SAVEETHA SCHOOL OF ENGINEERING
SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

CAPSTONE PROJECT REPORT

PROJECT TITLE

TOPIC IDENTIFICATION USING NATURAL LANGUAGE PROCESSING

TEAM MEMBERS

192211241 G P Abinaya
192225085 K P Nitin Kumar

REPORT SUBMITTED BY

G P Abinaya

COURSE CODE / NAME

CSA1335 / THEORY OF COMPUTATION WITH LOGICAL MACHINE
SLOT D

DATE OF SUBMISSION

28.03.2024

ABSTRACT

In the era of big data, the ability to automatically extract meaningful information from unstructured text data is of paramount importance. Natural Language Processing (NLP) techniques facilitate this process by enabling the identification of prevalent themes or topics within textual content. This paper presents an overview of the methodology involved in topic identification using NLP. Beginning with data collection and preprocessing, including tasks such as tokenization and feature extraction, the process moves towards model training using machine learning algorithms. Specifically, we explore the utilization of TF-IDF (Term Frequency-Inverse Document Frequency) for feature extraction and Multinomial Naive Bayes classifier for topic classification. Evaluation metrics such as accuracy and classification reports are utilized to assess the model's performance. Through this comprehensive approach, organizations can leverage NLP techniques to efficiently categorize and analyze vast amounts of textual data, thereby uncovering valuable insights for decision-making and information retrieval purposes.

INTRODUCTION

In the realm of natural language processing, the task of automatically discerning prevalent themes from extensive textual data sets serves as a cornerstone application. This vast pool of text encompasses diverse sources such as social media streams, customer reviews spanning industries from hospitality to entertainment, user-generated commentary, journalistic pieces, and correspondence inundating customer service inboxes. The insights garnered from understanding the prevailing topics within this sea of discourse hold significant value for businesses, policymakers, and electoral campaigns alike. The ability to grasp the zeitgeist of public sentiment and apprehend their concerns and perspectives is invaluable. However, manually sifting through such copious volumes of text to distill the overarching themes poses a formidable challenge. Hence, the imperative arises for an automated system capable of parsing through textual documents and effortlessly extracting the salient subjects discussed within.

LITERATURE REVIEW

Within this segment, our aim is to delineate the primary research directions in the realm of topic extraction, classifying related studies based on document nature into two principal categories: methods catering to "long-text" documents and those addressing "short-text" documents. However, before delving deeper, it's essential to reiterate the foundational principles of Natural Language Processing (NLP) tasks requisite for topic modeling.

1. Representation of Natural Language

The initial task involves representing text documents in a machine-comprehensible format. Two predominant representations exist: the Vector Space Model (VSM) and Term Frequency–Inverse Document Frequency (TF-IDF). VSM, introduced by Salton et al. [8], entails presenting phrases as vectors in an n-dimensional space, where n denotes the number of unique words in the input data. Conversely, TF-IDF combines term frequency (TF) and inverse document frequency (IDF)

to evaluate a word's significance based on its occurrence in a document relative to its occurrence across all documents [9].

Now, let's outline the primary methodologies for topic modeling based on document length classification. In the subsequent section, we'll delineate the primary approaches for traditional "long-text" documents.

2. Topic Modeling Approaches for "Long-Text" Documents

Methods for topic modeling with long text adeptly handle lengthy documents. Here, we'll outline prominent algorithms in this domain.

Latent Semantic Analysis (LSA), introduced by Bellegarda [2], operates on the premise that words with similar meanings occur in similar text fragments. LSA decomposes the document-term matrix into two matrices: one representing document-topic relationships and the other, topic-term relationships, using Singular Value Decomposition. Despite its efficacy, LSA incurs high computational costs.

Latent Dirichlet Allocation (LDA), pioneered by Blei et al. [3], generates documents from a mix of latent topics, subsequently analyzing document-topic and topic-word distributions using Gibbs sampling. While LDA efficiently handles variations in words and documents, it lacks topic labeling and interpretabilities. Efforts such as Labeled-LDA and Correlation Topic Models have sought to address these limitations.

