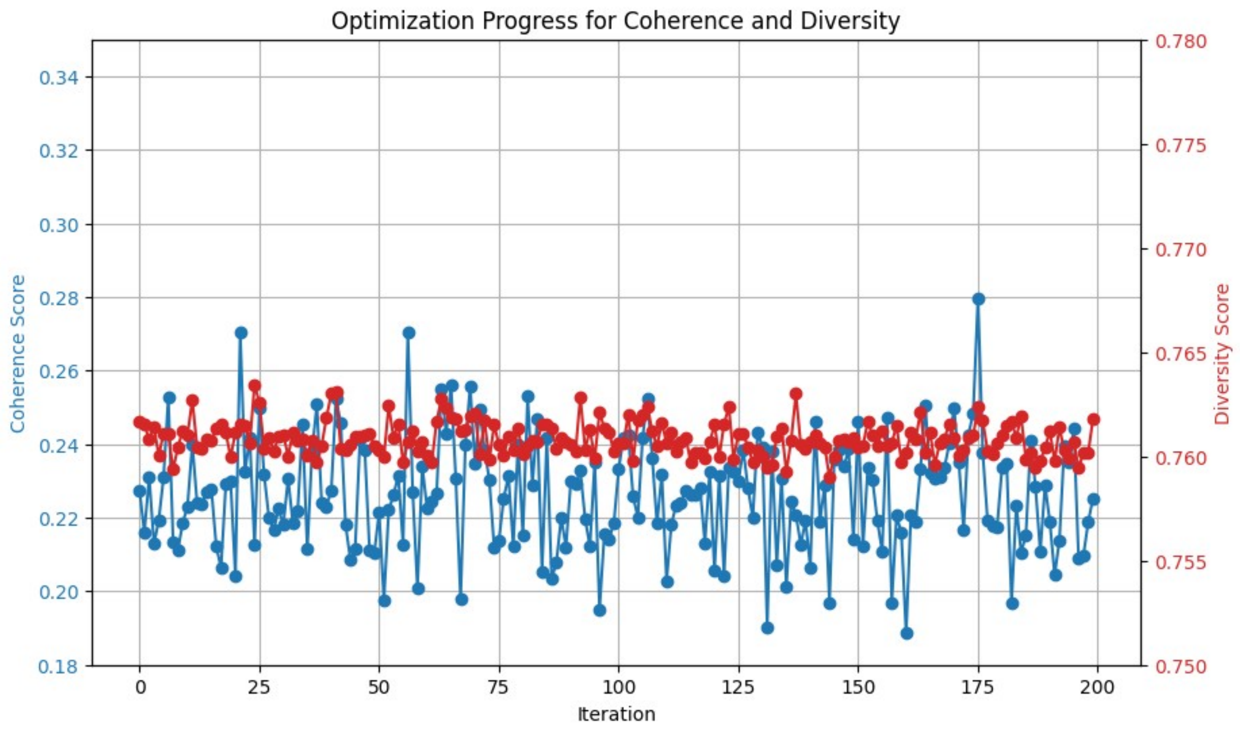
This optimization was less smooth and clear, additional runs could potentially still improve upon the model

```
#The image is just dumped into the working directory, we shall move it
somewhere more fitting
os.replace('./CTM_Optimization_Convergence.png',
'./octis/results/CTM_Optimization_Convergence.png')

Image(filename='./octis/results/CTM_Optimization_Convergence.png')
```



```python
# Plotting WECoherenceCentroid' and
'WordEmbeddingsInvertedRBOCentroid' our two metrics
coherence_scores_CTM = [max(run) for run in
optimizer_results_CTM.info['dict_model_runs']
['WECoherenceCentroid'].values()]
diversity_scores_CTM = [max(run) for run in
optimizer_results_CTM.info['dict_model_runs']
['0_WordEmbeddingsInvertedRBOCentroid'].values()]

# Create a figure with twin y-axis for the second metric
fig, ax1 = plt.subplots(figsize=(10, 6))

ax1.set_xlabel('Iteration')
ax1.set_ylabel('Coherence Score', color='tab:blue')
ax1.plot(coherence_scores_CTM, marker='o', color='tab:blue')
ax1.tick_params(axis='y', labelcolor='tab:blue')
ax1.set_ylim(0.18, 0.35)

ax2 = ax1.twinx()
ax2.set_ylabel('Diversity Score', color='tab:red')
ax2.plot(diversity_scores_CTM, marker='o', color='tab:red')
ax2.tick_params(axis='y', labelcolor='tab:red')
```

```
ax2.set_ylim(0.75, 0.78)

plt.title('Optimization Progress for Coherence and Diversity')
ax1.grid(True)

plt.show()
```



```
# Extract the best configurations based on coherence scores
coherence_scores_CTM = optimizer_results_CTM.info['dict_model_runs']
['WECoherenceCentroid']
diversity_scores_CTM = optimizer_results_CTM.info['dict_model_runs']
['0_WordEmbeddingsInvertedRBOCentroid']

# Extracting scores and hyperparameter configurations
combined_scores = []
for i, (coherence_run, diversity_run) in
enumerate(zip(coherence_scores_CTM.values(),
diversity_scores_CTM.values())):
    max_coherence = max(coherence_run)
    max_diversity = max(diversity_run)
    num_topics = optimizer_results_CTM.info['x_iters']['num_topics']
[i]
    num_layers = optimizer_results_CTM.info['x_iters']['num_layers']
[i]
    num_neurons = optimizer_results_CTM.info['x_iters']['num_neurons']
[i]
    learn_rate = optimizer_results_CTM.info['x_iters']['learn_rate']
```

```
[i]
    optimizer_param = optimizer_results_CTM.info['x_iters']
['optimizer'][i]
    dropout = optimizer_results_CTM.info['x_iters']['dropout'][i]

    combined_scores.append((max_coherence, max_diversity, num_topics,
num_layers, num_neurons, learn_rate, optimizer_param, dropout))

# Sort by coherence score first and take the top ten
top_ten_by_coherence = sorted(combined_scores, key=lambda x: x[0],
reverse=True)[:10]

# Select the configuration with the highest diversity score from the
top ten
best_overall_CTM = max(top_ten_by_coherence, key=lambda x: x[1])

# Print the best configuration
print("Best Coherence Score:", best_overall_CTM[0])
print("Corresponding Diversity Score:", best_overall_CTM[1])
print("Best Configuration: num_topics =", best_overall_CTM[2], ",
num_layers =", best_overall_CTM[3], ", num_neurons =",
best_overall_CTM[4],
      ", learn_rate =", best_overall_CTM[5], ", optimizer =",
best_overall_CTM[6], ", dropout =", best_overall_CTM[7])
```

```
Best Coherence Score: 0.25242652543794375
Corresponding Diversity Score: 0.7631295026862059
Best Configuration: num_topics = 52 , num_layers = 1 , num_neurons =
1000 , learn_rate = 0.026875343602901114 , optimizer = sgd , dropout =
0.336284177347719
```

```
best_num_topics = best_overall_CTM[2]
best_num_layers = best_overall_CTM[3]
best_num_neurons = best_overall_CTM[4]
best_learn_rate = best_overall_CTM[5]
best_optimizer = best_overall_CTM[6]
best_dropout = best_overall_CTM[7]

# Now we instantiate the CTM model with the best hyperparameters
best_model_CTM = CTM(num_topics=best_num_topics,
                     num_layers=best_num_layers,
                     num_neurons=best_num_neurons,
                     solver=best_optimizer,
                     dropout=best_dropout,
                     batch_size=128,
                     use_partitions=False,
                     bert_path='./octis/models/'
                     #bert_path='.'
