Spring 5-20-2019

TOPIC CLASSIFICATION USING HYBRID OF UNSUPERVISED AND SUPERVISED LEARNING

Jayant Shelke
San Jose State University

TOPIC CLASSIFICATION USING HYBRID OF UNSUPERVISED AND SUPERVISED LEARNING

A Project

Presented to

Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science

by

Jayant Shelke

May 2019

The Designated Project Committee Approves the Project Titled

Topic Classification Using Hybrid of Unsupervised and Supervised Learning

by

Jayant Shelke

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

MAY 2019

Dr. Mike Wu	Department of Computer Science
Dr. Robert Chun	Department of Computer Science
Dr. Katerina Potika	Department of Computer Science

Abstract

There has been research around the idea of representing words in text as vectors and many models proposed that vary in performance as well as applications. Text processing is used for content recommendation, sentiment analysis, plagiarism detection, content creation, language translation, etc. to name a few. Specifically, we want to look at the problem of topic detection in text content of articles/blogs/summaries. With the humungous amount of text content published each and every minute on the internet, it is imperative that we have very good algorithms and approaches to analyze all the content and be able to classify most of it with high confidence for further use.

The project aims to work with unsupervised and supervised machine learning algorithms in an effort to tackle the topic detection problem. The project will target various unsupervised learning algorithms like Word2vec, doc2vec and LDA for corpus and language dictionary learning to have a trained model which understand the semantic of texts. The objective of the project is to combine this unsupervised learning with supervised learning algorithms like Support Vector Machine and deep learning methods to analyze and hopefully better the performance in terms of accuracy of topic detection. The project also aims at performing user interest-based modelling, which is orthogonal to topics modeling. The idea is to make sure the model is free of predefined categories.

The project results show that hybrid models are comfortably accurate when classifying text in a particular topic category. The project also concludes that user interest modelling can also be accurately achieved along with topic detection. The project successfully determines these results without any meta information about the input text and purely based on the corpus of the input text. This makes the project framework really robust as it has no dependency on source of text, length of text or any other meta information about the text content.

Index Terms – **topic detection, topic modelling, hybrid, topic mixtures, SVM, neural network, doc2vec, LDA.**

Acknowledgments

I would like to express my sincere gratitude to my project advisor, Dr. Mike Wu, who precisely guided me through my master project providing valuable guidance and keeping my efforts on the right track. I would like to thank him for his guidance in helping me complete this project.

I would also like to express my gratitude to Dr. Robert Chun and Dr. Katerina Potika for being on my project committee and providing with valuable feedback that has immensely helped the project.

Last but not the least, I would like to thank my parents, Gajanan and Sunita, for providing me with the opportunity of undertaking higher education and for making me who I am today. It would not have been possible without your support and constant encouragement.

Table of Contents

List of Figures

List of Tables

I. Introduction

Some of the conventional models used for topic detection are text to vector models like LDA, LSA, PLSA, doc2vec, etc. Latent Dirichlet Allocation is one of the first models to create topics out of word vector models and understand the thematic patterns within the text. It builds a mixture of topics per document and words per topic in the training phase. Some other approaches which are used for classification problems singularly or in combination with other techniques are Support Vector Machines, K-Means, Random Forest, etc. When used in ensemble approach, the applications of these classification techniques range from classifying parts of face in face detection, text and hypertext categorization, bioinformatics, protein fold and remote homology detection, handwriting recognition, etc.

In this project, the primary objective is to create a model which uses unsupervised and supervised learning in conjunction for modelling topic detection, user interest, content flagging/prohibition, etc. This project will use an unsupervised model to create a learning corpus and generate vector which could act as input to the supervised models and validate if this setup can give results which help us model the previously mentioned ideas.

We take a look at a few researches done previously in the “Related Work” section which will help us form a strong foundation for this project which will aim at combining supervised and unsupervised learning algorithms to model topic detection solution for various datasets.

A. Background

Across the internet, there is tons of user content that is generated every day in the form of audio, video, images, text, etc. Content curation as well as recommendations is one of the core problems for today's online content providers. If we look at the text data, there are many cases where users subscribe to newsletters, blogs, promotions, updates, etc. to stay on top of a certain field or area of their interest. One other case is search engines using user search patterns and history to give more relevant and suitable suggestions for articles/blogs that might suit their interest or what the user is looking for specifically. Some other cases of content curation are when users mark a set of topic categories as their 'interests' for recommendation engines and these engines then provide them with the relevant content. All these cases show that content curation, topic modelling and text recommendation are things that are really crucial in today's world.

However, there are some issues with the current topic modelling and recommendation systems such as –

- Content whose topics are not known or newly added topics.
- Topics of the content is not present in the explicitly curated topic categories.
- Topic categories are modelled based on a specific set of sources.
- Inter topic or intermingling topic interests are not modelled well.
- Topic categories are named to be either too broad or too specific causing problem of too strict or very loose boundaries for content recommendation.

These issues cause the users to lose trust on the recommendation or finding of the search engine since they might end up eyeballing articles which end up being of no interest or remote interest to them. One of the common cases where this happens a lot is with mingling interests.

For example. If a user has marked 2 topics of interest in a certain recommendation system, namely Sports and Machine Learning. When there is an article which talks about the use of machine learning tools to improve the facial features of the players in the console game FIFA, this article is missed because it is neither primarily tagged as 'Machine Learning' nor as 'Sports' but as 'Console Games'. This is marginal side track from the user's marked interests but it is still a combination of both the interests and so should be recommended to the user and not missed out. Such inter mingling interest categories are a problem for the topic modelled systems in place today and there needs to be a better way to handle this.

Another issue with the same example could be thought of. Suppose the article was published by the company which manufactures 'console games' and it was tagged as a gaming article based on the source of the article. The user has no mention of games in his topics of interest, but a core topic of interest is 'Machine Learning'. Since the article talks about the use of machine learning in improvising facial features of the graphically animated players, the user should be recommended this article as a topic of interest, however, this is missed since the topic of interest is tightly coupled in deriving meaning from the source of the article rather than the content.

Topic categories can also be broad or too narrow. A topic category like 'Global news' is too wide and will recommend anything and everything, whereas, a topic category like 'Nvidia GE Force 4500' is too specific and will hardly curate any results or will keep showing the stale results again and again. This is why topic categorization is a critical problem and designing topic categories is very crucial in having a good content curation strategy in place. The topics should be

designed in a way that leave some room for gaining breadth around the topic content and not be so broad that the area of surface for the topic overlaps with many other topics.

Some other cases of topic modelling can be used in bizarre cases where there may be objectionable content on the internet being subscribed and shared which is brought down once the topic category is noticed. However, the menace can still continue if the same content is posted with misleading topic categories since the topic categories could be derived based on the source or a manual curator who is also of the same objectionable view. This can only be solved if we have a way of detecting topics of an article based on the content itself.

One other issue occurs if the topic recommended to the user does contain the area of interest, but it is relatively very less compared to the whole content of the article making it a bad choice for recommendation. A good example would be an article about launch of a new 'Dominos' store, the pizza chain, which talks about its new recipes and ingredients and gets recommended to the user since his topic of interest was 'San Francisco'.

These are some of the issues that we have around topic modelling or user interest modelling for textual data.

B. Problem Definition

Topic modelling and classification being a really integral part of content curation and content recommendation needs to be orthogonal to the parameters that are surrounding the content. Topic modelling based on meta features about the content like length of the text, language of the text, source of the text, etc. should be avoided since they give rise to all the issues that we saw in the previous sections.

To be able to evolve from these problems, we suggest the basic rules that our proposed topic/user interest modelling framework should follow. We should propose a framework:

- that is combined result of supervised and unsupervised algorithm.
- that works well with intermingling categories of interests
- that is not rigid and strongly coupled to predefined categories
- that predicts a set of topics for the given content
- that predicts the contribution of each predicted topic to the article
- that generalizes well for unseen categories.
- that does not depend on the source of information for context
- that purely predicts based on the text content and not based on meta features like size, length, headline, etc.

We propose a framework that a combination of supervised and unsupervised machine learning techniques that will follow all the above-mentioned rules for topic modelling. This will help us eliminate the problems with previous topic modelling techniques and create a better version of content curation and recommendation engines.

Some of the previous work done around content topic modelling can be seen in the next section ‘Related Work’.

II. Related Work

LDA has been used with HMM [4] and genetic algorithms [12] to approach topic modelling in innovative ways. One other approach to topic modelling was proposed in [5] where no topic labels were previously known, and the model performed really well for articles from Wikipedia. The topic word clusters that were created in this approach were very similar to what was mentioned in the Wikipedia topic categories. One other novel approach including LDA was [6] which used online stream processing model on top of LDA which gave better results in terms of adapting to change in topic vectors with time over the course of the stream.

Blei et. al [1] proposed the base Latent Dirichlet allocation model which is so widely used. It is a probability generated model which uses Dirichlet allocation to project a document into vector space with many topics where each topic consists of many words representing the topic. LDA is an extension to Probability Latent Semantic Analysis [PLSA] proposed by Hoffman [2].

LDA model's [3] understanding of topics is implicit and so AlSumait et.al [6] proposed to model a genetic algorithm on top of LDA to help with the topic classification in a more meaningful way. [6] proposed to model the genetic algorithm in a way that it optimizes the weights of the topics which are stored in the θ matrix of the LDA model. It was seen that results for the individual topic classification out of the identified 8 categories of topics as compared to the previous studies were improved by LDA-GA.

One other variation of a topic model is the generative type. In this approach, the model is able to uncover the different themes within a topic and different topics within a topic. An innovative use of this generative behavior was proposed by Kaur et. al [4] where Hidden Markov Model [HMM] was used on top of plain LDA to spotlight on extricate words from the topic with length of more than one word. This used a level 4 Bayesian model and was proposed by the name of Latent Dirichlet Markov Allocation [LDMA]. The core functionality which forms the basis of LDMA is the extraction of unigrams from LDA and then creating a markov chain process which will generate further topic links. The topic links could be unigrams/bigrams which in turn allows for multiword topic generation capturing the idea in a more robust way. This could also help with the problem of having labelling generated topic vectors since it is easier to get more meaningful topics for ideas which are expressed in the data text itself and not latent in expression. SentiNetWord was used to analyze the sentiment of the text and give a stochastic probability value for 3 parameters - positive, negative, and neutral/objective. These values were also used as a feature to decide on the final topic selection from the generated list of unigram/bigram topics by looking at the synonyms of the lexical tokens given by SentiWordNet.

Wartena et. al. [5] had proposed one of the first clustering approach based on keywords in alignment with the distribution of co-occurring keywords of a text. In this research, they did not use any labels of topic classifications for training the dataset which was created from a host of Wikipedia articles. Instead, the documents were used to create word frequencies and then a similarity measure was developed based on the cosine similarity between the keyword. It was also observed that the Jensen-Shannon divergence of probability distribution performs better than the cosine similarity measure for keyword similarity. This research defines likelihood of words defined as distributions associated with words by measuring the co-occurrences in documents. It

must be noted that this paper treats bag of words or terms as a word which means that similarity between words could means similarity between a window of bag of words. While evaluating the performance of different distance measures for similarity - 4 main types of distances are considered:

- Cosine similarity for document distributions
- Cosine similarity of vectors of tf-idf values of words
- Jensen-Shannon divergence between document distributions
- Jensen-Shannon divergence between term/word distributions

The clustering mechanism used is pretty straight forward. Start by selecting 2 farthest of the data points. Assign each of the points based on proximity of the first 2 points as seeds for the clustering. Once all points are clustered and if the radius of individual clusters is more than a predefined threshold value, we repeat this process within that radius for the inner points. This gives a binary tree of clusters. The paper also looks at agglomerative hierarchical clustering.