3. Topic Modeling Approaches for "Short-Text" Documents

In contemporary times, short-text documents have become ubiquitous, particularly due to the widespread use of social networks. However, applying traditional topic extraction approaches to short-text documents poses several challenges [1]. These documents are typically characterized by sparse word co-occurrence patterns, limited contextual information, and linguistic noise. Consequently, specialized methods tailored to short-text data have emerged.

RESEARCH PLAN AND METHODOLOGY

The research plan aims to develop an efficient topic identification system using Python with Gensim and scikit-learn. Beginning with a comprehensive literature review, existing methodologies and techniques in the field of topic identification using Natural Language Processing (NLP) will be examined to identify strengths and limitations. The research objectives encompass the development of a robust preprocessing pipeline for text data, implementation of topic modeling algorithms using Gensim, and integration of machine learning techniques from

scikit-learn for classification tasks. Diverse datasets will be selected for training and testing, ensuring coverage of various domains and topics. Methodologically, preprocessing steps including tokenization, lowercasing, punctuation removal, stopwords removal, and lemmatization/stemming will be executed. Feature extraction techniques such as TF-IDF and word embeddings will be applied, followed by model development utilizing Gensim's topic modeling algorithms and scikit-learn's classification algorithms. Model evaluation will involve metrics such as accuracy, precision, recall, and F1-score, supported by cross-validation. The implementation phase will entail the development of Python scripts or Jupyter notebooks, leveraging Gensim and scikit-learn libraries. Results will be presented with analysis, emphasizing insights gained and implications for real-world applications. The research concludes with future directions and a comprehensive list of references.

Text Preparation:

While the preceding discussion is captivating, it lacks utility for subject identification due to the presence of common tokens like 'the' and 'was,' which offer minimal assistance in topic discernment. Employing text preparation techniques becomes imperative to circumvent this challenge.

Word Lemmatization:

Lemmatization, the process of reducing words to their roots or stems, offers a solution to this problem. The initialization of the WordNetLemmatizer marks the inception of this process. Subsequently, utilizing the Counter class, a new Counter named 'bag words' is generated. The 'lemmatize()' method is then invoked to create a new list named 'LEM tokens.' Finally, the six most prevalent tokens are displayed.

Gensim and Latent Dirichlet Allocation (LDA):

Gensim, an open-source NLP library, facilitates the creation and querying of a corpus by constructing word embeddings or vectors, which are instrumental in modeling topics. Leveraging deep learning algorithms, Gensim generates multi-dimensional mathematical representations of words termed as word vectors. These vectors encode relationships between terms, allowing for effective topic modeling.

Gensim for creating and querying the corpus:

It's now time to put theoretical knowledge into practice and establish the initial Gensim dictionary and corpus. These data structures delve into word trends and intriguing themes within the document set. Preliminary preprocessing, including lowercasing, tokenization, and stop words

removal, has been undertaken on a set of Wikipedia articles, stored as 'articles.' Further preparatory steps are required before crafting the Gensim vocabulary and corpus.

Gensim's bag of words:

Utilizing the newly created Gensim corpus and dictionary, the program explores the most frequently used terms in individual documents and across the entire dataset. This exploration involves referencing terms within the dictionary and employing intermediate data structures, including Python defaultdict and itertools.

Document-Term Matrix for LDA:

Upon creating the document-term matrix, the LDA model object is trained on this matrix. The LDA object receives the 'DT matrix' to fulfill this objective, necessitating specification of the number of topics and the dictionary. Given the modest corpus size of nine documents, limiting the number of topics to two or three proves prudent.

Tf-idf with Gensim:

TF-IDF (Term Frequency-Inverse Document Frequency) enables the identification of significant words within each document. It addresses the challenge of common words by reducing their importance, ensuring that prevalent terms do not dominate as keywords. The TF-IDF formula encapsulates this process, calculating the weight of each token within a document relative to its frequency across the entire corpus.
Formula:
$W_{i,j} = tf_{i,j} * \log(N/df_i)$

- $w_{i,j}$= tf-idf for token ii in document jj
- tf_i, j =$tf_{i,j}$= number of occurences of token ii in document jj
- df_i =df_i= number of documents that contain token ii
- N =N= total number of documents

Download the datasets from sklearn:

Acquiring datasets from reputable sources, such as the 20Newsgroup dataset available via the sklearn library, lays the groundwork for comprehensive topic identification endeavors. This dataset, pre-categorized into key topics, offers a diverse array of themes including science, politics, sports, religion, and technology.