                     )
```

```python
# Train the model with the dataset
best_model_CTM = best_model_CTM.train_model(dataset_final)

#dump(best_model_CTM, './octis/models/best_model_CTM.joblib',
compress=('lzma', 9))
best_model_CTM = load('./octis/models/best_model_CTM.joblib')

top_words_per_topic_CTM = best_model_CTM['topics']

# Display the top words for each topic
for topic_id, words in enumerate(top_words_per_topic_CTM):
    print("Topic", topic_id + 1, ":", words)
```

```
Topic 1 : ['driver', 'prophet', 'fruit', 'pit', 'royal', 'hither',
'queen', 'revenge', 'interview', 'consequences']
Topic 2 : ['driver', 'barn', 'marquis', 'marchioness', 'landlady',
'burrow', 'camels', 'vicar', 'farmer', 'crows']
Topic 3 : ['er', 'major', 'thro', 'evils', 'base', 'feeble',
'splendid', 'horrors', 'goodness', 'victims']
Topic 4 : ['curate', 'card', 'murdered', 'ceiling', 'punch', 'cornet',
'harbor', 'mill', 'treasure', 'cock']
Topic 5 : ['beds', 'crypt', 'lid', 'candles', 'lodged', 'honours',
'hath', 'daddy', 'trenchers', 'untied']
Topic 6 : ['lovers', 'build', 'dismal', 'site', 'earthly', 'suites',
'portal', 'brows', 'millions', 'slab']
Topic 7 : ['honourable', 'valid', 'reserving', 'vicar', 'arbours',
'partisan', 'sum', 'fashionable', 'crape', 'terrace']
Topic 8 : ['valid', 'mill', 'wagon', 'showman', 'vicar', 'hay',
'pinched', 'cabinets', 'barn', 'honourable']
Topic 9 : ['swiftly', 'churchyard', 'onward', 'ing', 'dressing',
'nobleman', 'con', 'woke', 'deliberately', 'brightly']
Topic 10 : ['honourable', 'harbor', 'toll', 'political', 'terrace',
'fashionable', 'sexton', 'auks', 'dismal', 'gills']
Topic 11 : ['column', 'recollections', 'marquis', 'frivolous',
'marchioness', 'fetch', 'meditation', 'deity', 'morbid', 'disorder']
Topic 12 : ['prophet', 'guilt', 'recollections', 'disgrace',
'remorse', 'sworn', 'heavens', 'cistern', 'nephew', 'ties']
Topic 13 : ['paw', 'sergeant', 'major', 'talisman', 'mask', 'abbot',
'knock', 'pinched', 'suites', 'examined']
Topic 14 : ['royal', 'galley', 'camp', 'proclaimed', 'steeple',
'tent', 'military', 'throne', 'queen', 'seclusion']
Topic 15 : ['beach', 'surf', 'weed', 'wave', 'pebbles', 'seclusion',
'precipice', 'chasm', 'tread', 'jammo']
Topic 16 : ['snaky', 'snakes', 'locks', 'wallet', 'sisters',
'slippers', 'helmet', 'shield', 'winged', 'fisherman']
Topic 17 : ['prophet', 'sworn', 'kings', 'beating', 'kingdom',
'queen', 'captain', 'sire', 'ridiculous', 'cistern']
Topic 18 : ['sorr', 'cavern', 'sez', 'cigar', 'spectre', 'loike',
'heat', 'gas', 'afterward', 'luncheon']
Topic 19 : ['camels', 'burrow', 'crows', 'wagon', 'showman', 'camp',
'brahmin', 'ledger', 'mummy', 'merry']
Topic 20 : ['mummy', 'congregation', 'unhappiness', 'vicar', 'galley',
'basely', 'steeple', 'fishermen', 'churchyard', 'fourth']
Topic 21 : ['fruit', 'gods', 'prince', 'ass', 'devotee', 'deity',
'throne', 'royal', 'religious', 'apple']
Topic 22 : ['governess', 'cheaile', 'brandy', 'gallery', 'link',
'background', 'slender', 'epitaph', 'm', 'housekeeper']
Topic 23 : ['crypt', 'sex', 'daddy', 'reserving', 'knack', 'valid',
'baby', 'vault', 'she', 'ride']
Topic 24 : ['mummy', 'flag', 'gods', 'harbor', 'hast', 'queen',
'gallery', 'singer', 'fourth', 'nobleman']
Topic 25 : ['honourable', 'willows', 'tent', 'vicar', 'housekeeper',
'fashionable', 'convict', 'sentry', 'poop', 'closet']
Topic 26 : ['crows', 'burrow', 'camels', 'valid', 'gas', 'arbours',
'portal', 'wedding', 'sails', 'chest']
Topic 27 : ['gallery', 'curate', 'helmet', 'military', 'emperor',
'punch', 'card', 'trap', 'encampment', 'camp']
Topic 28 : ['christabel', 'er', 'shield', 'prayed', 'spake', 'valid',
'sire', 'hath', 'doth', 'dove']
Topic 29 : ['sensibility', 'regiment', 'prince', 'stole', 'gaiety',
'continue', 'elegance', 'colonel', 'poverty', 'fort']
Topic 30 : ['rickshaw', 'mistake', 'farmer', 'harbor', 'bungalow',
'bonnet', 'auks', 'ride', 'gills', 'etched']
Topic 31 : ['snakes', 'shield', 'snaky', 'helmet', 'winged', 'locks',
'sisters', 'wallet', 'slippers', 'galley']
Topic 32 : ['ernest', 'album', 'snapped', 'isn', 'fountain',
'sculptor', 'audience', 'doesn', 'studio', 'grandfather']
Topic 33 : ['luck', 'club', 'goblins', 'miner', 'fun', 'millionaire',
'stake', 'players', 'if', 'copies']
Topic 34 : ['painter', 'convict', 'sentry', 'poop', 'canvass',
'paintings', 'cuddy', 'turret', 'main', 'painting']
Topic 35 : ['onnur', 'pinched', 'vault', 'revenge', 'chaise',
'disorder', 'size', 'crape', 'yard', 'minster']
Topic 36 : ['hay', 'barn', 'vicar', 'harbor', 'knife', 'realised',
'rope', 'pine', 'vivid', 'auks']
Topic 37 : ['flag', 'paint', 'gas', 'cart', 'singer', 'riding',
'bangle', 'purple', 'outline', 'attacked']
Topic 38 : ['rickshaw', 'mistake', 'bungalow', 'fan', 'sailor',
'wrist', 'verandah', 'whir', 'unhappiness', 'bungalows']
Topic 39 : ['toll', 'sergeant', 'captain', 'pirates', 'ice', 'now',
'fort', 'fishermen', 'science', 'cave']
Topic 40 : ['bushes', 'onnur', 'tent', 'pit', 'chasm', 'afterward',
'onward', 'structure', 'gun', 'continually']
Topic 41 : ['canoe', 'landing', 'cypresses', 'bespeak', 'sexton',
'ice', 'hath', 'betrothal', 'echoed', 'upstairs']
Topic 42 : ['onnur', 'pinched', 'rickshaw', 'mistake', 'bungalow',
'ing', 'con', 'chaise', 'tent', 'rode']
Topic 43 : ['wrist', 'mummy', 'wagon', 'showman', 'pinched', 'claws',
'camp', 'instructions', 'examined', 'bangle']
```

```
Topic 44 : ['monster', 'labyrinth', 'bull', 'housekeeper', 'royal',
'throne', 'goblet', 'sandals', 'closet', 'nephews']
Topic 45 : ['camp', 'canoe', 'bag', 'vacuum', 'landing', 'pistol',
'keyhole', 'realised', 'fireplace', 'whisky']
Topic 46 : ['curate', 'suites', 'knack', 'murdered', 'untied',
'cheaile', 'brig', 'chisel', 'editor', 'punch']
Topic 47 : ['youthful', 'ages', 'nunnery', 'lid', 'governor',
'untied', 'eldest', 'sensibility', 'gaiety', 'lively']
Topic 48 : ['canoe', 'nunnery', 'landing', 'upstairs', 'mask',
'verandah', 'governor', 'onnur', 'indian', 'rushing']
Topic 49 : ['plague', 'grocer', 'apprentice', 'willows', 'gallant',
'rejoined', 'lane', 'tent', 'pestilence', 'verger']
Topic 50 : ['mummy', 'singer', 'ceiling', 'swelling', 'gods',
'harpsichord', 'phrase', 'queen', 'hast', 'organ']
Topic 51 : ['sculptor', 'chisel', 'epitaph', 'wagon', 'showman',
'beds', 'earthly', 'slabs', 'slab', 'pirates']
Topic 52 : ['mill', 'abbot', 'gallery', 'helmet', 'sailor', 'godly',
'vault', 'lid', 'transformation', 'regent']
```

```
dump(top_words_per_topic_CTM,'./analysis/
top_words_per_topic_CTM.joblib')
```

```
['./analysis/top_words_per_topic_CTM.joblib']
```

```
topic_document_matrix_CTM = best_model_CTM["topic-document-matrix"]
topic_distribution_df_CTM =
get_document_topic_percentages(topic_document_matrix_CTM)
topic_distribution_df_CTM.head(10)
```

```
             Topic 1  Topic 2  Topic 3  Topic 4  Topic 5  Topic 6
Topic 7  \
Document ID

0               0.78     1.18     2.27     1.59     0.70     1.37
1.25
1               2.97     1.33     1.90     3.48     1.10     1.64
1.85
2               2.11     2.37     1.06     3.34     2.25     2.11
2.52
3               0.58     0.92     1.63     0.73     0.29     0.67
0.41
4               0.88     1.27     1.37     1.83     2.86     1.16
1.34
5               1.27     0.82     1.26     1.07     3.70     2.22
1.25
6               4.73     1.62     0.86     2.07     1.09     1.16
2.49
7               3.19     2.26     1.36     2.19     4.39     1.96
1.61
8               7.44     2.23     1.62     1.23     3.45     1.84
```

```
1.78
9                2.79     2.82     1.06     3.18     2.21     1.62
2.52

             Topic 8  Topic 9  Topic 10  ...  Topic 43  Topic 44
Topic 45  \
Document ID                                ...