The results of the research clearly state that the Jensen-Shannon term distribution provides the best results with document distributions of same flavor a close second. Thus, it was clear that for clustering purposes, Jensen-Shannon distribution as a distance measure was a better choice as compare to the cosine similarity for word topic detection using keyword clustering.

L. AlSumait et. al [6] published a research working on a stream processing online model of LDA for adaptive topic modelling of text content. This main motive of making LDA topic modelling online was to detect different thematic patterns and identifying the different changes and emerging patterns in topics over time. This would provide as a useful insight in understand the topic transition behavior. The model does not need information from the previously viewed/processed articles and can easily update the new information presented by the stream with latent vectors. Since it is a stream processing model, this LDA variant also has the ability to magnify and look at the relevant topics in a particular instance of time. The ability to work in an online mode is based on the assumption that the documents are arriving as part of a time slice and OLDA considers the temporal ordering information. A topic model with n components is used to model the newly arrived document. This generated model is then used as a prior for the new documents that are processed from the stream. The OLDA model is efficient in complexity and memory usage since the inference problem in static LDA is solved using chunks of document for inference. This makes it really useful for genuine applications. OLDA uses Gibbs sampling as an approximate inference method for topic-word assignments. The OLDA approach takes text streams, confidence levels, weight vectors and Dirichlet values as inputs which are used to initialize the α and β priors. One of the important features of OLDA is to detect emerging topics which is a good indicator of knowing when something is novel, interesting and catching up pace.

W. Xei et. al [7] published a ‘TopicSketch’ model which will can be used for topic detection in a ‘bursty’ fashion. Twitter with its popularity in the micro-blogging domain deals with millions of tweets per day. There are often times when a certain topic or event triggers a huge surge of relevant tweets about that topic within a really short period of time. This generally marks that topic as a topic of user interest for that specific time period. This research was focused on detecting such ‘bursty’ topics. All researches previous to this tried capturing bursty words but not topics and this failed to represent sufficient information about the burst of tweets. TopicSketch is mainly based on 2 techniques:

- Sketch-based topic model
- Hashing-based dimension reduction technique

The algorithms are divided in two steps. The first step consists of creating pairs and triples of words. These values are updated based on the surge of tweets about these pairs which helps understand the popularity rate of the topic and in turn increase the acceleration factor. The acceleration is calculated based on the number of tweets about that tweet in a small duration of time slice. This time slice helps guide the omission of more common/popular topics that are trending but are not bursts. Essentially most of the bursty topic turns into a trend topic, however, the main purpose of this research was to look at how to detect bursts in topics based on the word pair or triples and their mentions. This forms the first step of the sketch-topic model. The second step is about learning the topic from the words by using hashing-based dimension reduction which is base of tensor decomposition technique. The main idea behind decomposition is calculating the acceleration of the word vector representing a particular topic using attention matrices to point out the inner details of a particular topic and understand what constitutes the acceleration and determine the trigger. One major advantages of this approach is the topic inference part which was using the tensor decomposition logic can be parallelized and thus executed efficiently. After the final bursty topics are determined a heuristic approach to filter the trivial, noisy, spam topics and rare words is used. The bursty topic accuracy was measured with variants of tensor decomposition and SVD. The results of this research showed that the topic modelling with bursty topics was a good idea for streaming content with high volume of data and the characteristic of accelerating rapidly towards a topic.

Q. He [8] et. al wrote an insightful paper about various techniques used in topic detection tracking and the comparisons between them. [8] proposed a discriminative probabilistic model which is relatively simple compared to the complicated generative models like von-Misses Fisher or LDA. The proposed DPM was equivalent in clustering to a variation of TF-iDF where only discriminative words were considered. The experiment results showed, for the datasets in consideration, that DPM was good in assigning multiple topics to a document in offline mode and was good at not raising false alarms for online mode. vMF and LDA do shine when the training data is substantial, however, a model as simple as DPM is well performant with less data.

Ibrahim et. al [9] also presented a comprehensive study on topic detection tools from twitter stream. This study revolved around 5 categories of topic detection techniques. Exemplar-based, matrix factorization, frequent pattern mining, clustering and probabilistic models. Each of these 5 categories were looked under the microscope with evaluation results and comparisons for all of them. Some of the common algorithms used in these categories were variations of k-means, DBSCAN, spectral clustering, stochastic LSI, Alternating Least Squares, Rank-One DownDate(R1D), Soft Frequent Pattern mining and Bngrams. Results on three twitter datasets showed that SFM and Bngrams achieve the best term precision while CSS achieves the best term recall and topic recall.

L. Chen et. al [10] recently published a paper addressing the problems that were involved in topic overlaps while modelling topic detection on online stream of topics. Topic overlaps happen in some cases depending on the data encountered by the model. This is also an evolving problem faced by many models which call for sanitization and normalization of topic vectors on a periodic basis to clean up the overlap. This paper proposes a hierarchical topic modelling approach which is a knowledge based semi-supervised approach and reduces the overlap between the nested or intermingling topics and helps with topic distinction that could otherwise be overlapped. The results of this research show that the use of external knowledge to discern categories that are partially overlapping or concentric is very helpful for better results.

Qian Zhou et. al [11] have recently proposed a single pass clustering algorithm which can be very useful in optimizing the implementation of this project's clustering analysis. This optimized approach is proposed for topic detection and so it would be interesting to see if this optimization could be incorporated in our implementation.

III. Machine Learning - Algorithms & Architecture

Types of Machine Learning Algorithms:

- Supervised
- Unsupervised

A. Unsupervised Algorithms

There are many unsupervised learning models that are used to great extents in production environments when dealing with data that is unstructured to be labelled or labels are not available. This can also be the case when the model is looking for clustering analysis and not really looking forward to making predictions. Clustering is basically segregation of data points into affinity regions where all points within a specific region have similar properties. Some common unsupervised learning algorithms are k-means, KNN, autoencoders, PCA, etc.

Some of the unsupervised methods that will be used in this project are specifically related to text processing. Here is a short description about each of them.

1. Word2Vec

One of the core models that started out with a novel approach to text processing using word embeddings is Word2Vec. Mikolov et al. [12] published this model which showed great promise in understanding and modelling the hidden meaning of natural languages. An example of vector operations that were possible on the word embeddings is as follows.

```
W("woman")-W("man") ≃ W("queen")-W("king")
```

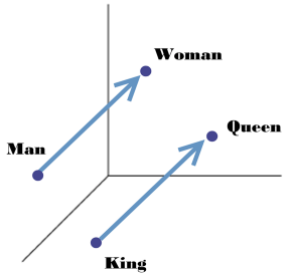

Figure 1: Gender Vectors [14]

This provides for a unique way of performing arithmetic operations on textual data. Word2Vec using negative sampling approach to derive word embeddings. This model proved super useful in learning semantic relationships that are very hard to model via programming models or other machine learning models.

Before we move on to Doc2Vec, it is imperative that we look at 2 algorithms which are core to Word2Vec.

- Continuous of Bag of Words
- Skip Gram

In CBOW, a sliding window of surrounding word vectors is used to predict the current word. The prediction is based on the context. This context is created from the surrounding words which are nothing but feature vectors for each surrounding word in the window. Feature vectors which represent words that are similar or have more affinity to be seen together are closer to each other based on a metric distance between the vectors. A simple diagram showing this can be seen below:

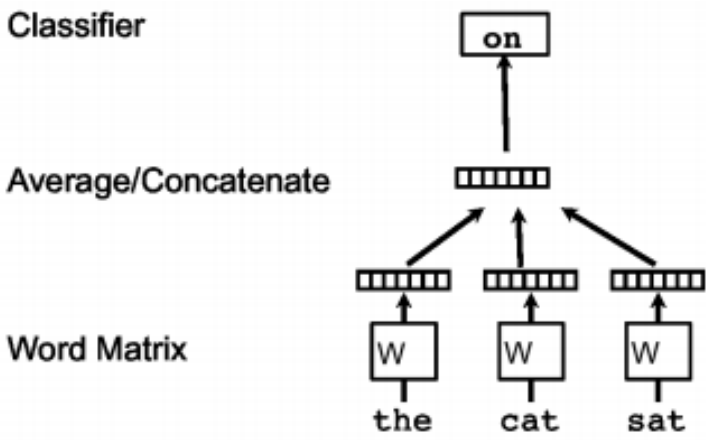

Figure 2: CBOW -- Surrounding words used to predict the current word [15]

This is the basic reason why Word2Vec is able to model arithmetic operation like the one seen in the Figure 1 since the floating-point vectors represent closeness for similar text content.

In Skipgram, the algorithm works completely opposite to CBOW. This algorithm predicts the surrounding words over a window based on the current word. This also means that the algorithm is computationally more intensive. The CBOW model is not great for infrequent words, however, Skipgram model does really well with infrequent words. The below diagram shows a basic idea of how the word window slides and what is used as training sample or each word in Skipgram training.

Source Text									Training Samples
the	quick	brown	fox	jumps	over	the	lazy	dog.	(The, quick), (The brown)
the	quick	brown	fox	jumps	over	the	lazy	dog.	(quick, the) , (quick, brown), (quick, fox)
the	quick	brown	fox	jumps	over	the	lazy	dog.	(brown, the), (brown, quick), (brown, fox), (brown, jumps)
the	quick	brown	fox	jumps	over	the	lazy	dog.	(fox, quick), (fox, brown), (fox, jumps), (fox, over)
the	quick	brown	fox	jumps	over	the	lazy	dog.	(jumps, brown), (jumps, fox), (jumps, over), (jumps,)the

Figure 3: Skipgram – Training samples to learn surrounding word probabilities

2. Doc2Vec

This is another unsupervised text learning model based on textual data. This model is an extension to Word2Vec where the floating-point continuous vectors represent documents instead of words. This is especially useful when trying to determine similarity of content of articles based on the entire content and not only the hot spot words. In Doc2Vec, there is an addition of information to the training set so that we also learn the corresponding document features. For every word that is trained in the Word2Vec model, we add a paragraph vector along with it. As the model

trains itself to understand the semantic relationships between words, it also trains itself to understand the semantics of each document based on this paragraph vector. This vector is crucial in determining the affinity or correlation between similar documents. There are variations to this approach by adding tagging vectors for meta information or by using topic vectors along with paragraph vectors. They all follow the same learning model though which can be seen in the diagrams below:

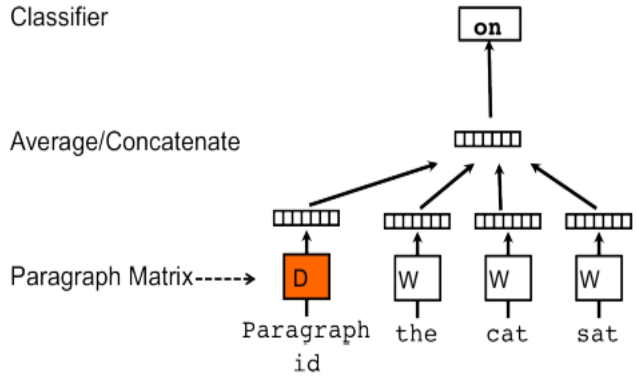

Figure 4: Doc2Vec model with paragraph vectors added to word model [15]

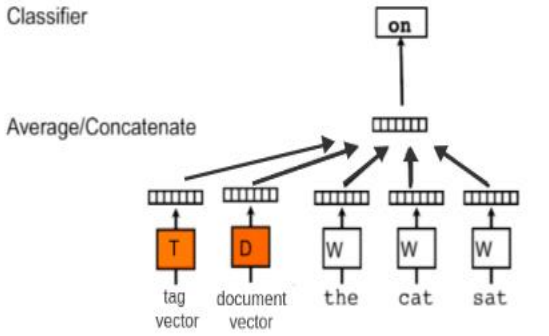

Figure 5: Doc2Vec model with document and tag vectors added to word model [15]

3. LDA

LDA is a machine learning model that is built on the idea of Latent Dirichlet Allocation. The idea is basically to generate topics from each document that give a cogent representation of the content of the document. LDA uses the information from word2vec and doc2vec vector to build its own distribution vector where each item in the vector represents a topic contained in the document. It is a model that can be termed as "*distribution over a distribution*". The base distribution is words per topic and the outer distribution is topics per document.