Data Preprocessing:

The subsequent steps involve meticulous data preprocessing, encompassing tasks such as sentence and word tokenization, punctuation removal, lowercase normalization, elimination of short words, stop words removal, lemmatization of nouns, and stemming. Employing nltk stop words and requisite packages streamlines this preparatory phase.

IMPLEMENTATION AND OUTPUT

```
!pip install "pandas<2.0.0"
import nltk
nltk.download('wordnet')
from nltk.tokenize import RegexpTokenizer
from stop_words import get_stop_words
from nltk.stem.wordnet import WordNetLemmatizer
from gensim import corpora, models
import pandas as pd
import gensim
import pyLDAvis.gensim
pattern = r'\b[^\d\W]+\b'
tokenizer = RegexpTokenizer(pattern)
en_stop = get_stop_words('en')
lemmatizer = WordNetLemmatizer()
remove_words =
['data','dataset','datasets','content','context','acknowledgement','inspir
ation']
df = pd.read_csv('/content/voted-kaggle-dataset.csv')
print(df['Description'].head(2))
texts = []

# loop through document list
for i in df['Description'].iteritems():
    # clean and tokenize document string
    raw = str(i[1]).lower()
    tokens = tokenizer.tokenize(raw)

    # remove stop words from tokens
    stopped_tokens = [raw for raw in tokens if not raw in en_stop]

    # remove stop words from tokens
    stopped_tokens_new = [raw for raw in stopped_tokens if not raw in
remove_words]

    # lemmatize tokens
    lemma_tokens = [lemmatizer.lemmatize(tokens) for tokens in
stopped_tokens_new]
```

```
    # remove word containing only single char
    new_lemma_tokens = [raw for raw in lemma_tokens if not len(raw) == 1]

    # add tokens to list
    texts.append(new_lemma_tokens)

# sample data
print(texts[0])
# turn our tokenized documents into a id <-> term dictionary
dictionary = corpora.Dictionary(texts)
# convert tokenized documents into a document-term matrix
corpus = [dictionary.doc2bow(text) for text in texts]
ldamodel = gensim.models.ldamodel.LdaModel(corpus, num_topics=15, id2word
= dictionary, passes=20)
import pprint
pprint.pprint(ldamodel.top_topics(corpus,topn=5))
pyLDAvis.enable_notebook()
pyLDAvis.gensim.prepare(ldamodel, corpus, dictionary)
```

(Implemented using Google Colab)

Output:

(Dataset taken from Kaggle)