0                4.50     1.59      2.67  ...      0.91      0.98
1.23
1                1.76     2.84      8.41  ...      3.51      1.10
0.51
2                2.24     0.66      2.45  ...      6.01      1.54
1.61
3                1.69     0.83      0.74  ...      0.30      0.45
1.51
4                1.29     1.35      1.67  ...      4.09      1.60
0.87
5                1.02     4.40      1.79  ...      5.15      1.42
0.78
6                1.82     0.56      1.84  ...      1.02      0.79
1.59
7                3.26     1.40      2.34  ...      3.38      1.50
1.29
8                1.19     1.02      2.62  ...      1.44      1.26
0.94
9                1.06     2.07      1.50  ...      2.41      0.85
1.77

             Topic 46  Topic 47  Topic 48  Topic 49  Topic 50  Topic
51  \
Document ID

0                 1.98      1.69      1.30      3.03      1.07
1.60
1                 2.30      1.73      1.40      0.78      3.97
0.98
2                 3.70      1.07      2.12      1.32      2.10
1.49
3                 1.10      2.33      0.88      0.54      4.56
0.52
4                 1.67      1.99      2.46      1.02      1.32
1.94
5                 1.79      0.89      0.59      1.14      1.85
1.42
6                 2.73      1.97      3.17      1.93      0.46
1.35
7                 1.79      1.93      1.70      1.02      0.69
1.67
8                 2.21      1.05      2.93      1.32      1.01
```

```
1.83
9                 2.93      1.59      1.19      1.04      0.88
2.11

             Topic 52
Document ID
0                6.30
1                2.00
2                0.65
3               25.20
4                1.39
5                0.31
6                2.75
7                0.52
8                1.17
9                1.65

[10 rows x 52 columns]

original_indexes = dataset_final._Dataset__original_indexes
topic_distribution_df_CTM['Original Document Index'] =
original_indexes

df_input['input_index'] = df_input.index
df_input['ref'] = df_input['reference'].apply(lambda x: x.rsplit('_',
1)[0])
df_merge = df_input.merge(df_ref, left_on='ref', right_on='reference',
how='left')
df_merge['date'] = df_merge['date'].astype('Int64')
df_merge['birthdate'] = df_merge['birthdate'].astype('Int64')
df_merge.head()

                                 preprocessed_text_x
reference_x  \
0  fragment adventure turned steed hoping cross d...
Aikin_SirBertran_1
1  plague portion ensuing relating street manner ...
Ainsworth_OldSaintPa_1
2  whatsoever away terms included language charac...
Ainsworth_Rookwood_1
3  doll wangos leaving justice skill witches spea...
Ainsworth_TheLancash_1
4  note text little work finished year intended i...
Austen_Northanger_1

   sentiment  input_index                   ref  index
reference_y  \
0    -0.9201            0      Aikin_SirBertran      0
Aikin_SirBertran
1     0.9773            1  Ainsworth_OldSaintPa      1
```

```
Ainsworth_OldSaintPa
2     0.9989            2    Ainsworth_Rookwood      2
Ainsworth_Rookwood
3    -0.9998            3  Ainsworth_TheLancash      3
Ainsworth_TheLancash
4     1.0000            4     Austen_Northanger      5
Austen_Northanger

                                               title  \
0                          Sir Bertrand, A Fragment
1  Old Saint Paul's: A Tale of the Plague and the...
2                                           Rookwood
3  The Lancashire Witches: A Romance of Pendle Fo...
4                                   Northanger Abbey

                          author  date  ...      mode   genre
gender  \
0  Aikin, John and Anna Laetitia  1773  ...  Fragment  Gothic    NaN

1    Ainsworth, William Harrison  1841  ...       NaN     NaN      m

2    Ainsworth, William Harrison  1834  ...     Novel  Gothic      m

3    Ainsworth, William Harrison  1848  ...       NaN     NaN      m

4                   Austen, Jane  1817  ...       NaN     NaN      f


  birthdate  nationality role (central/peripheral/influence)  \
0      <NA>      English                             Central
1      1805      English                                 NaN
2      1805      English                             Central
3      1805      English                                 NaN
4      1775      English                                 NaN

                                                text     source  \
0  SIR BERTRAND, A FRAGMENT:\n\nAFTER this advent...     colors
1  OLD SAINT PAUL\'S\n\n  _A TALE OF THE PLAGUE\n...  pb-manual
2  \nThe Project Gutenberg EBook of Rookwood, by ...     colors
3  Proofreading Team.\n\n\n\n\n\n\n\n[Illustration:...  pb-manual
4  Northanger Abbey\n\n\nby Jane Austen\n\n(1803)...  gutenberg

                                 preprocessed_text_y  \
0  fragment adventure turned steed hoping cross d...
1  plague portion ensuing relating street manner ...
2  whatsoever away terms included language charac...
3  doll wangos leaving justice skill witches spea...
4  note text little work finished year intended i...

                                      tokenized_text
```

```
0  ['fragment', 'adventure', 'turned', 'steed', '...
1  ['plague', 'portion', 'ensuing', 'relating', '...
2  ['whatsoever', 'away', 'terms', 'included', 'l...
3  ['doll', 'wangos', 'leaving', 'justice', 'skil...
4  ['note', 'text', 'little', 'work', 'finished',...

[5 rows x 21 columns]

# merging of topic distribution with features
# reorganizing the order of columns and clean up
df_txt_features_CTM= df_merge.merge(topic_distribution_df_CTM,
right_on='Original Document Index', left_on='input_index')
df_txt_features_CTM=df_txt_features_CTM.drop(['text',
'preprocessed_text_y','tokenized_text','preprocessed_text_x', 'index',
'ref', 'Original Document Index'], axis=1)
df_txt_features_CTM.rename(columns={'reference_x':
'reference','reference_y': 'text_key'}, inplace=True)
df_txt_features_CTM = df_txt_features_CTM[['input_index'] + [col for
col in df_txt_features_CTM.columns if col != 'input_index']]
df_txt_features_CTM.rename(columns={'role
(central/peripheral/influence)': 'role'}, inplace=True)
df_txt_features_CTM.head()

   input_index               reference  sentiment
text_key  \
0            0      Aikin_SirBertran_1    -0.9201
Aikin_SirBertran
1            1  Ainsworth_OldSaintPa_1     0.9773
Ainsworth_OldSaintPa
2            2    Ainsworth_Rookwood_1     0.9989
Ainsworth_Rookwood
3            3  Ainsworth_TheLancash_1    -0.9998
Ainsworth_TheLancash
4            4     Austen_Northanger_1     1.0000
Austen_Northanger

                                               title  \
0                            Sir Bertrand, A Fragment
1  Old Saint Paul's: A Tale of the Plague and the...
2                                            Rookwood
3  The Lancashire Witches: A Romance of Pendle Fo...
4                                    Northanger Abbey

                            author  date    period      mode
genre  ...  \
0  Aikin, John and Anna Laetitia  1773  Romantic  Fragment
Gothic  ...
1    Ainsworth, William Harrison  1841       NaN       NaN
NaN  ...
2    Ainsworth, William Harrison  1834       NaN     Novel
```

```
Gothic  ...
3    Ainsworth, William Harrison  1848        NaN        NaN
NaN  ...