Topic models are defined around the idea that the human understanding and semantics of a document are governed by some hidden/latent variables that are not directly observable. The main goal of all topic modelling is to uncover/model these latent variables that shape the meaning of our document. In LDA, each document is constituted by a set of topics and each topic is constituted by a set of words. A trained LDA model should output a vector of topics along with their presence measurement vector values.
Example topic mixture output for a document:

Topic1: 39%, Topic2: 23%, Topic3: 13%, Topic4: 4%, ...

The parameters of LDA can be seen as follows:

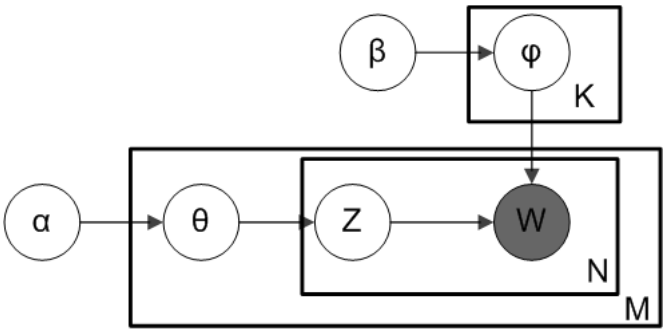

Figure 6: Plate Diagram for parameters of LDA [16]

Parameters:

α – Number of topics per document
β – Number of words per topic
θ – Topic distribution for particular document M
φ – Distribution of words for particular topic K
Z – Topic for Nth word in document M
W – Specific Word

B. Supervised Algorithms

Supervised machine learning models follow the idea of having a labelled training dataset which acts as a reference point for learning a function which can predict the correct output. The model learns this function with the expectation that it should be able to predict the correct or relatively correct output for unseen data samples. Two most broad categories of supervised models are regression models and classification models. SVC, SVM, Neural Networks are among the most common classification models. Some brief details about them are as follows.

1. Support Vector Machines

Support vector machines are a type of machine learning algorithms that are used for classification problems – binary or multiclass. The intuition states that when we have a set of points, we should be able to separate those points into classification based on a separating hyperplane. This hyperplane should ideally be able to classify with complete confidence and no errors. A hyperplane is defined as a plane which separates the data points into required classes. The hyperplane could be line if the input dimension is 2 or a plane if the input dimension is 3. It becomes difficult to imagine the plane with higher dimensions.

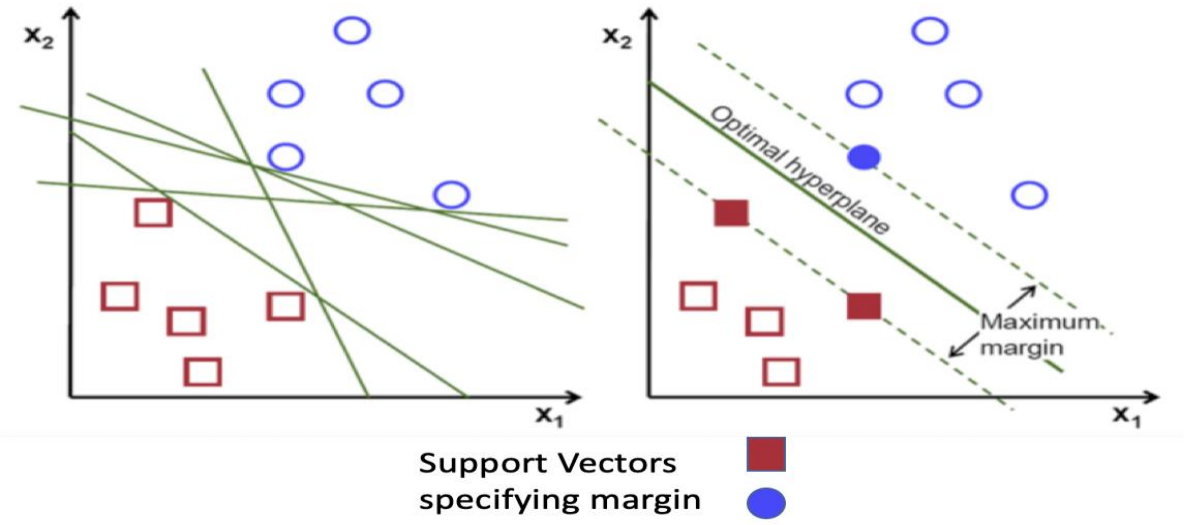

Figure 7: SVM model for classification [17]

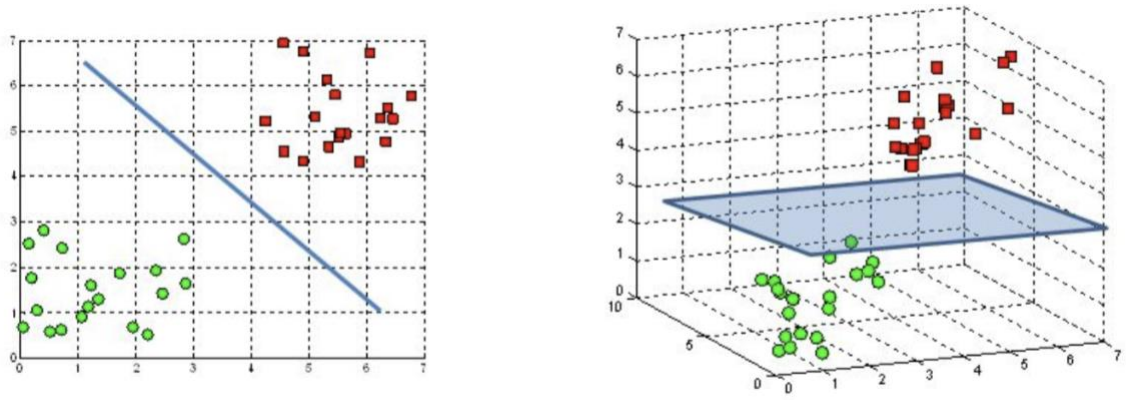

Figure 8: SVM separating hyperplane in 2D and 3D spaces [17]

2. Neural Networks

Neural networks are probably one of the most discussed machine learning topics in today's world. In its most simple form, a neural network is a set of hidden layers of neurons which are modelled just like the neurons in a human brain. There are activation functions which decide whether a neuron fires or not based on the input to that function which is a weighted sum from the previous layer. There are various activation functions like relu, sigmoid, tanh, softmax, etc. The final consolidated output passing through all hidden layers is then given to the output layer where predictions are made. These predictions are measured against the labelled output values and an error is calculated. The major difference between other machine learning models learning capabilities as opposed to neural networks comes from the way it treats the error/loss along with the numbers of neurons it can propagate the error/loss to. This step is known as Back Propagation and is crucial for any learning to happen in the model. The error function could be any distance metric like root mean squared error, etc. A simple form of Neural Network is as shown below.

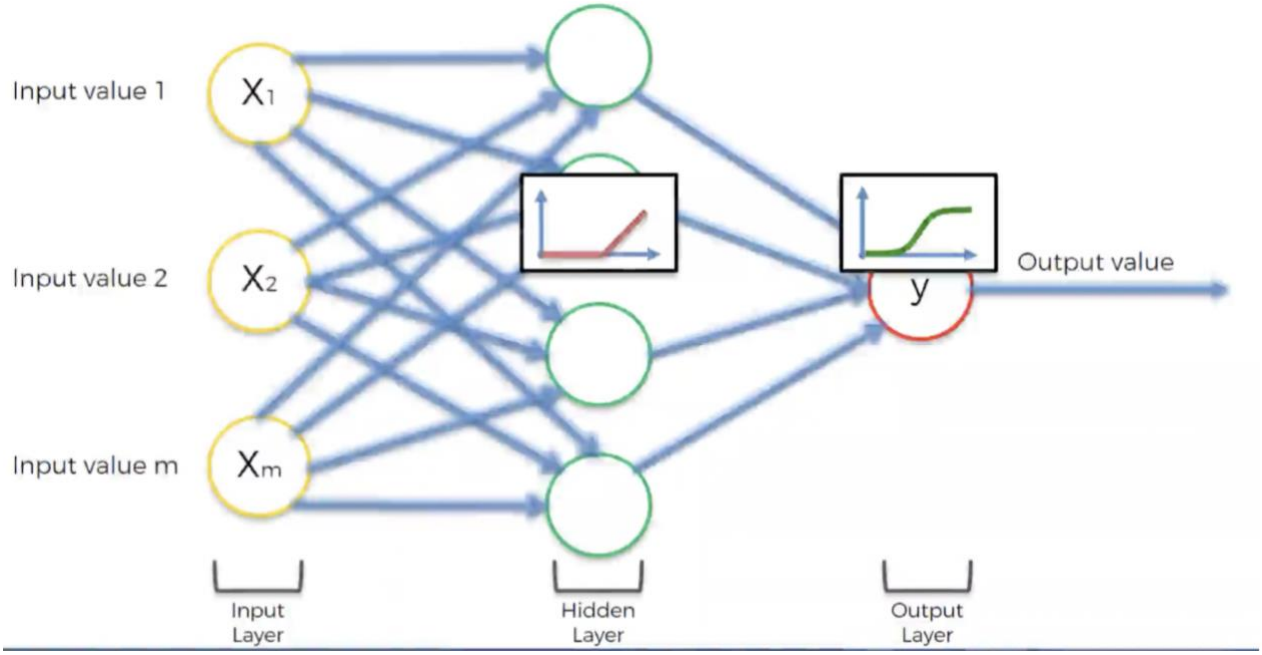

Figure 9: Fully connected simple neural network

C. Dimensionality Reduction

We may have a set of input features which are recorded in a given dataset but not all of them are necessary for the model to learn a function. Thus, dimensionality reduction offers a way of considering only the input features which are minimally required to learn the required regression or classification function. Some of the advantages of dimensionality reduction are improved training time, performance efficiency of the model, clarity of contributing factors, etc. These factors make dimensionality reduction a very crucial part of input feature engineering for machine learning models. One of the most common dimension reduction techniques is Principal Component Analysis.

1. Principal Component Analysis

PCA is basically a method to calculate the eigen values and eigen vectors of a covariance matrix after applying normalization to the data in the matrix. The data points in the scoring matrix can be looked up as the contributing factors to decide on the features for the model. PCA is the simplest of the eigenvector based multivariate analyses. PCA is specially known to expose the internal structure of the data in a way that best explains the variance of the data.