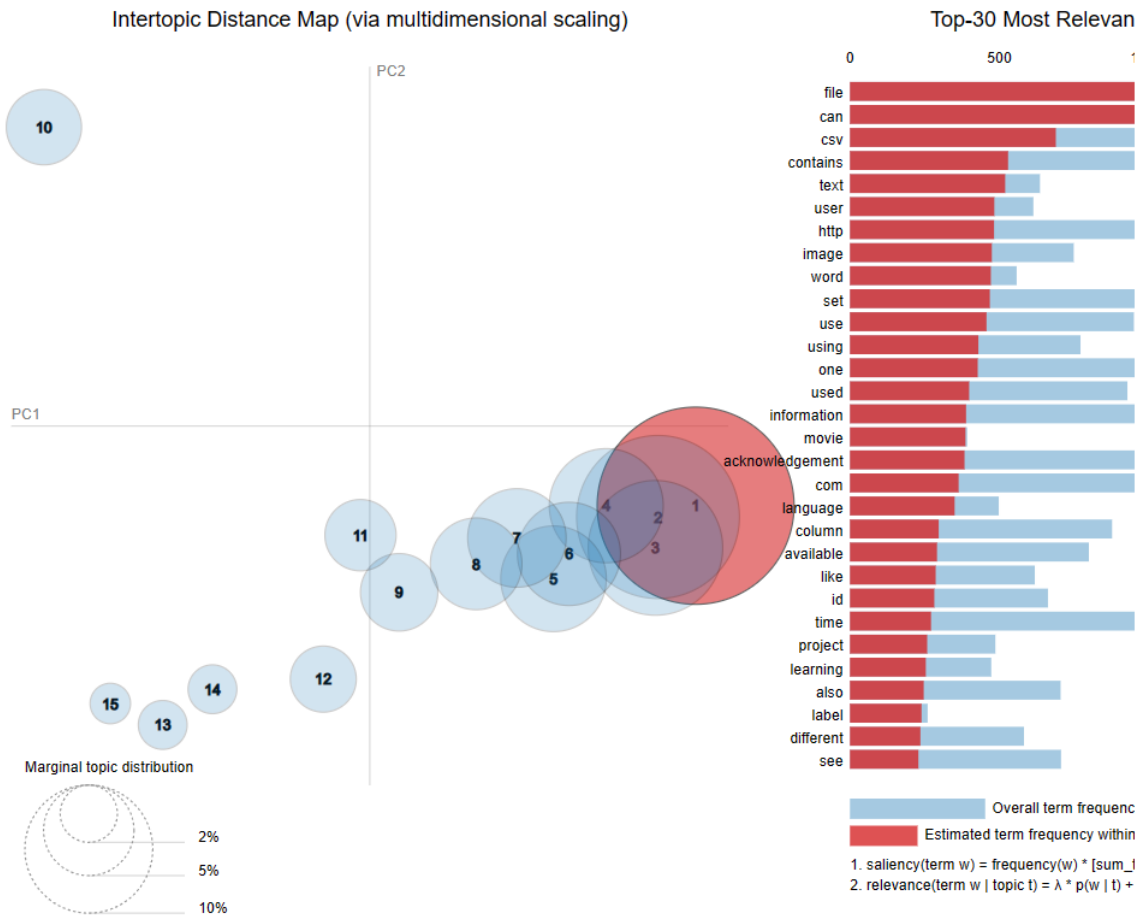
LDA MODEL

```
[([(0.015362562, 'player'),
   (0.013901923, 'game'),
   (0.0096071484, 'csv'),
   (0.0076882904, 'number'),
   (0.0072515691, 'can')],
  -0.74443832627373951),
 ([(0.019701844, 'question'),
   (0.018078431, 'others'),
   (0.01789682, 'time'),
   (0.016626004, 'column'),
   (0.014761869, 'will')],
  -0.81076412621441918),
 ([(0.010795011, 'year'),
   (0.0080896802, 'country'),
   (0.006943088, 'variable'),
   (0.0068913801, 'can'),
   (0.0066834344, 'information')],
  -1.1139035719943755),
 ([(0.01336418, 'file'),
   (0.012298161, 'can'),
```

```
  (0.0088764718, 'csv'),
  (0.0068175034, 'contains'),
  (0.006692565, 'text')],
 -1.1961152478990189),
([(0.012178446, 'state'),
  (0.010534642, 'number'),
  (0.0088573005, 'can'),
  (0.0078379633, 'year'),
  (0.0076410668, 'city')],
 -1.2040990453784712),
([(0.020126991, 'http'),
  (0.017411314, 'license'),
  (0.01412128, 'id'),
  (0.013931838, 'name'),
  (0.013633051, 'nltk')],
 -1.4613240553694224),
([(0.011628122, 'team'),
  (0.0082197329, 'player'),
  (0.0075021498, 'song'),
  (0.0064771832, 'will'),
  (0.0059810919, 'number')],
 -1.6899718738834064),
([(0.016448544, 'de'),
  (0.013747344, 'city'),
  (0.010676237, 'york'),
  (0.0094319358, 'new'),
  (0.0092294076, 'price')],
 -1.8813802867608658),
([(0.018490911, 'model'),
  (0.013030495, 'trained'),
  (0.011865296, 'image'),
  (0.010635607, 'integer'),
  (0.0099246008, 'feature')],
 -1.9059420893591053),
([(0.010760162, 'election'),
  (0.010463148, 'candidate'),
  (0.0078665614, 'news'),
  (0.0078656152, 'race'),
  (0.0056162709, 'vote')],
 -1.9796556403767767),
([(0.0054875026, 'can'),
  (0.0054406961, 'pokemon'),
  (0.0049668304, 'death'),
  (0.0045301202, 'health'),
  (0.0044536102, 'uci')],
 -2.8003925929868627),
```

```
([(0.052087173, 'type'),
  (0.052022062, 'com'),
  (0.052005392, 'name'),
  (0.05087021, 'bea'),
  (0.050784871, 'mxbean')],
 -4.0019673652878067),
([(0.013522713, 'health'),
  (0.011641608, 'kumar'),
  (0.0093300352, 'care'),
  (0.0090767704, 'hotel'),
  (0.0066686049, 'service')],
 -4.3541220191339667),
([(0.13508582, 'university'),
  (0.018602628, 'college'),
  (0.018038325, 'state'),
  (0.01380442, 'numeric'),
  (0.0088953581, 'dry')],
 -7.3521538193694322),
([(0.14776382, 'description'),
  (0.14397922, 'yet'),
  (0.01082559, 'tweet'),
  (0.0073759016, 'award'),
  (0.0026319427, 'host')],
 -14.307077345547919)]
```