4                  Austen, Jane  1817        NaN        NaN
NaN  ...

  Topic 43  Topic 44 Topic 45 Topic 46 Topic 47  Topic 48  Topic 49
Topic 50  \
0     0.91       0.98     1.23     1.98     1.69      1.30      3.03
1.07
1     3.51       1.10     0.51     2.30     1.73      1.40      0.78
3.97
2     6.01       1.54     1.61     3.70     1.07      2.12      1.32
2.10
3     0.30       0.45     1.51     1.10     2.33      0.88      0.54
4.56
4     4.09       1.60     0.87     1.67     1.99      2.46      1.02
1.32

   Topic 51  Topic 52
0      1.60      6.30
1      0.98      2.00
2      1.49      0.65
3      0.52     25.20
4      1.94      1.39

[5 rows x 67 columns]
```

```
df_txt_features_CTM.to_csv('./analysis/df_txt_features_CTM.csv',
index=False)
#
df_txt_features_CTM=pd.read_csv('./analysis/df_txt_features_CTM.csv')
```

pyLDAvis exports

```
topic_term_dists_CTM= best_model_CTM["topic-word-matrix"]
doc_topic_dists_CTM = best_model_CTM["topic-document-matrix"]
doc_topic_dists_CTM = doc_topic_dists_CTM.T

dump(topic_term_dists_CTM, './analysis/topic_term_dists_CTM.joblib')
dump(doc_topic_dists_CTM, './analysis/doc_topic_dists_CTM.joblib')
```

```
['./analysis/doc_topic_dists_CTM.joblib']
```

## ETM

```
# Initialize the CTM model with some default parameters
etm_model = ETM(batch_size=128, num_epochs=30,
embeddings_path='word2vec_model.bin')
```

```
# Define the hyperparameter space for CTM
parameter_space = {
    'num_topics': Integer(50, 100),
    'num_layers': Categorical([1, 2, 3]),
    'num_neurons': Categorical([100, 200, 300, 500, 750, 1000]),
    'learn_rate':Real(0.001, 0.1),
    'activation': Categorical(['sigmoid', 'softplus', 'selu']),
    'optimizer': Categorical(['adam', 'sgd', 'msprop']),
    'dropout': Real(0.0, 0.9, prior='uniform')
}


# Define the evaluation metric
coherence_centroid = WECoherenceCentroid(topk=20,
word2vec_path='./word2vec/word2vec_model.bin', binary=True)
DiversityCentroid = WordEmbeddingsInvertedRBOCentroid(topk=20,
weight=0.9, normalize=True,
word2vec_path='./word2vec/word2vec_model.bin', binary=True)

# Run the hyperparameter optimization
start = time.time()

optimizer = Optimizer()
optimizer_results_ETM = optimizer.optimize(model=etm_model,
dataset=dataset, metric=coherence_centroid,
                        search_space=parameter_space,
number_of_call=200,
                        n_random_starts=5, surrogate_model='RF',
                        model_runs=3, save_models=True, topk=20,
                        extra_metrics=[DiversityCentroid],
save_path='./octis/results/', plot_best_seen=True)

end = time.time()
duration = end - start

# Save the results
optimizer_results_ETM.save_to_csv("./octis/results/Opt_ETMresults.csv"
)

print('Optimizing model took: ' + str(round(duration)) + ' seconds.')

dump(optimizer_results_ETM,
'./octis/models/optimizer_results_ETM.joblib', compress=('lzma', 9))
```

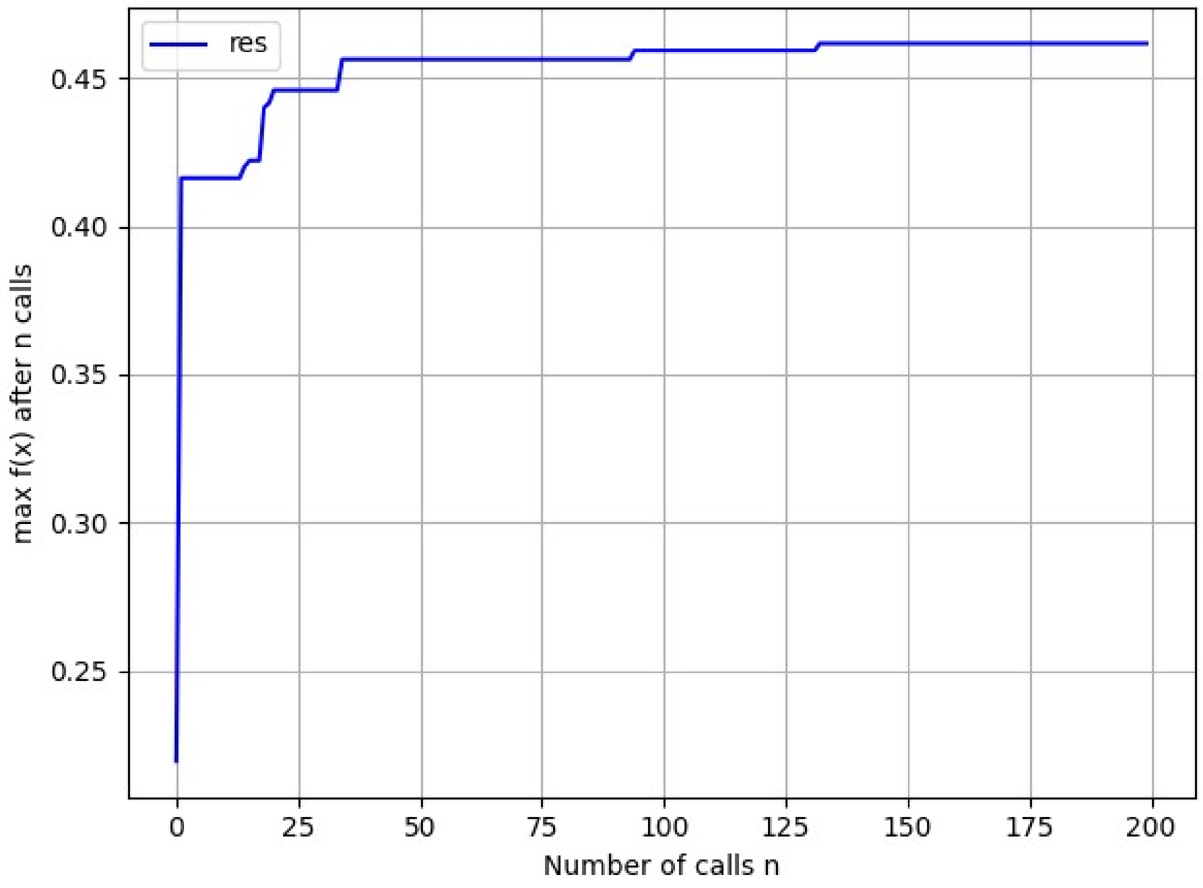
```
['./octis/models/optimizer_results_ETM.joblib']
```

```
optimizer_results_ETM =
load('./octis/models/optimizer_results_ETM.joblib')

# the optimization process here is one of maximization, yet the
provided function for plotting
```