D. Framework Architecture

The framework architecture is the plan of action / rule outline which will be followed by the project. A high-level diagram of the framework is described below:

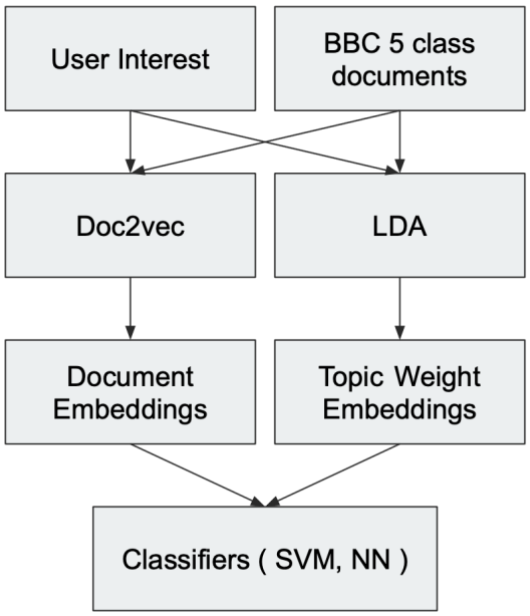

Figure 10: Proposed Framework Architecture

The architecture specifies that we will have 2 datasets for checking the accuracy of our model. The third dataset which is mentioned in the datasets but is not a part of the architecture is basically just used to create the corpus understanding for the LDA model. Corpus understanding can be thought of as the model trying to learn most of the vocabulary that it will encounter in these articles. The bigger the size of the corpus, generally, it is better for the model performance.

1. Setup

As per the architecture diagram, the user interest and BBC dataset will be used to train as well as test the doc2vec and LDA models.

- Dataset of 1.4 million articles is provided to unsupervised learning models in their respective experiments for corpus learning.
- Dataset BBC is only provided to LDA while dataset user interest is provided to both the unsupervised models.
- Then for each document from the same dataset, we infer output weight vectors / topic vectors based on the respective models.
- These vectors are made fixed length vectors of length 100 for processing power ease.
- These vectors are provided are input features to the supervised classifier models which are SVC, SVM and NN.
- The classification results are noted, and accuracy is calculated.

The train and test split will be a ratio of 80:20 and we will see results with cross validation as well as without cross validation. The output vectors generated by the doc2vec and LDA models will be fed individually to classifiers with different parametric architectures and the classification results will be captured.

IV. Datasets and Model Engineering

A. Datasets

There are three primary datasets for this project.

1. One-week global news feed dataset

Kaggle dataset [19] of 1.4 million news article URLs published between August 24,2018 to August 31, 2018. The features of the dataset are – 'publish_time', 'feed_code', 'source_url', 'headline_text'. After data transformation the actual corpus of the unsupervised models was created from about 400,000 articles.

2. BBC news Dataset

Kaggle dataset [20] of 2225 news article texts with labels as ''business', 'entertainment'. 'politics', 'sport', 'tech'

3. User Interest Dataset

This dataset [21] is manually created as part of a class project from one of my machine learning classes. The dataset consists of article texts mined from newsletter archives of a blog. All articles from one blog were labelled as 'Interested' while all articles downloaded from other blog newsletter archives were termed as 'Not Interested'. The idea here was to model user interests and not topic to see if modelling user interests completely orthogonal to topic categories is possible. This was a small set of approximately 200 articles.

B. Data Engineering

The data engineering efforts are distributed across following parts:

- Data Gathering
- Data Cleaning
- Data Transformation
- Feature Engineering
- Parameter Tuning

1. Data Gathering

Dataset 1 was a dataset of 1.4 million news article URLs. A custom downloader/scraper application was written to download content from these URLs. Each URL was visited, and the entire HTML content was parsed. Python's 'newspaper3k' library was used to parse the main text of the article from the HTML payload. All these texts along with the URLs were saved to a MySQL relational database for later use of training the machine learning models.

Dataset 3 was a dataset of newsletter archives that was created by manually curating and downloading the articles from a newsletter archive [21]. This dataset was created as part of a project in a graduate course along with my project partner in that project. It is reused here since it is a perfect fit for the user interest modelling problem.

Dataset 2 was downloaded as it is from Kaggle. This is a labelled dataset with 5 specific categories of data namely, business, entertainment, politics, sport, tech.

2. Data Cleaning

Cleaning data does not seem like but is an iterative process. This is solely because every time the model fails at something or categorizes something wrong, you go back and try to find the sample that went wrong. While looking at such samples, I realized all the different types of cleaning that I need to do on the original downloaded texts.

- Language filter: For all the downloaded articles, there were many articles which were not English. They were in Spanish or Dutch languages. It is important to understand that the models used do not really have a problem with different languages as far as it has the necessary corpus. However, it is my limitation of not understanding these languages, that I had to filter them out for making correct sense of the results. This filter was developed manually but with some use of Python's 'nltk' library.

- Blank / useless articles filter: For all the downloaded articles, there were some articles which were either blank because they contained all images for that article content or were gibberish because they only had tweets etc. in them and no texts that could be extracted from the payload.

3. Data Transformation

Data transformation is an activity which generally involves molding the data into formats which are useful to be fed to the machine learning models. It also involves filtering out content that is specifically not useful based on the type of machine learning model being used. Some of the data transformations that were done on the base datasets after data cleaning are as follows:

- Filter stop words:
  Remove the words like *'is', 'and', 'the',* etc. which are common to all topics and don't add or remove any meaning from the latent variables of those topics. These words are good at digressing the meaning of the latent variables and so removing them helps a lot with the accuracy of the model. These words can also be treated as fillers which are cogent for language understanding flow but act against the purpose of model learning.

- Filter extremely infrequent words:
  Words which occur either once or twice in the entire corpus are contributing no meaning to the understanding of the model. Hence, they are removed to avoid digression from the latent variables.

- Filter punctuation:
  Punctuation marks like *'?', ',', '!',* etc. do add meaning for a human reading the articles and provide a context to the way a sentence of the article is meant to be read. However, as far as the model is concerned, these add no value to the model understanding and only confuse the model's understanding at best.

- Filter non-standard English words:
  Many articles are news articles which have sentences which are quotes about someone speaking a particular way or people quoted as saying something. Thus, there are many words which, though never seen in written communications, occur in these datasets that are of no value. Ex: *'gotcha', 'lol', 'attaboy', 'asap'*, etc.

4. Feature Engineering

When both the unsupervised machine learning models were trained with the corpus of text articles from datasets, id->token pairs were created, and a dictionary of such pairs was stored for the model understanding of both doc2vec and LDA. After training the unsupervised models, output weight / topic vectors were deduced from these models and converted to a matric format. The matrix definition was as follows:

- Each row represented a document in the dataset
- Each column represented a topic that the model learned
- Each [row, col] floating point value represented the contribution of that topic in the document

This matrix of features, where each row acted as an input, was given to the classifier with output labels from the dataset for classification.

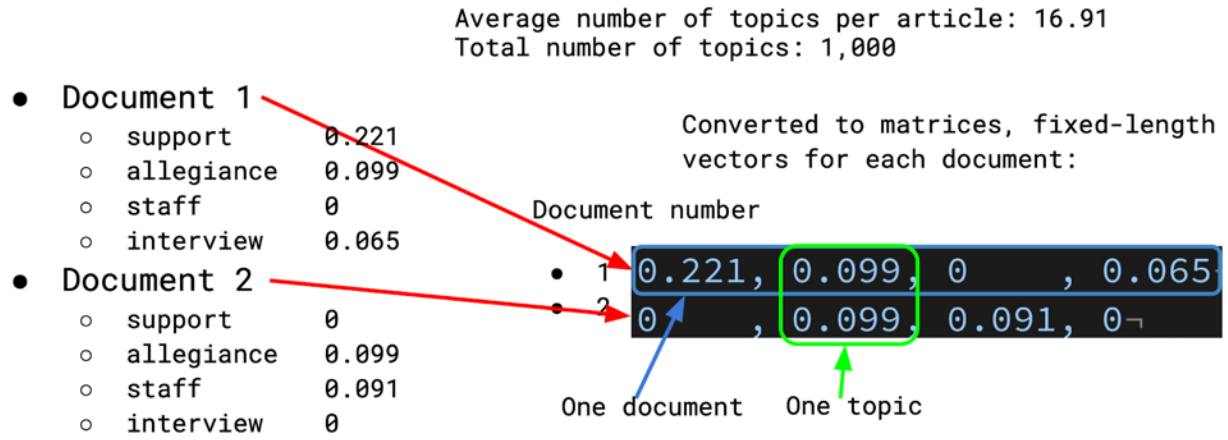

Figure 11: Feature Handholding - Unsupervised to supervised – LDA

C. Model Parameter Tuning

After trying various parameters for different models, the final hyper parameters for each of the machine learning models used in this project as follows:

1. Output Vector length for Doc2Vec

The output of doc2vec inferred vector was a fixed length vector of length 100. This was decided after trying other values like 10, 50, 200 and 350.

2. Topic Vector length for LDA

The output of LDA inferred vector was a fixed length vector of length 1000. This was decided after trying other values like 10, 50, 200, 350, 2500, 5000 and 31000.

3. Neural Network Hyper Parameters

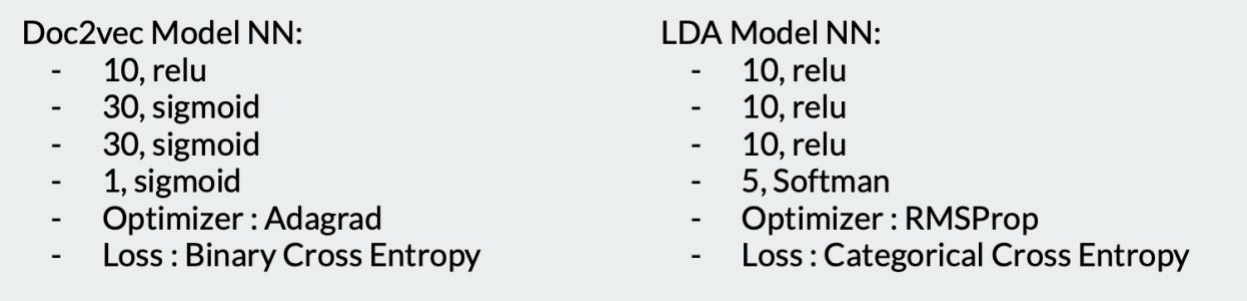

Figure 12: Neural Network Hyper Parameters

4. LDA Parameters

There are 2 main parameters which can crucially direct the performance of an LDA model.
α – Number of topics per document
β – Number of words representing / contributing to a topic

It is necessary to tune α since a less value of alpha can result in the model learning very less. This happens because it is not allowed to be specific about the understanding of the context and is made to limit it to a very small set of topics. This makes the topic categories very broad, thus, making them so intermingled that a clear classification is very difficult.

On the contrary, if α is chosen too large, the number of topics learned by the model is too many and it makes the model too picky or strict about classifying a certain text as an article even if it pretty relevant to or may be well described by aa broader category. This causes the problem of classifications that are too strict and don't generalize well.

β on the other hand is a less trickier parameter to tune. This mostly depends on the amount of corpus that you have as base learning for the model. If the corpus is really large, we could have more words and still be okay with the results. However, if the corpus is not that big, β should be kept not too high or else each word essentially ends up contributing to each topic in some way or the other, thus disrupting the classification.

V. Experiments and Results

A. Working Environment

Each of the experiments described below was run on the following machine.