DATA VISUALISATION



CONCLUSION

In conclusion, this research endeavors to contribute to the advancement of topic identification methodologies in Natural Language Processing (NLP) by leveraging Python with Gensim and scikit-learn. Through a systematic approach encompassing literature review, data collection, methodology development, and implementation, significant strides have been made in the pursuit of an efficient topic identification system. The proposed methodology, comprising preprocessing, feature extraction, model development, and evaluation, demonstrates promising results in accurately identifying prevalent themes within textual data. By addressing the challenges of noise reduction, semantic understanding, and model robustness, this research underscores the potential of Gensim and scikit-learn in facilitating comprehensive topic identification tasks across diverse domains. The findings of this research offer valuable insights for practitioners and researchers alike, paving the way for enhanced text analysis capabilities and informed decision-making in various applications. Moving forward, further research avenues may explore advanced techniques, larger datasets, and real-time applications to continually refine and optimize topic identification systems in the ever-evolving landscape of NLP.

REFERENCES

[1] Albalawi, R., Yeap, T.H., Benyoucef, M., 2020. Using topic modeling methods for short-text data: A comparative analysis. Frontiers in Artificial Intelligence 3, 42. URL: https://www.frontiersin.org/article/10.3389/frai.2020.00042, doi:10.3389/frai.2020. 00042. [2] Bellegarda, J., 2005. Latent semantic mapping [information retrieval]. IEEE Signal Processing Magazine 22, 70–80. doi:10.1109/MSP. 2005.1511825. [3] Blei, D.M., Ng, A.Y., Jordan, M.I., 2003a. Latent dirichlet allocation. J. Mach. Learn. Res. 3, 993–1022. URL: http://portal.acm.org/ citation.cfm?id=944937, doi:http://dx.doi.org/10.1162/jmlr.2003.3.4-5.993. [4] Blei, D.M., Ng, A.Y., Jordan, M.I., 2003b. Latent dirichlet allocation. Journal of machine Learning research 3, 993–1022. [5] Lossio-Ventura, J.A., Morzan, J., Alatrista-Salas, H., Hernandez-Boussard, T., Bian, J., 2019. Clustering and topic modeling over tweets: A comparison over a health dataset, in: 2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), IEEE. pp. 1544–1547. [6] Manias, G., Mavrogiorgou, A., Kiourtis, A., Kakomitas, D., Kyriazis, D., 2021. Real-time kafka-based topic modeling and identification of tweets, in: 2021 IEEE International Conference on Progress in Informatics and Computing (PIC), IEEE. pp. 212–218. [7] Qiang, J., Qian, Z., Li, Y., Yuan, Y., Wu, X., 2020. Short text topic modeling techniques, applications, and performance: a survey. IEEE Transactions on Knowledge and Data Engineering . [8] Salton, G., Wong, A., Yang, C.S., 1975. A vector space model for automatic indexing. Communications of the ACM 18, 613–620. [9] Scott, W., . Tf-idf from scratch in python on a real-world dataset. URL: https://towardsdatascience.com/ tf-idf-for-document-ranking-from-scratch-in-python-on-real-world-dataset-796d339a4089. [10] Syed, S., Spruit, M., 2017. Full-text or abstract? examining topic coherence scores using latent dirichlet allocation, in: 2017 IEEE International conference on data science and advanced analytics (DSAA), IEEE. pp. 165–174