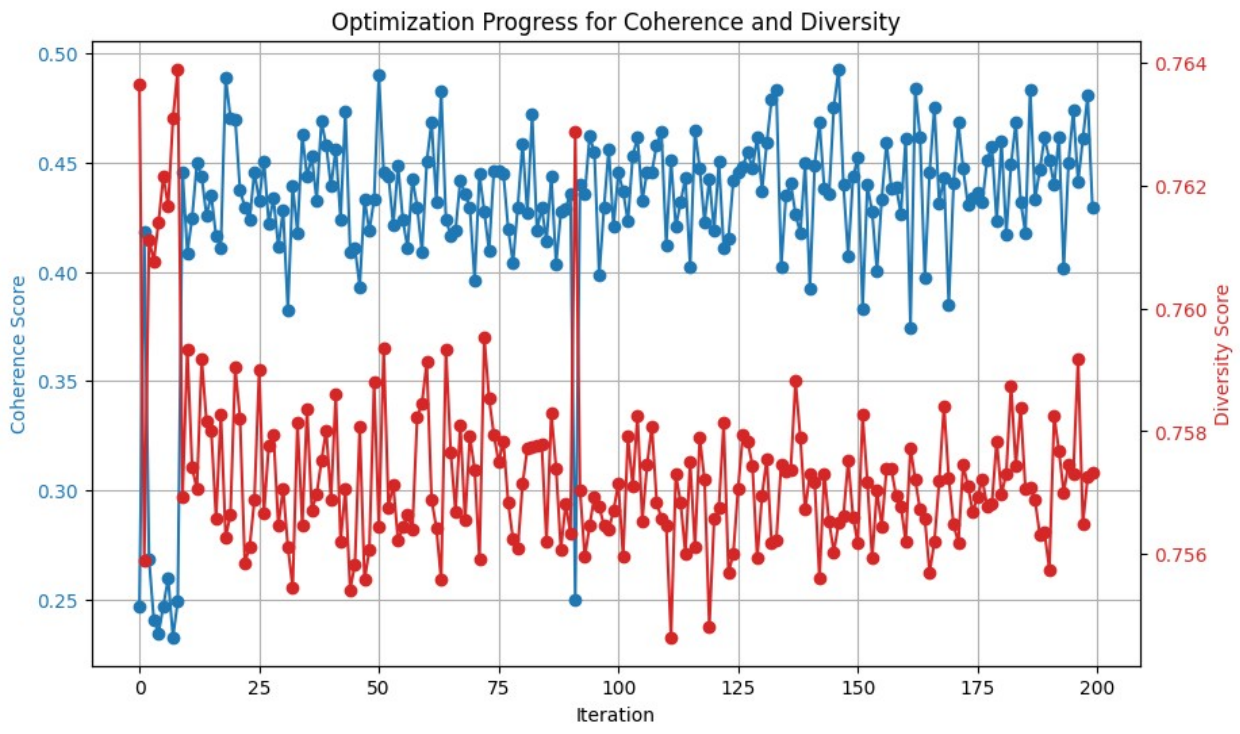
```
# assumes the oposite, so we shall invert the result.

values_to_plot_ETM = [-x for x in optimizer_results_ETM.info['f_val']]

plot_bayesian_optimization(values=values_to_plot_ETM,
                           name_plot="ETM_Optimization_Convergence",
                           log_scale=False,
                           conv_max=True)
```

The ETM seems to have performed best on this constellation. It seems that the quality of this model is highly dependant on the size of the text segments and imporves with fewer splits within one document. Which is in intuitive, given that word2vec embeddings put a lot of mphasis on the ducument context and the relatins between its words.

```
#The image is just dumped into the working directory, we shall move ti
somewhere more fitting

os.replace('./ETM_Optimization_Convergence.png',
'./octis/results/ETM_Optimization_Convergence.png')

Image(filename='./octis/results/ETM_Optimization_Convergence.png')
```



```
# Plotting WECoherenceCentroid' and
'WordEmbeddingsInvertedRBOCentroid' our two metrics
coherence_scores_ETM = [max(run) for run in
optimizer_results_ETM.info['dict_model_runs']
['WECoherenceCentroid'].values()]
diversity_scores_ETM = [max(run) for run in
optimizer_results_ETM.info['dict_model_runs']
['0_WordEmbeddingsInvertedRBOCentroid'].values()]

# Create a figure with twin y-axis for the second metric
fig, ax1 = plt.subplots(figsize=(10, 6))

ax1.set_xlabel('Iteration')
ax1.set_ylabel('Coherence Score', color='tab:blue')
ax1.plot(coherence_scores_ETM, marker='o', color='tab:blue')
ax1.tick_params(axis='y', labelcolor='tab:blue')
#ax1.set_ylim(0.23, 0.50)

ax2 = ax1.twinx()
ax2.set_ylabel('Diversity Score', color='tab:red')
ax2.plot(diversity_scores_ETM, marker='o', color='tab:red')
ax2.tick_params(axis='y', labelcolor='tab:red')
#ax2.set_ylim(0.75, 0.78)

plt.title('Optimization Progress for Coherence and Diversity')
ax1.grid(True)

plt.show()
```



```
# Extract the best configurations based on coherence scores
coherence_scores_ETM = optimizer_results_ETM.info['dict_model_runs']
['WECoherenceCentroid']
diversity_scores_ETM = optimizer_results_ETM.info['dict_model_runs']
['0_WordEmbeddingsInvertedRBOCentroid']

# Extracting scores and hyperparameter configurations
combined_scores = []
for i, (coherence_run, diversity_run) in
enumerate(zip(coherence_scores_ETM.values(),
diversity_scores_ETM.values())):
    max_coherence = max(coherence_run)
    max_diversity = max(diversity_run)
    num_topics = optimizer_results_ETM.info['x_iters']['num_topics']
[i]
    num_layers = optimizer_results_ETM.info['x_iters']['num_layers']
[i]
    num_neurons = optimizer_results_ETM.info['x_iters']['num_neurons']
[i]
    learn_rate = optimizer_results_ETM.info['x_iters']['learn_rate']
[i]
    activation = optimizer_results_ETM.info['x_iters']['activation']
[i]
    optimizer_param = optimizer_results_ETM.info['x_iters']
['optimizer'][i]
    dropout = optimizer_results_ETM.info['x_iters']['dropout'][i]
    combined_scores.append((max_coherence, max_diversity, num_topics,
num_layers, num_neurons, learn_rate, activation, optimizer_param,
dropout))

# Sort by coherence score first and take the top ten
top_ten_by_coherence = sorted(combined_scores, key=lambda x: x[0],
reverse=True)[:10]

# Select the configuration with the highest diversity score from the
top ten
best_overall_ETM = max(top_ten_by_coherence, key=lambda x: x[1])

# Print the best configuration
print("Best Coherence Score:", best_overall_ETM[0])
print("Corresponding Diversity Score:", best_overall_ETM[1])
print("Best Configuration: num_topics =", best_overall_ETM[2],
      ", num_layers =", best_overall_ETM[3],
      ", num_neurons =", best_overall_ETM[4],
      ", learn_rate =", best_overall_ETM[5],
      ", activation =", best_overall_ETM[6],
      ", optimizer =", best_overall_ETM[7],
      ", dropout =", best_overall_ETM[8])
```

```
Best Coherence Score: 0.4806725309903266
Corresponding Diversity Score: 0.7572589879575945
Best Configuration: num_topics = 99 , num_layers = 1 , num_neurons =
750 , learn_rate = 0.04655802555184136 , activation = sigmoid ,
optimizer = adam , dropout = 0.03639473996509752
```

```
best_num_topics = best_overall_ETM[2]
best_num_layers = best_overall_ETM[3]
best_num_neurons = best_overall_ETM[4]
best_learn_rate = best_overall_ETM[5]
best_activation = best_overall_ETM[6]
best_optimizer = best_overall_ETM[7]
best_dropout = best_overall_ETM[8]

# Now we instantiate the ETM model with the best hyperparameters
best_model_ETM = ETM(num_topics=best_num_topics,
                     #num_layers=best_num_layers,
                     #num_neurons=best_num_neurons,
                     #solver=best_optimizer,
                     activation=best_activation,
                     dropout=best_dropout,
                     batch_size=128,
                     use_partitions=False,
                     embeddings_path='./word2vec/word2vec_model.bin',
                     )
```

```
# Train the model with the dataset
best_model_ETM = best_model_ETM.train_model(dataset_final)

dump(best_model_ETM, './octis/models/best_model_ETM.joblib',
compress=('lzma', 9))
best_model_ETM = load('./octis/models/best_model_ETM.joblib')

top_words_per_topic_ETM = best_model_ETM['topics']

# Display the top words for each topic
for topic_id, words in enumerate(top_words_per_topic_ETM):
    print("Topic", topic_id + 1, ":", words)

Topic 1 : ['willing', 'mistake', 'examined', 'understanding',
'pardon', 'travel', 'vivid', 'stage', 'sum', 'labour']
Topic 2 : ['remote', 'rapid', 'willing', 'sum', 'the', 'enthusiasm',
'forgive', 'obtained', 'labour', 'ice']
Topic 3 : ['sex', 'the', 'obtained', 'disappointment', 'ice',
'mistake', 'vivid', 'sum', 'forgive', 'willing']
Topic 4 : ['stage', 'sum', 'travel', 'the', 'willing', 'obtained',
'mistake', 'pardon', 'sex', 'understanding']
Topic 5 : ['mistake', 'willing', 'the', 'sum', 'endeavour',
'understanding', 'obtained', 'rapid', 'vivid', 'impatient']
Topic 6 : ['willing', 'vivid', 'stage', 'travel', 'size', 'forgive',
'remote', 'mistake', 'the', 'ice']
Topic 7 : ['willing', 'obtained', 'affections', 'mistake', 'date',