Operating System: MacOS Mojave 10.14.4
Processor: 2.3 GHz Intel Core i5
Memory: 8 GB 2133 MHz LPDDR3
Code Editors: VS Code, Jupyter Notebooks
Programming Language: Python 3.6+
Database: PostgreSQL

This hardware is not the optimal choice for this project since it takes a lot of time to train the corpus of the unsupervised models with huge text content (about 40 hours). Thus, the model corpus understanding was built in parts by running the model over chunks of text from the entire dataset because of the time required to train the entire corpus. An environment with GPU or at least 16 GB memory should be more preferable. The processing was also divided in parts to avoid the degradation in performance of the machine along increasing time in the training causing the system to freeze on a few occasions. Text chunks of 50,000 articles were used to train the model in each part completing the overall dataset after data transformations in about 15 iterations.

B. Experiment Steps

For every experiment that we perform, the basic steps remain the same as below:

- Train the model with respective dataset
- Make a test set out of the dataset
- Infer vectors for each of the test articles from model
- Classify these output vectors with the help of a classifier
- Measure results in terms of prediction accuracy

C. Experiment Configurations

Experiment Number	Dataset	Unsupervised	Supervised
1	User Interest	Doc2Vec	Linear SVM
1	User Interest	Doc2Vec	SVM
1	User Interest	Doc2Vec	nuSVM
1	User Interest	Doc2Vec	ANN
2	User Interest	LDA	Linear SVM
2	User Interest	LDA	SVM
2	User Interest	LDA	nuSVM
2	User Interest	LDA	ANN
3	BBC	Doc2Vec	Linear SVM
3	BBC	Doc2Vec	SVM
3	BBC	Doc2Vec	nuSVM
3	BBC	Doc2Vec	ANN
4	BBC	LDA	Linear SVM
4	BBC	LDA	SVM
4	BBC	LDA	nuSVM
4	BBC	LDA	ANN

Table 1: 4 Experiments performed for user interest and topic modelling

D. Results

All the experiments that we perform are based on the latent understanding of the various articles from the corpus trained on the models. Thus, before we move on to actual results, it is only insightful to visually peek at the understanding of the model after they have been trained on the datasets before predictions.

1. Model Learning

a) Doc2Vec

Following are the learning representations of the unsupervised models after training them on the respective datasets.

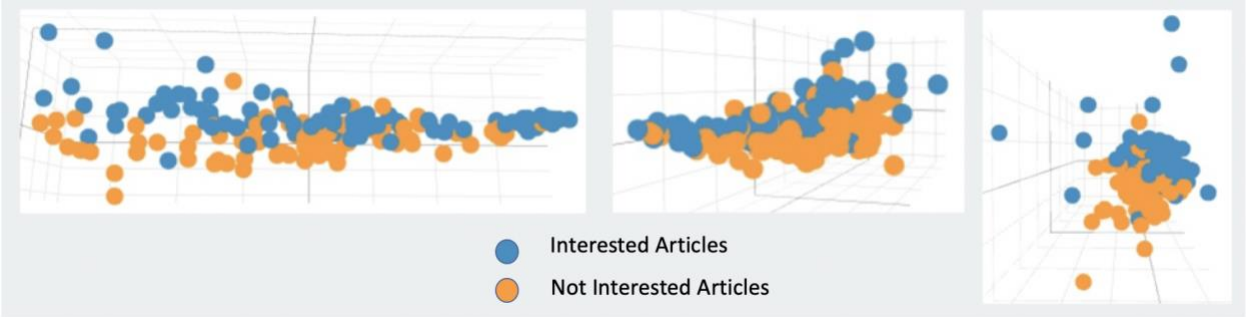

Figure 13: Doc2vec model learning after training on user Interest Dataset

The diagram above is a dimensionally reduced view of the doc2vec model. The doc2vem model was trained to be in 100 dimensions. However, for plotting and understanding purposes, this diagram was creating by reducing the dimension to 3 using PCA. These 3 images are of the same plot but shown with different angles of the 3 dimensions for visibility of separation between the two classes – Interested and Not Interested.

b) LDA

For a learning representation of the 5 class BBC dataset, we choose to show LDA visualization representation since it is a better and more succinct for understanding. Below figure 12 consists of 2 parts. The quadrant plot represents the distribution of topics from the corpus learning and is the InterTopic distance map. The bar plot represents the words that contribute to the selected topic. The red bar signifies the number of times that word contributed to this topic. The blue bar signifies the frequency of that word in the entire corpus.

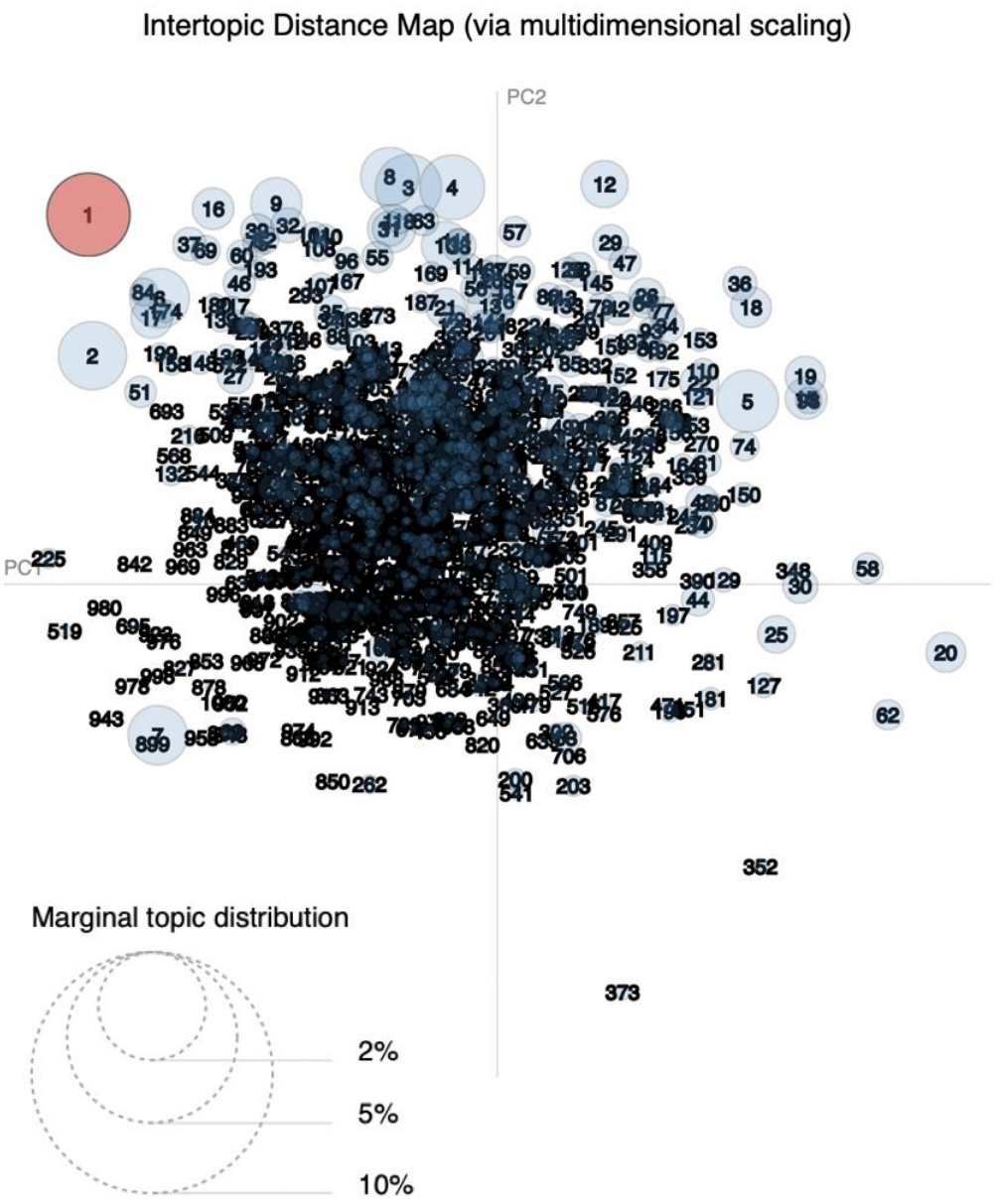

Figure 14: Topic distribution of 1000 topics of LDA learning on BBC datasets

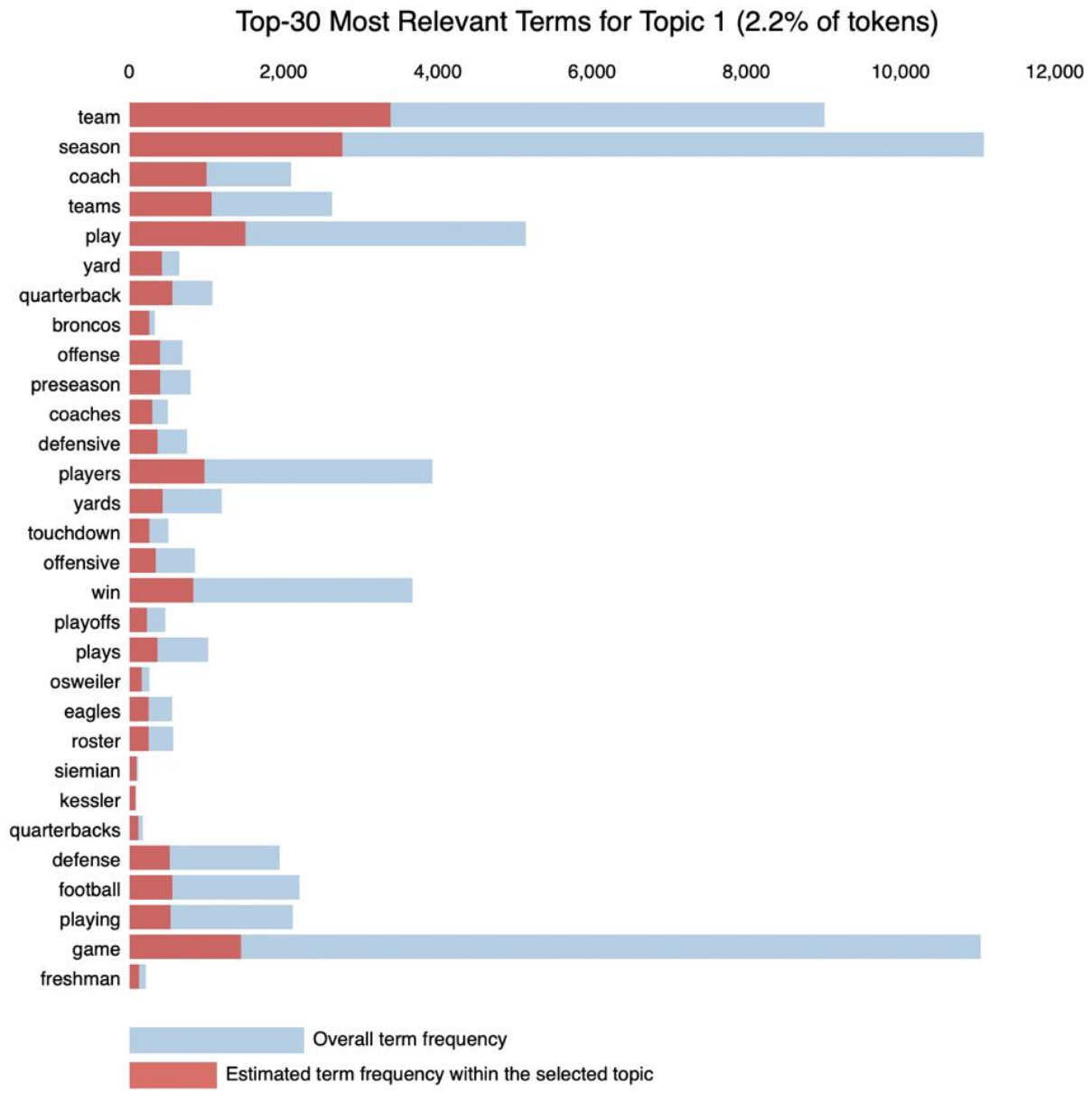

Figure 15: LDA Term distribution for the selected topic 1 from Figure 14

Following are the results of experiment wise results based on the type of modelling to develop a clearer understanding.

2. User Interest Modelling

The results are for each experiment are given in the following table.
The meaning of the metrics is as follows:
Accuracy – Mean of accuracies across 5-fold cross validation sets.
STD - Standard deviation that happened across different cross validation folds.
Mean Normalization – With/Without Normalization of the input vectors to avoid bias.

Exp Number	Dataset	Un-supervised	Supervised	Accuracy (%)	STD (%)	With Mean Normalization (%)	Without Mean Normalization (%)
1	User Interest	Doc2Vec	Linear SVM	96.30	+/- 17	80.55	96.30
1	User Interest	Doc2Vec	SVM	81.25	+/- 10	83.33	81.25
1	User Interest	Doc2Vec	nuSVM	95.13	+/- 03	86.11	95.13
1	User Interest	Doc2Vec	ANN	88.90	+/- 06	88.90	85.40
2	User Interest	LDA	Linear SVM	96.66	+/- 09	95.00	96.66
2	User Interest	LDA	SVM	64.20	+/- 08	55.30	64.20
2	User Interest	LDA	nuSVM	97.25	+/- 04	95.00	97.25
2	User Interest	LDA	ANN	98.93	+/- 02	98.93	97.82

Table 2: Accuracies for experiments targeted for User Interest modelling

a) Observations

- Most stability with ANN or nuSVM for binary classification
- Higher accuracy but really unstable results with strong STD in linear SVM
- Accuracy generally decreased with mean normalization in SVM variations. This could be because the separation margin between the support vectors where the hyper plane lies was narrowed due to normalization. Due to this narrowing of separation margin, the number of misclassifications could go up since the goal of SVM is to maximize the separating margin.
- Accuracy generally increases with mean normalization in ANNs
- Best model for User Interest modelling is unsupervised LDA in combination with supervised ANN providing best accuracy and with extremely low standard deviation.
- It is possible to model user's interest without defining predefined categories simply on the basis of a binary flag specifying interest.

3. Topic Modelling

The results are for each experiment are given in the following table.

The meaning of the metrics is as follows:
Accuracy – Mean of accuracies across 5-fold cross validation sets.
STD - Standard deviation that happened across different cross validation folds.
Mean Normalization – With/Without Normalization of the input vectors to avoid bias.

Exp Number	Dataset	Unsupervised	Supervised	Accuracy (%)	Standard Deviation (%)	With Mean Normalization (%)	Without Mean Normalization
3	BBC	Doc2Vec	Linear SVM	95.04	+/- 04	87.21	95.04
3	BBC	Doc2Vec	SVM	77.00	+/- 18	46.34	77.00
3	BBC	Doc2Vec	nuSVM	94.00	+/- 09	81.27	94.00
3	BBC	Doc2Vec	ANN	93.00	+/- 02	93.00	92.03
4	BBC	LDA	Linear SVM	96.00	+/- 08	80.55	96.30
4	BBC	LDA	SVM	76.25	+/- 19	83.33	81.25
4	BBC	LDA	nuSVM	94.47	+/- 12	86.11	95.13
4	BBC	LDA	ANN	96.33	+/- 01	96.33	93.00

Table 3: Accuracies for experiments based on BBC dataset targeted for Topic modelling

a) Observations

- Most stability with ANN for 5 class classification with either unsupervised algorithms.
- Almost equally good accuracies with Linear/nu SVMs but really unstable results with strong STD
- Accuracy co-relation with mean normalization is exactly similar to what we found in the previous experiment with NN as opposed to other classifiers and for the same reasons
- Accuracy generally increases with mean normalization in ANNs
- Best model for topic modelling is unsupervised LDA in combination with supervised ANN providing best accuracy and with extremely low standard deviation.
- It is possible to model topics across a range of predefined categories.

4. Additional Insight

After this framework was created and all the results were observed, it was used to see if there was a possibility of using this exact same design to work on a problem where the text content does not have variations and most of the words in all the content's individual samples are similar. This test was possible using Myers-Briggs personality type dataset [18]. This dataset has 16 distinct personality types across 4 different axes.

- Introversion (I) – Extroversion (E)
- Intuition (N) – Sensing (S)
- Thinking (T) – Feeling (F)
- Judging (J) – Perceiving (P)

The tests performed on this dataset using our framework were to model the different personality types. However, the dataset is highly representative of a certain personality type as opposed to all the others, thus, having a biased representation as a whole. Also, the text content pertaining to each of these personality types and users has very similar vocabulary. The word occurrences in each sample are extremely similar to other samples. This is the primary reason for our framework to perform really poor on such problem. However, this was a test that was done just to see a possibility / limitation of the framework and was not the primary focus of the project. This test did reveal that the hybrid framework is not a good choice for problems where the individual samples of text are extremely similar to each other, thus, not allowing any room for the model to learn different word correlations contributing to topic semantics.

5. Parameter Tuning Explanations

Following is a peek at how accuracies changed across the cross-validation folds while training the model with 5-fold cross-validation. These cross-fold validation accuracies are not normally output by the model and are sourced from the logging outputs of the model while running each epoch and cross validation folds across each epoch. After every fold, the model logs the output of that fold giving us the accuracy and loss for that fold.

a) Cross Validation:

Cross validation is crucial for a machine learning model since the learning could be over fitting. Over fitting is a case of model learning when the model learns to follow the training dataset in a tightly coupled fashion and gives bad results when the testing data digresses even slightly from the training data. This is known as over fitting since the model behaves like 'learning' rather than 'understanding' the relationship of independent variables with the dependent variables.

By applying cross-validation it can be seen that a batch size of 10 works best as opposed to 50 or 100 for error correction step. The results with no cross-validation look really good but they might cause problems as discussed above.

Following are the plots of changes in accuracy and the values that guide the hyperparameter tuning for batch size of error correction and cross-validation in the training phase of the models.

```
training accuracy  : 0.941%
validation accuracy: 0.939%
```

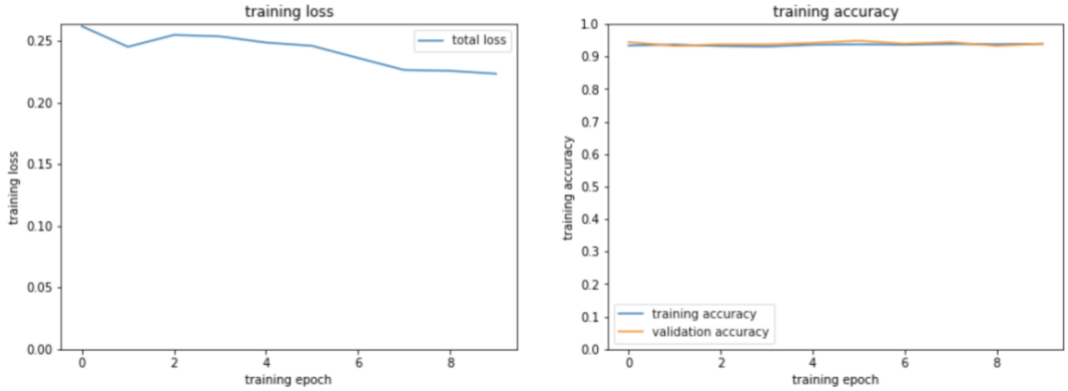

Figure 16: Accuracies with 10 epochs no CV batch size = 100

```
training accuracy  : 0.458%
validation accuracy: 0.458%
```

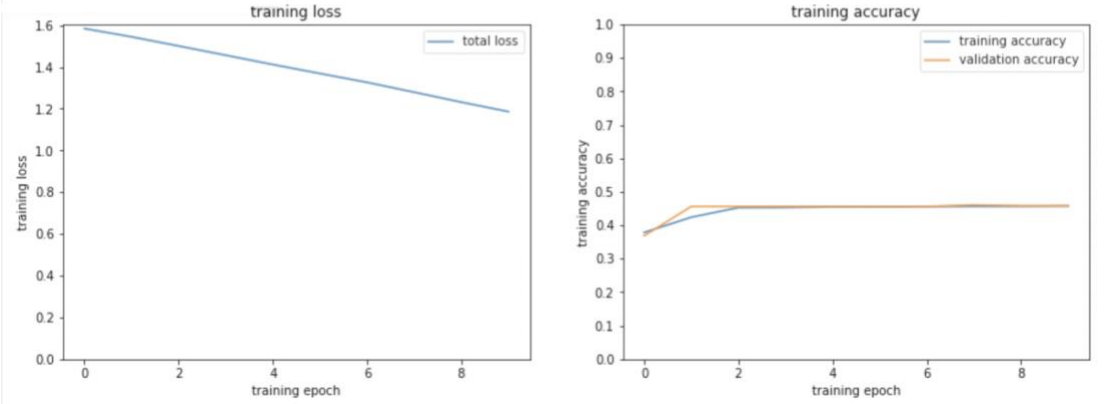

Figure 17: Accuracies with 10 epochs with 5-fold CV batch size = 100

```
training accuracy  : 0.622%
validation accuracy: 0.627%
```

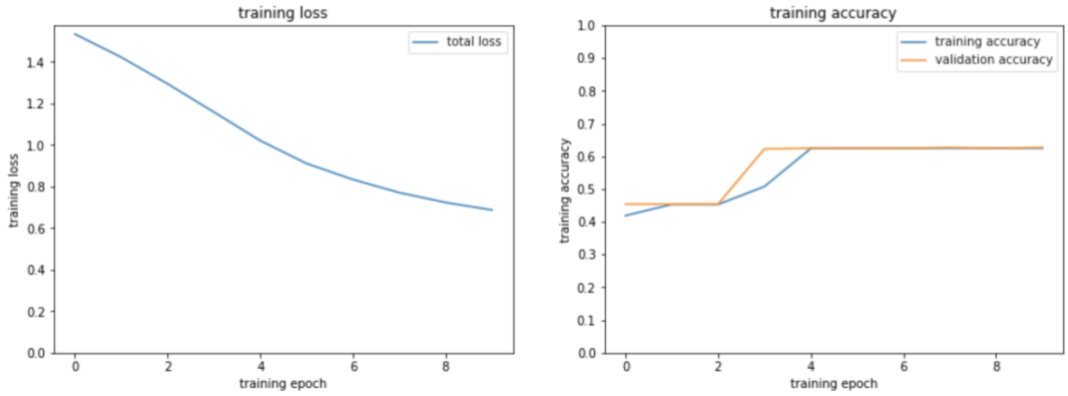

Figure 18: Accuracies with 10 epochs with 5-fold CV batch size = 50

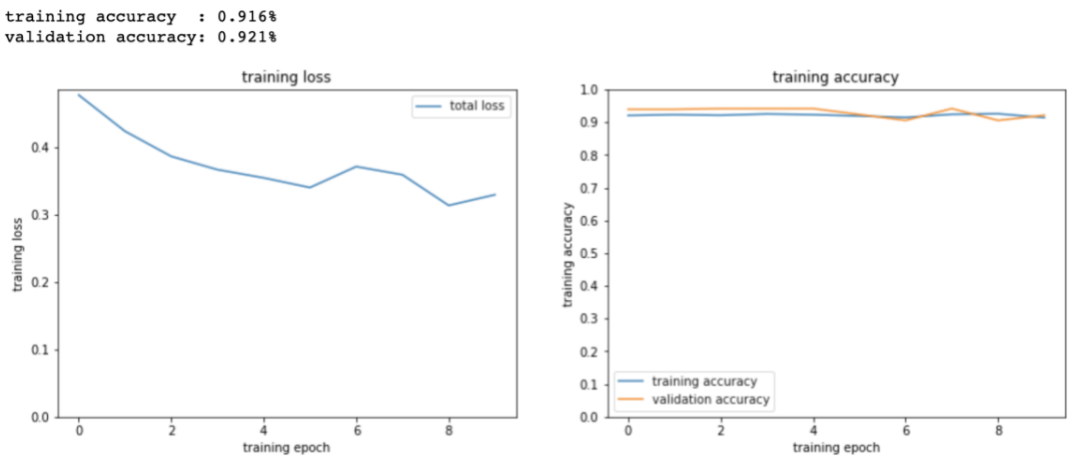

Figure 19: Accuracies with 10 epochs with 5-fold CV batch size = 10

b) Mean Normalization:

Following table shows the stability that was achieved after applying mean normalization. After every cross-validation fold, the accuracy is shown as below for the LDA-ANN hybrid experiment.