'rapid', 'examined', 'momentary', 'the', 'conducted']
Topic 8 : ['forgive', 'mistake', 'afterward', 'size', 'travel',
'willing', 'difference', 'examined', 'authority', 'rapid']
Topic 9 : ['understanding', 'forgive', 'momentary', 'willing',
'stage', 'travel', 'mistake', 'affections', 'vivid', 'the']
Topic 10 : ['sex', 'travel', 'mistake', 'coloured', 'magnificent',
'obtained', 'serve', 'afterward', 'chill', 'stage']
Topic 11 : ['stage', 'vivid', 'willing', 'mistake', 'the', 'examined',
'coloured', 'forgive', 'ice', 'affections']
Topic 12 : ['remote', 'willing', 'the', 'vivid', 'heir', 'enthusiasm',
'conducted', 'stage', 'sum', 'obtained']
Topic 13 : ['the', 'disappointment', 'vivid', 'obtained', 'forgive',
'examined', 'size', 'enthusiasm', 'travel', 'stage']
Topic 14 : ['the', 'vivid', 'willing', 'understanding', 'stage',
'travel', 'momentary', 'mistake', 'sum', 'examined']
Topic 15 : ['stage', 'travel', 'forgive', 'examined', 'willing',
'conducted', 'disappointment', 'affections', 'enthusiasm', 'mistake']
Topic 16 : ['travel', 'obtained', 'disappointment', 'forgive',
'enemies', 'mistake', 'vivid', 'genuine', 'reward', 'hearth']
Topic 17 : ['the', 'abode', 'sex', 'stage', 'forgive', 'willing',
'affections', 'lightly', 'pardon', 'travel']
Topic 18 : ['stage', 'vivid', 'mistake', 'willing', 'travel',
'enthusiasm', 'examined', 'momentary', 'size', 'pardon']
Topic 19 : ['willing', 'size', 'heir', 'stage', 'sum', 'serve',
'vivid', 'travel', 'understanding', 'the']
Topic 20 : ['willing', 'the', 'mistake', 'stage', 'enthusiasm',
'enemies', 'sum', 'conducted', 'accordingly', 'pardon']
Topic 21 : ['travel', 'forgive', 'mistake', 'stage', 'vivid', 'sum',
'coloured', 'afterward', 'ice', 'momentary']
Topic 22 : ['willing', 'disappointment', 'stage', 'affections',
'understanding', 'vivid', 'profound', 'the', 'examined', 'mistake']
Topic 23 : ['travel', 'enthusiasm', 'endeavour', 'obtained', 'guilty',
'mistake', 'pardon', 'disappointment', 'fairy', 'genuine']
Topic 24 : ['willing', 'sum', 'momentary', 'disappointment',
'examined', 'understanding', 'obtained', 'the', 'folly', 'date']
Topic 25 : ['mistake', 'affections', 'the', 'sum', 'forgive',
'understanding', 'stage', 'obtained', 'travel', 'sex']
Topic 26 : ['willing', 'the', 'vivid', 'fairy', 'momentary', 'travel',
'obtained', 'faintly', 'disappointment', 'ice']
Topic 27 : ['understanding', 'stage', 'disappointment', 'the',
'travel', 'mistake', 'examined', 'momentary', 'endeavour', 'willing']
Topic 28 : ['examined', 'sex', 'remote', 'mistake', 'conducted',
'willing', 'vivid', 'obtained', 'lightly', 'the']
Topic 29 : ['stage', 'understanding', 'sum', 'willing', 'travel',
'housekeeper', 'obtained', 'heir', 'forgive', 'size']
Topic 30 : ['enemies', 'the', 'guide', 'lightly', 'momentary',
'mistake', 'foreign', 'stage', 'vivid', 'folly']
Topic 31 : ['the', 'obtained', 'lightly', 'date', 'travel', 'sex',
'contempt', 'housekeeper', 'mistake', 'enthusiasm']
Topic 32 : ['stage', 'ice', 'willing', 'vivid', 'enthusiasm', 'the',
'disappointment', 'momentary', 'obtained', 'travel']
Topic 33 : ['the', 'understanding', 'willing', 'vivid', 'travel',
'affections', 'stage', 'disappointment', 'obtained', 'enthusiasm']
Topic 34 : ['the', 'forgive', 'willing', 'guide', 'stage',
'understanding', 'mistake', 'disappointment', 'labour', 'guilty']
Topic 35 : ['willing', 'enthusiasm', 'disappointment',
'understanding', 'stage', 'the', 'coloured', 'mistake', 'sex',
'vivid']
Topic 36 : ['the', 'travel', 'stage', 'examined', 'enthusiasm',
'vivid', 'forgive', 'mistake', 'fairy', 'inclined']
Topic 37 : ['the', 'understanding', 'mistake', 'ice', 'travel',
'size', 'double', 'sum', 'stage', 'afterward']
Topic 38 : ['understanding', 'stage', 'affections', 'forgive',
'mistake', 'willing', 'rapid', 'sum', 'vivid', 'reputation']
Topic 39 : ['willing', 'disappointment', 'the', 'stage', 'conducted',
'affections', 'forgive', 'abode', 'vivid', 'understanding']
Topic 40 : ['willing', 'the', 'mistake', 'momentary', 'stage',
'rapid', 'ice', 'obtained', 'understanding', 'disappointment']
Topic 41 : ['stage', 'enemies', 'agitation', 'willing', 'travel',
'mistake', 'examined', 'affections', 'student', 'the']
Topic 42 : ['forgive', 'mistake', 'willing', 'stage',
'disappointment', 'examined', 'understanding', 'travel', 'conducted',
'the']
```

```
Topic 43 : ['understanding', 'reputation', 'guilty', 'disappointment',
'the', 'sum', 'willing', 'guide', 'enthusiasm', 'conducted']
Topic 44 : ['mistake', 'disappointment', 'stage', 'enemies',
'fellows', 'pardon', 'understanding', 'willing', 'sex', 'affections']
Topic 45 : ['mistake', 'disappointment', 'forgive', 'vivid',
'momentary', 'stage', 'willing', 'enthusiasm', 'ice', 'chill']
Topic 46 : ['willing', 'sum', 'ice', 'the', 'affections', 'momentary',
'conducted', 'heir', 'lightly', 'mistake']
Topic 47 : ['affections', 'the', 'stage', 'understanding', 'vivid',
'momentary', 'obtained', 'sum', 'willing', 'mistake']
Topic 48 : ['understanding', 'disappointment', 'vivid', 'stage',
'agitation', 'obtained', 'the', 'ice', 'travel', 'sex']
Topic 49 : ['willing', 'stage', 'the', 'sum', 'heir', 'travel',
'understanding', 'genuine', 'mistake', 'obtained']
Topic 50 : ['understanding', 'willing', 'vivid', 'the', 'stage',
'rapid', 'travel', 'mistake', 'remote', 'obtained']
Topic 51 : ['the', 'stage', 'sum', 'vivid', 'sex', 'enthusiasm',
'travel', 'remote', 'forgive', 'willing']
Topic 52 : ['the', 'willing', 'understanding', 'conducted',
'disappointment', 'obtained', 'sum', 'travel', 'heir', 'momentary']
Topic 53 : ['travel', 'sex', 'agitation', 'obtained', 'vivid',
'understanding', 'mistake', 'pardon', 'afterward', 'affections']
Topic 54 : ['understanding', 'travel', 'vivid', 'sole', 'stage',
'obtained', 'disappointment', 'the', 'afterward', 'sum']
Topic 55 : ['willing', 'vivid', 'the', 'understanding', 'stage',
'heir', 'affections', 'mistake', 'size', 'conducted']
Topic 56 : ['willing', 'conducted', 'understanding', 'stage', 'the',
'mistake', 'affections', 'obtained', 'examined', 'rapid']
Topic 57 : ['affections', 'willing', 'sex', 'enthusiasm', 'ice',
'mistake', 'understanding', 'labour', 'abode', 'lightly']
Topic 58 : ['heir', 'obtained', 'foreign', 'fairy', 'size',
'momentary', 'forgive', 'understanding', 'willing', 'mistake']