Fold #	Without Mean Normalization	With Mean Normalization
1	0.927	0.950
2	0.966	0.966
3	0.955	0.955
4	0.955	0.950
5	0.843	0.961

Table 4: Sample 5-fold cross-validation accuracies in LDA-ANN relative to mean normalization

c) Number of Topics

Following is a plot of the results with LDA hyperparameter tuning of number of topics set to 31,000, 2000 and 1000. These plots demonstrate how accuracy of the model changed across varying topic numbers before finally figuring out the optimal topics to be 1000 and what guided the change in hyperparameters.

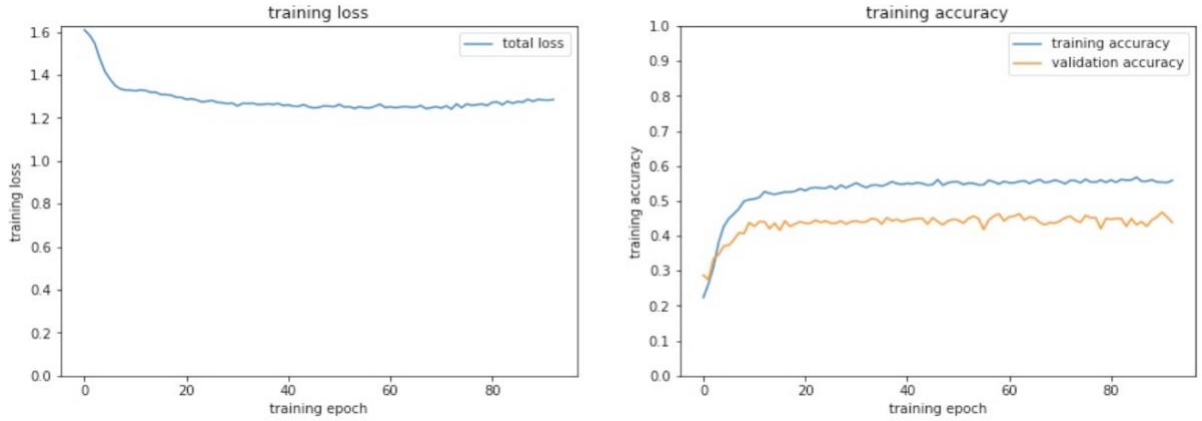

Figure 20: Training loss & accuracy and validation accuracy for LDA with 31,000 topics

```
testing accuracy   : 0.935%
validation accuracy: 0.888%
```

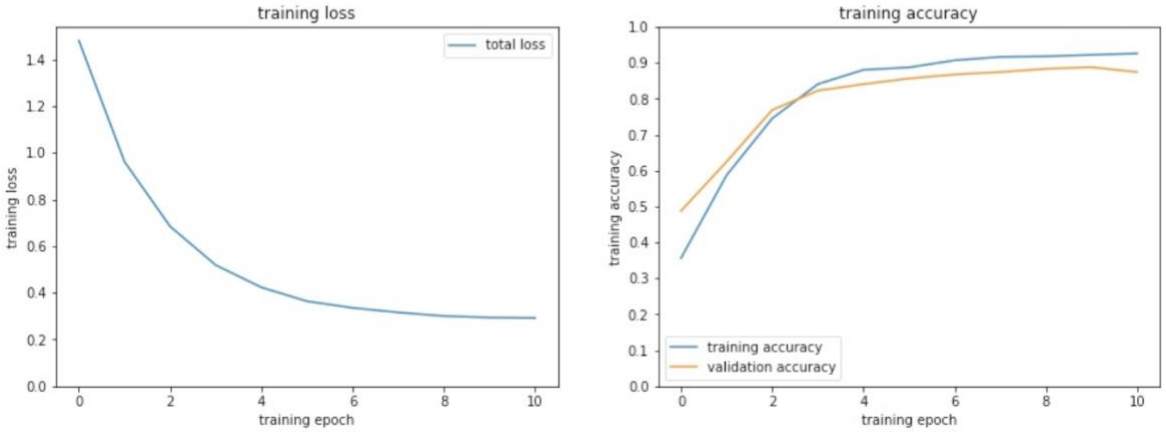

Figure 21: Training loss & accuracy and validation accuracy for LDA with 2,000 topics

```
training accuracy  : 0.966%
validation accuracy: 0.980%
```

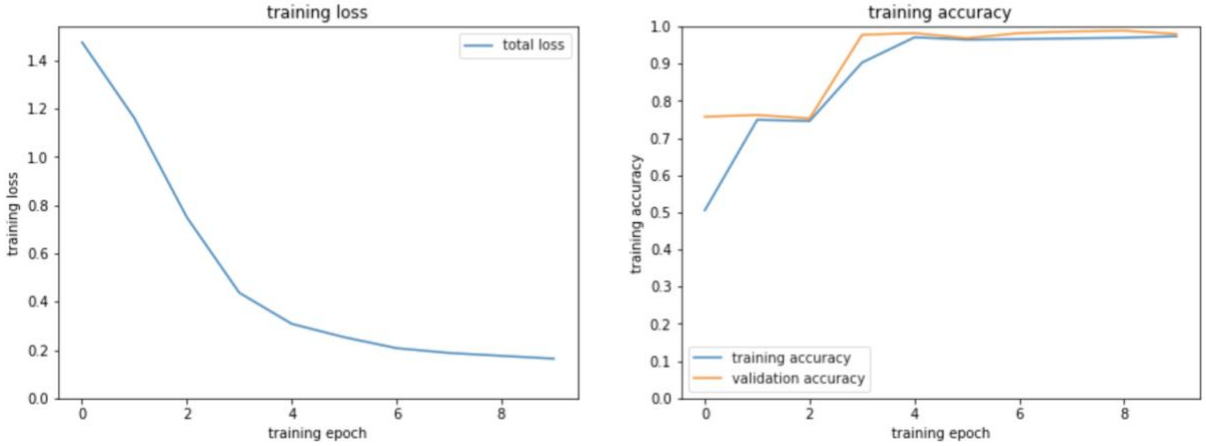

Figure 22: Training loss & accuracy and validation accuracy for LDA with 1,000 topics

6. LDA Effectiveness Plots

Before moving on to final conclusions, the following figures shows the LDA model learning and how topic modelling could be used for other purpose than just topic modelling. Also, it must be noted that some of the topics are clearly intuitive to humans based on the words representing the topic, whereas, some other topics which are rightly classified may not be clearly intuitive. This is still a big leap compared to visualizing the understanding of a neural network and so is a great advantage of LDA in terms of tuning the model for better performance.

For the following two visualizations, topic number 12 and 7 were selected for visualization of the model understanding respectively from the InterTopic distance map showcased in Figure 14.