Topic 59 : ['the', 'stage', 'travel', 'enemies', 'sex', 'heir',
'willing', 'ice', 'sum', 'obtained']
Topic 60 : ['travel', 'enemies', 'endeavour', 'size', 'stage', 'sex',
'the', 'disappointment', 'willing', 'understanding']
Topic 61 : ['stage', 'mistake', 'size', 'endeavour', 'sum', 'travel',
'enthusiasm', 'understanding', 'willing', 'ice']
Topic 62 : ['vivid', 'honourable', 'understanding', 'enemies', 'heir',
'fairy', 'enthusiasm', 'warning', 'authority', 'the']
Topic 63 : ['conducted', 'obtained', 'understanding', 'willing',
'examined', 'stage', 'vivid', 'lightly', 'affections',
'disappointment']
Topic 64 : ['mistake', 'stage', 'willing', 'travel', 'examined',
'pardon', 'rapid', 'contempt', 'lightly', 'vivid']
Topic 65 : ['mistake', 'sex', 'stage', 'obtained', 'ice', 'examined',
'the', 'pardon', 'enemies', 'understanding']
Topic 66 : ['mistake', 'represented', 'the', 'stage', 'housekeeper',
'momentary', 'labour', 'willing', 'size', 'enthusiasm']
```

```
Topic 67 : ['willing', 'agitation', 'contempt', 'prince', 'sum',
'momentary', 'vivid', 'mistake', 'stage', 'examined']
Topic 68 : ['stage', 'the', 'endeavour', 'sum', 'enthusiasm',
'willing', 'mistake', 'size', 'examined', 'vivid']
Topic 69 : ['willing', 'stage', 'disappointment', 'warning',
'obtained', 'travel', 'mistake', 'the', 'forgive', 'conducted']
Topic 70 : ['willing', 'mistake', 'guide', 'the', 'conducted',
'rapid', 'disappointment', 'travel', 'affections', 'understanding']
Topic 71 : ['vivid', 'remote', 'obtained', 'disappointment', 'the',
'understanding', 'willing', 'sex', 'endeavour', 'sole']
Topic 72 : ['willing', 'understanding', 'the', 'disappointment',
'travel', 'vivid', 'stage', 'lightly', 'size', 'remote']
Topic 73 : ['vivid', 'the', 'momentary', 'willing', 'enemies',
'obtained', 'sum', 'rapid', 'heir', 'endeavour']
Topic 74 : ['willing', 'understanding', 'stage', 'the', 'obtained',
'lightly', 'pardon', 'mistake', 'enthusiasm', 'sex']
Topic 75 : ['stage', 'the', 'sex', 'mistake', 'guilty', 'double',
'forgive', 'willing', 'obtained', 'endeavour']
Topic 76 : ['sum', 'the', 'willing', 'understanding', 'ice',
'disappointment', 'enthusiasm', 'remote', 'travel', 'heir']
Topic 77 : ['travel', 'mistake', 'vivid', 'the', 'disappointment',
'obtained', 'stage', 'ice', 'enemies', 'endeavour']
Topic 78 : ['understanding', 'sum', 'the', 'enthusiasm', 'conducted',
'guilty', 'affections', 'travel', 'labour', 'forgive']
Topic 79 : ['mistake', 'willing', 'enthusiasm', 'stage', 'size',
'the', 'vivid', 'affections', 'warning', 'disappointment']
Topic 80 : ['understanding', 'the', 'willing', 'vivid', 'mistake',
'enthusiasm', 'affections', 'stage', 'savage', 'conducted']
Topic 81 : ['willing', 'mistake', 'sum', 'fellows', 'vivid',
'understanding', 'heir', 'serve', 'lightly', 'disappointment']
Topic 82 : ['stage', 'willing', 'obtained', 'the', 'housekeeper',
'sum', 'agitation', 'disappointment', 'ice', 'conducted']
Topic 83 : ['stage', 'willing', 'mistake', 'sex', 'understanding',
'the', 'obtained', 'sum', 'enthusiasm', 'travel']
Topic 84 : ['mistake', 'travel', 'vivid', 'stage', 'examined', 'the',
'sum', 'forgive', 'pardon', 'heir']
Topic 85 : ['travel', 'prince', 'disappointment', 'fellows',
'mistake', 'stage', 'willing', 'hearth', 'sum', 'affections']
Topic 86 : ['mistake', 'travel', 'obtained', 'ice', 'foreign',
'momentary', 'the', 'stage', 'folly', 'willing']
Topic 87 : ['stage', 'willing', 'momentary', 'obtained',
'understanding', 'disappointment', 'fairy', 'sex', 'guide',
'agitation']
Topic 88 : ['the', 'willing', 'stage', 'beating', 'forgive',
'understanding', 'travel', 'endeavour', 'obtained', 'mistake']
Topic 89 : ['disappointment', 'savage', 'examined', 'understanding',
'willing', 'the', 'ice', 'vivid', 'pardon', 'enthusiasm']
Topic 90 : ['travel', 'vivid', 'ice', 'fairy', 'mistake', 'willing',
'guilty', 'rapid', 'the', 'forgive']
```

```
Topic 91 : ['willing', 'understanding', 'remote', 'mistake', 'travel',
'the', 'vivid', 'stage', 'sum', 'enemies']
Topic 92 : ['disappointment', 'examined', 'remote', 'willing', 'the',
'mistake', 'obtained', 'sex', 'understanding', 'endeavour']
Topic 93 : ['enthusiasm', 'willing', 'stage', 'sex', 'forgive', 'the',
'obtained', 'conducted', 'sum', 'lightly']
Topic 94 : ['travel', 'splendid', 'stage', 'foreign', 'mistake',
'the', 'understanding', 'size', 'coloured', 'obtained']
Topic 95 : ['travel', 'disappointment', 'sex', 'the', 'mistake',
'obtained', 'stage', 'pardon', 'guilty', 'fellows']
Topic 96 : ['mistake', 'enemies', 'travel', 'forgive', 'stage', 'the',
'size', 'vivid', 'sex', 'guide']
Topic 97 : ['mistake', 'the', 'forgive', 'stage', 'pardon', 'vivid',
'enemies', 'understanding', 'serve', 'willing']
Topic 98 : ['mistake', 'travel', 'stage', 'willing', 'understanding',
'forgive', 'rapid', 'affections', 'the', 'date']
Topic 99 : ['examined', 'understanding', 'stage', 'enemies',
'willing', 'serve', 'the', 'forgive', 'sum', 'faintly']
```

```
dump(top_words_per_topic_ETM,'./analysis/
top_words_per_topic_ETM.joblib')
```

```
['./analysis/top_words_per_topic_ETM.joblib']
```

```
topic_document_matrix_ETM = best_model_ETM["topic-document-matrix"]
topic_distribution_df_ETM =
get_document_topic_percentages(topic_document_matrix_ETM)
topic_distribution_df_ETM.head(10)
```

```
             Topic 1  Topic 2  Topic 3  Topic 4  Topic 5  Topic 6
Topic 7  \
Document ID

0               1.01     1.01     1.01     1.01     1.01     1.01
1.01
1               1.01     1.01     1.01     1.01     1.01     1.01
1.01
2               1.01     1.01     1.01     1.01     1.01     1.01
1.01
3               1.01     1.01     1.01     1.01     1.01     1.01
1.01
4               1.01     1.01     1.01     1.01     1.01     1.01
1.01
5               1.01     1.01     1.01     1.01     1.01     1.01
1.01
6               1.01     1.01     1.01     1.01     1.01     1.01
1.01
7               1.01     1.01     1.01     1.01     1.01     1.01
1.01
8               1.01     1.01     1.01     1.01     1.01     1.01
```

```
1.01
9                1.01     1.01     1.01     1.01     1.01     1.01
1.01

             Topic 8  Topic 9  Topic 10  ...  Topic 90  Topic 91
Topic 92  \
Document ID                              ...

0                1.01     1.01      1.01  ...      1.01      1.01
1.01
1                1.01     1.01      1.01  ...      1.01      1.01
1.01
2                1.01     1.01      1.01  ...      1.01      1.01
1.01
3                1.01     1.01      1.01  ...      1.01      1.01
1.01
4                1.01     1.01      1.01  ...      1.01      1.01
1.01
5                1.01     1.01      1.01  ...      1.01      1.01
1.01
6                1.01     1.01      1.01  ...      1.01      1.01
1.01
7                1.01     1.01      1.01  ...      1.01      1.01
1.01
8                1.01     1.01      1.01  ...      1.01      1.01
1.01
9                1.01     1.01      1.01  ...      1.01      1.01
1.01

             Topic 93  Topic 94  Topic 95  Topic 96  Topic 97  Topic
98  \
Document ID

0                 1.01      1.01      1.01      1.01      1.01
1.01
1                 1.01      1.01      1.01      1.01      1.01
1.01
2                 1.01      1.01      1.01      1.01      1.01
1.01
3                 1.01      1.01      1.01      1.01      1.01
1.01
4                 1.01      1.01      1.01      1.01      1.01
1.01
5                 1.01      1.01      1.01      1.01      1.01
1.01
6                 1.01      1.01      1.01      1.01      1.01
1.01
7                 1.01      1.01      1.01      1.01      1.01
1.01
8                 1.01      1.01      1.01      1.01      1.01
```

```
1.01
9                 1.01      1.01      1.01      1.01      1.01
1.01

             Topic 99
Document ID
0                1.01
1                1.01
2                1.01
3                1.01
4                1.01
5                1.01
6                1.01
7                1.01
8                1.01
9                1.01

[10 rows x 99 columns]

original_indexes = dataset_final._Dataset__original_indexes
topic_distribution_df_ETM['Original Document Index'] =
original_indexes

df_input['input_index'] = df_input.index
df_input['ref'] = df_input['reference'].apply(lambda x: x.rsplit('_',
1)[0])
df_merge = df_input.merge(df_ref, left_on='ref', right_on='reference',
how='left')
df_merge['date'] = df_merge['date'].astype('Int64')
df_merge['birthdate'] = df_merge['birthdate'].astype('Int64')
df_merge.head()

                                 preprocessed_text_x
reference_x  \
0  fragment adventure turned steed hoping cross d...
Aikin_SirBertran_1
1  plague portion ensuing relating street manner ...
Ainsworth_OldSaintPa_1
2  whatsoever away terms included language charac...
Ainsworth_Rookwood_1
3  doll wangos leaving justice skill witches spea...
Ainsworth_TheLancash_1
4  note text little work finished year intended i...
Austen_Northanger_1

   sentiment  input_index                   ref  index
reference_y  \
0    -0.9201            0      Aikin_SirBertran      0
Aikin_SirBertran
1     0.9773            1  Ainsworth_OldSaintPa      1
```

```
Ainsworth_OldSaintPa
2     0.9989            2    Ainsworth_Rookwood      2
Ainsworth_Rookwood
3    -0.9998            3  Ainsworth_TheLancash      3
Ainsworth_TheLancash
4     1.0000            4     Austen_Northanger      5
Austen_Northanger

                                               title  \
0                           Sir Bertrand, A Fragment
1  Old Saint Paul's: A Tale of the Plague and the...
2                                           Rookwood
3  The Lancashire Witches: A Romance of Pendle Fo...
4                                   Northanger Abbey

                          author  date  ...      mode   genre
gender  \
0  Aikin, John and Anna Laetitia  1773  ...  Fragment  Gothic    NaN

1    Ainsworth, William Harrison  1841  ...       NaN     NaN      m

2    Ainsworth, William Harrison  1834  ...     Novel  Gothic      m

3    Ainsworth, William Harrison  1848  ...       NaN     NaN      m

4                   Austen, Jane  1817  ...       NaN     NaN      f


  birthdate  nationality role (central/peripheral/influence)  \
0      <NA>      English                             Central
1      1805      English                                 NaN
2      1805      English                             Central
3      1805      English                                 NaN
4      1775      English                                 NaN

                                                text     source  \
0  SIR BERTRAND, A FRAGMENT:\n\nAFTER this advent...     colors
1  OLD SAINT PAUL\'S\n\n  _A TALE OF THE PLAGUE\n...  pb-manual
2  \nThe Project Gutenberg EBook of Rookwood, by ...     colors
3  Proofreading Team.\n\n\n\n\n\n\n[Illustration:...  pb-manual
4  Northanger Abbey\n\n\nby Jane Austen\n\n(1803)...  gutenberg

                                 preprocessed_text_y  \
0  fragment adventure turned steed hoping cross d...
1  plague portion ensuing relating street manner ...
2  whatsoever away terms included language charac...
3  doll wangos leaving justice skill witches spea...
4  note text little work finished year intended i...

                                      tokenized_text
```

```
0  ['fragment', 'adventure', 'turned', 'steed', '...
1  ['plague', 'portion', 'ensuing', 'relating', '...
2  ['whatsoever', 'away', 'terms', 'included', 'l...
3  ['doll', 'wangos', 'leaving', 'justice', 'skil...
4  ['note', 'text', 'little', 'work', 'finished',...

[5 rows x 21 columns]
```

```
# merging of topic distribution with features
# reorganizing the order of columns and clean up
df_txt_features_ETM = df_merge.merge(topic_distribution_df_ETM,
right_on='Original Document Index', left_on='input_index')
df_txt_features_ETM=df_txt_features_ETM.drop(['text',
'preprocessed_text_y','tokenized_text','preprocessed_text_x', 'index',
'ref', 'Original Document Index'], axis=1)
df_txt_features_ETM.rename(columns={'reference_x':
'reference','reference_y': 'text_key'}, inplace=True)
df_txt_features_ETM = df_txt_features_ETM[['input_index'] + [col for
col in df_txt_features_ETM.columns if col != 'input_index']]
df_txt_features_ETM.rename(columns={'role
(central/peripheral/influence)': 'role'}, inplace=True)
df_txt_features_ETM.head()
```

```
   input_index               reference  sentiment
text_key  \
0            0      Aikin_SirBertran_1    -0.9201
Aikin_SirBertran
1            1  Ainsworth_OldSaintPa_1     0.9773
Ainsworth_OldSaintPa
2            2    Ainsworth_Rookwood_1     0.9989
Ainsworth_Rookwood
3            3  Ainsworth_TheLancash_1    -0.9998
Ainsworth_TheLancash
4            4     Austen_Northanger_1     1.0000
Austen_Northanger

                                               title  \
0                            Sir Bertrand, A Fragment
1  Old Saint Paul's: A Tale of the Plague and the...
2                                            Rookwood
3  The Lancashire Witches: A Romance of Pendle Fo...
4                                    Northanger Abbey

                          author  date    period      mode
genre  ...  \
0  Aikin, John and Anna Laetitia  1773  Romantic  Fragment
Gothic  ...
1    Ainsworth, William Harrison  1841       NaN       NaN
NaN  ...
2    Ainsworth, William Harrison  1834       NaN     Novel
```

```
Gothic  ...
3    Ainsworth, William Harrison  1848       NaN       NaN
NaN  ...
4                  Austen, Jane  1817       NaN       NaN
NaN  ...

  Topic 90  Topic 91 Topic 92 Topic 93 Topic 94  Topic 95  Topic 96
Topic 97  \
0     1.01      1.01     1.01     1.01     1.01      1.01      1.01
1.01
1     1.01      1.01     1.01     1.01     1.01      1.01      1.01
1.01
2     1.01      1.01     1.01     1.01     1.01      1.01      1.01
1.01
3     1.01      1.01     1.01     1.01     1.01      1.01      1.01
1.01
4     1.01      1.01     1.01     1.01     1.01      1.01      1.01
1.01

   Topic 98  Topic 99
0      1.01      1.01
1      1.01      1.01
2      1.01      1.01
3      1.01      1.01
4      1.01      1.01

[5 rows x 114 columns]
```

```
df_txt_features_ETM.to_csv('./analysis/df_txt_features_ETM.csv',
index=False)
#df_txt_features_ETM=pd.read_csv('./analysis/df_txt_features_ETM.csv')
```

pyLDAvis exports

```
topic_term_dists_ETM = best_model_ETM["topic-word-matrix"]
doc_topic_dists_ETM = best_model_ETM["topic-document-matrix"]
doc_topic_dists_ETM = doc_topic_dists_ETM.T

dump(topic_term_dists_ETM, './analysis/topic_term_dists_ETM.joblib')
dump(doc_topic_dists_ETM, './analysis/doc_topic_dists_ETM.joblib')
```

```
['./analysis/doc_topic_dists_ETM.joblib']
```