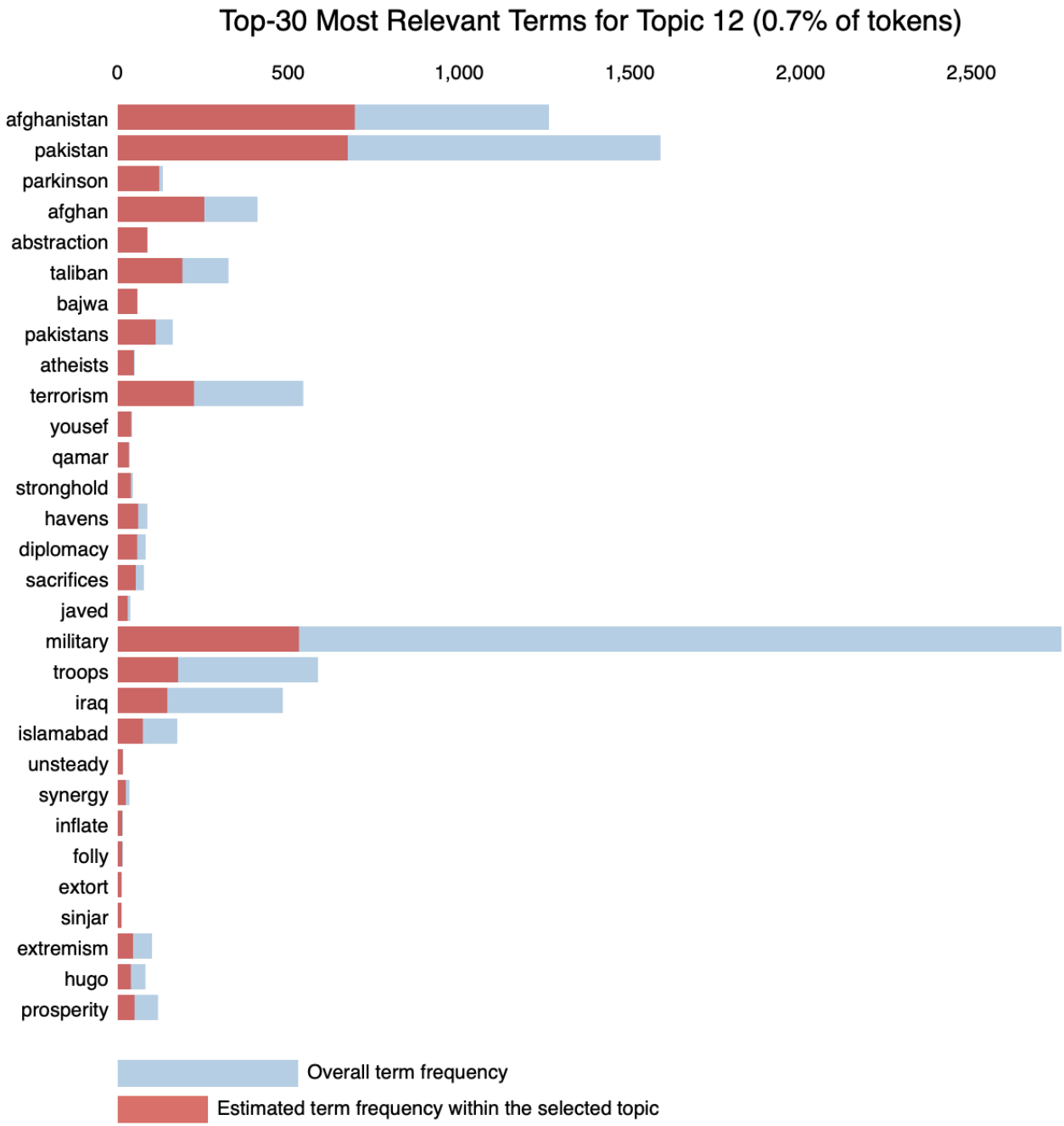

Figure 23: Word contribution of topic 12 from the topic distribution

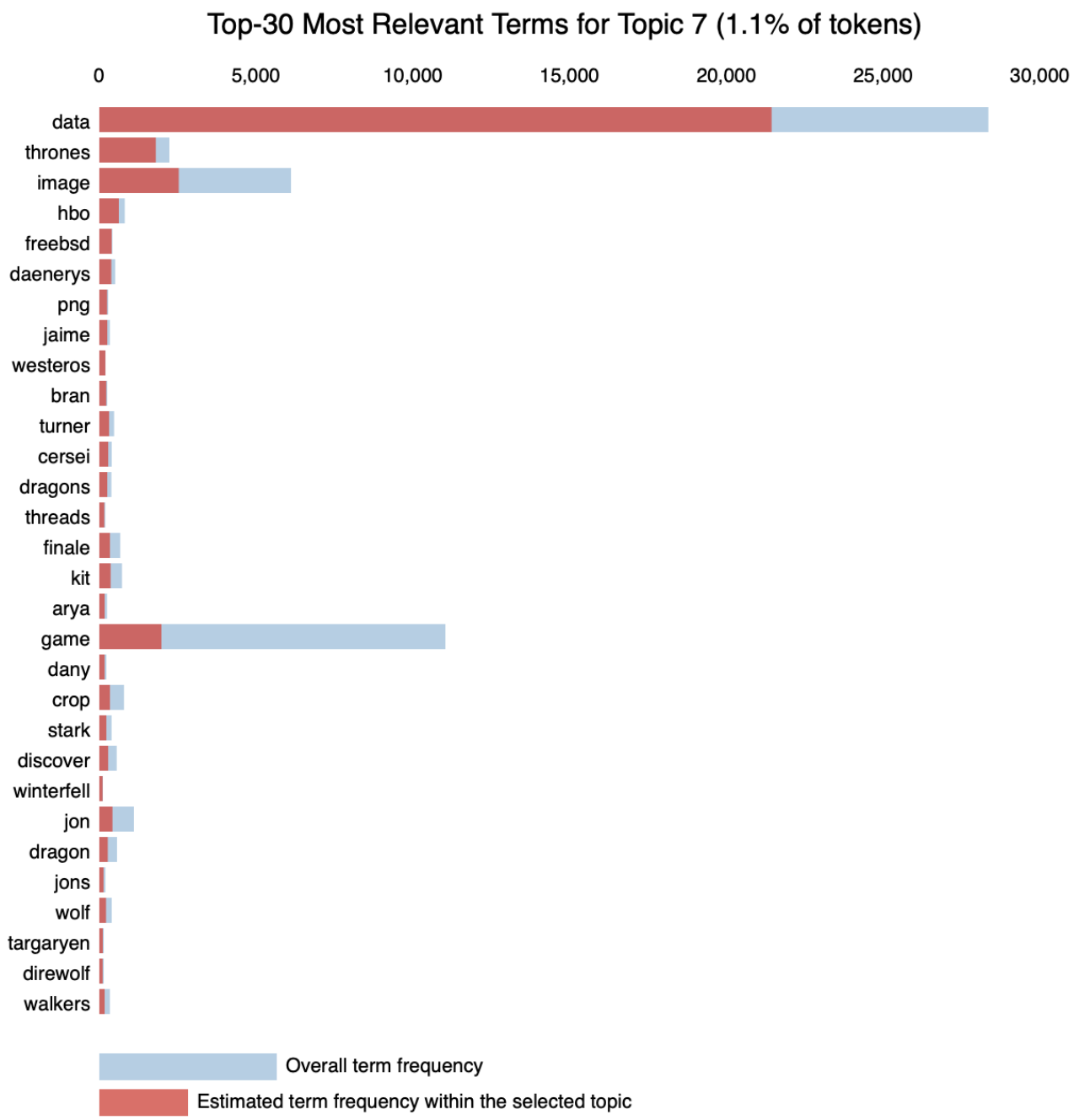

Figure 24: Word contribution of Topic 7 from topic distribution

It can be seen from the above example; this topic modelling could also be used for curating content online which is considered to spread rumor or disdain for a certain act. Such regulations are already implemented using various methods, but it is worth noting that the approach using LDA does not rely on any information other than the text itself. This is what makes it truly independent and powerful.

One other important point to note about content moderation is need of a stimulus – the 'report objectionable' stimulus is required form a user for something to be flagged and checked by various

techniques and humans. However, with an automated scraper, this model could detect objectionable content like provocative / offensive / demeaning / derogatory, etc. even without an actionable stimulus. Also, it should be able to do that irrespective of the source and for unseen content based on the assumption of having learned corpus library. This makes it a potent solution even for zero-day problems of this type.

VI. Conclusion

According to the problem definition specified in the earlier sections, we have accomplished the following. The proposed machine learning framework

- Has more representative and identifying power since it is a combination of supervised and unsupervised algorithms.

- Works well with text documents that have topics or interests intermingled and the central themes of the document could be argued to be more than 1.

- Does not strictly adhere to predefined categories as it was successfully able to model user interests which did not have specific predefined categories other than the basic guiding principle of a user's liking or disliking. However, when given predefined categories, the proposed framework performs equally well in categorical classification of the documents.

- Has ability to predict a topic mixture for each document. Topic mixture represents the presence of each identified topic in the document.

- Has ability to reasonably generalize well to unseen content with respect to user's interest or topic categories. This is based on the assumption that text content is present in the corpus brain of the model. A simple analogy could be thought of based on human mind. A human can only categorize a topic if it is aware of the meaning of the words that it sees. It is for this reason, that this assumption is not held as a drawback.

- Has ability to understand and tweak the number of words contributing to a topic. This is crucial to tune topic modelling so that neither too less nor too many words contribute to form the meaning of a topic, thus, avoiding the problem of too broad vs too narrow topics.

- Does not rely on any meta information like the source of the content, medium of delivery of the content, authenticity and reliability of the content. This is a vital merit of this framework.

- Does not derive meaning or context based on the source, size, length, formatting features or headline text of the document. The model learning is completely based on the text content irrespective of these meta factors about the text.

A. Future Work

This model can be used for some purposes which may not seem obvious initially while looking at the problem definition.

One such case is content moderation. If there is set of topics / articles that are flagged as hateful / derogatory / strongly disrespectful, then articles are generally taken down when someone on the internet reports those articles as a problem. Also, there are tools in place which can keep a watch on such flagged online sources for moderation. However, every time a new source creates an article that is offensive / provocative or any other objectionable category, this framework could target such articles just based on the latent understanding of topic modelling that it has gained. This way, we can solve the zero-day problem for content moderation. Also, we can eliminate the need for stimulus of 'reporting content' before an action needs to be flagged.

Other future work could be to make the model even better for real time use by creating labels for intermediate topic categories. Each document that is being classified, has a set of topic mixtures. These topic mixtures are words which give an understanding of a topic. This understanding if labelled for all these topic mixture words and labelled manually/programmatically, we can achieve much more human like interactions based on this topic modelling framework.

One last aspect is trying to test this framework with use of minimal corpus. It would be interesting to see if this framework performs equally well when the corpus learning is not thorough or is limited in extent.

References

1. D. M. Blei, A. Y. Ng, M. I. Jordan, "Latent Dirichlet allocation", *Journal of Machine Learning Research*, vol. 3, no. 4-5, pp. 993-1022, 2003.

2. T. Hofmann, "Probabilistic latent semantic indexing", *Fifteenth Annual Conference on Uncertainty in Artificial Intelligence*, pp. 289-296, 1999.

3. C. Hsu and C. Chiu, "A hybrid Latent Dirichlet Allocation approach for topic classification," *2017 IEEE International Conference on Innovations in Intelligent Systems and Applications (INISTA)*, Gdynia, 2017, pp. 312-315.

4. P. C. Kaur, T. Ghorpade and V. Mane, "Extraction of unigram and bigram topic list by using Latent Dirichlet Markov allocation and sentiment classification," *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, Chennai, 2017, pp. 2332-2339.

5. C. Wartena and R. Brussee, "Topic Detection by Clustering Keywords," *2008 19th International Workshop on Database and Expert Systems Applications*, Turin, 2008, pp. 54-58.

6. L. AlSumait, D. Barbará and C. Domeniconi, "On-line LDA: Adaptive Topic Models for Mining Text Streams with Applications to Topic Detection and Tracking," *2008 Eighth IEEE International Conference on Data Mining*, Pisa, 2008, pp. 3-12.

7. W. Xie, F. Zhu, J. Jiang, E. Lim and K. Wang, "TopicSketch: Real-Time Bursty Topic Detection from Twitter," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 8, pp. 2216-2229, 1 Aug. 2016.

8. Q. He, K. Chang, E. Lim and A. Banerjee, "Keep It Simple with Time: A Reexamination of Probabilistic Topic Detection Models," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 10, pp. 1795-1808, Oct. 2010.

9. Ibrahim, R., Elbagoury, A., Kamel, M.S. et al. Knowl Inf Syst (2018) 54: 511.

10. L. Chen, D. Tu, M. Lv and G. Chen, "A Knowledge-Based Semi Supervised Hierarchical Online Topic Detection Framework," in *IEEE Transactions on Cybernetics*.

11. Q. Zhou, L. Shi, L. Xu and W. Liu, "An Improved Single-Pass Topic Detection Method," *2018 10th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*, Changsha, Hunan, China, 2018, pp. 317-320.

12. Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, "Efficient Estimation of Word Representations in Vector Space", *Proceedings of International Conference on Learning Representations ser. ICLR '13*, 2013.

13. Omer Levy, Yoav Goldberg, "Dependency-Based Word Embeddings", *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers)*, pp. 302-308, June 23–25, 2014.

14. Figure Source: https://www.oreilly.com/learning/capturing-semantic-meanings-using-deep-learning Dated: May 18, 2019

15. Figure Source: https://cs.stanford.edu/~quoc/le/paragraph_vector.pdf

16. Figure Source: https://en.wikipedia.org/wiki/Latent_Dirichlet_allocation Dated: May 18, 2019

17. Figure Source: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47 Dated: May 18, 2019

18. Dataset Source: (MBTI) Myers-Briggs Personality Type dataset, Retrieved from :[https://www.kaggle.com/datasnaek/mbti-type] Dated: May 18, 2019

19. Dataset: Rohit Kulkarni (2018), One Week of Global Feeds [News CSV Dataset], doi:10.7910/DVN/ILAT5B, Retrieved from: [https://www.kaggle.com/therohk/global-news-week] Dated: May 18, 2019

20. Dataset: BBC, "Consists of 2225 documents from the BBC news website corresponding to stories in five topical areas from 2004-2005.", D. Greene and P. Cunningham. "Practical Solutions to the Problem of Diagonal Dominance in Kernel Document Clustering", Proc. ICML 2006., http://mlg.ucd.ie/datasets/bbc.html

21. Dataset: User Interest Dataset curated from two sources. Unsupervised Learning, Daniel Miessler https://us8.campaign-archive.com/home/?u=6a9e465ab1570df8aaecb2292&id=49fdb7d723 and Casual Spectator Sports, Casual Spectator, https://us9.campaign-archive.com/home/?u=fd29bb4bf60ba78d658031aba&id=2e9018b243