2021

Feature Augmentation for Improved Topic Modeling of Youtube Lecture Videos using Latent Dirichlet Allocation

Nakul Srikumar

Technological University Dublin

Feature Augmentation for Improved Topic Modeling of Youtube Lecture Videos using Latent Dirichlet Allocation

Nakul Srikumar

A dissertation submitted in partial fulfilment of the requirements of
Dublin Institute of Technology for the degree of
M.Sc. in Computing (Data Analytics)

January 2021

Declaration

I certify that this dissertation, which I now submit for examination for the award of MSc in Computing (Data Analytics), is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

This dissertation was prepared according to the regulations for postgraduate study of the Dublin Institute of Technology and has not been submitted in whole or part for an award in any other Institute or University.

The work reported on in this dissertation conforms to the principles and requirements of the Institute's guidelines for ethics in research.

Signed: Nakul Srikumar

Date:05 January 2020

Abstract

Application of Topic Models in text mining of educational data and more specifically, the text data obtained from lecture videos, is an area of research which is largely unexplored yet holds great potential. This work seeks to find empirical evidence for an improvement in Topic Modeling by pre−extracting bigram tokens and adding them as additional features in the Latent Dirichlet Allocation (LDA) algorithm, a widely−recognized topic modeling technique.

The dataset considered for analysis is a collection of transcripts of video lectures on Machine Learning scraped from YouTube. Using the cosine similarity distance measure as a metric, the experiment showed a statistically significant improvement in topic model performance against the baseline topic model which did not use extra features, thus confirming the hypothesis.

By introducing explainable features before modeling and using deep learning based text representation only at the post−modeling evaluation stage, the overall model interpretability is retained. This empowers educators and researchers alike to not only benefit from the LDA model in their own fields but also to play a substantial role in efforts to improve model performance. It also sets the direction for future work which could use the feature augmented topic model as the input to other more common text mining tasks like document categorization and information retrieval.

Keywords: Latent Dirichlet Allocation, Cosine Similarity, Bigrams, Word2Vec, YouTube, Topic models

Acknowledgments

First of all, I would like to thank my supervisor, Dr.Svetlana Hensman, for her valuable and timely inputs to guide this thesis to completion. Her questions and perspectives helped give proper shape to this work.

My sincere appreciation to Dr.John Gilligan and Dr.Luca Longo who, through their taught modules on scientific research process and research proposal writing, introduced a complete novice like me to the world of research and gave systematic guidance up to producing a formal research proposal. I carry many fond memories of their classroom lectures.

Finally, special gratitude to my parents for their unstinted support and encouragement. I humbly dedicate this work to them.

Contents

List of Figures

List of Tables

List of Acronyms

TM	Topic Models or Text Mining
LDA	Latent Dirichlet Allocation
CRISP-DM	Cross Industry Standard Process for Data Mining
KNN	K-Nearest Neighbour
ML	Machine Learning
DL	Deep Learning
SVM	Support Vector Machine
MCMC	Markov Chain Monte Carlo
LSTM	Long Short Term Memory
CNN	Convolutional Neural Network
SVD	Singular Value Decomposition
PCA	Principal Component Analysis
LL	Log-Likelihood
PMI	Point-wise Mutual Information
NPMI	Normalized Point-wise Mutual Information
TC	Topic Coherence
CS	Cosine Similarity
TF-IDF	Term Frequency - Inverse Document Frequency

Chapter 1

Introduction

1.1 The background

The digital era has made text books and classroom learning almost irrelevant, as vast amounts of high−quality learning resources are now available online. The rise of the internet and social media has led to incredible amounts of valuable knowledge being shared through videos, blogs, e−mails, community discussion forums, chats, social networks, etc. This is also reflected in increased enrolments in online courses. A report by (Snyder, De Brey, & Dillow, 2018) revealed that the percentage of students taking one or more online undergraduate classes increased from 15.6% in 2004 to 43.1% in 2016. Furthermore, the percentage of undergraduate students taking fully online degree programs increased from 3.8% in 2008 to 10.8% in 2016, and the percentage of graduate students who took entirely online graduate (postgraduate) degree programs has increased from 6.1% in 2008 to 27.3% in 2016.

Online education is also the logical choice for mature learners who are expected by their employers to constantly re−skill themselves to keep up with the technological changes at the workplace. The COVID-19 pandemic too has further accelerated the adoption of online learning as it is increasingly being viewed as the only safe and viable option for education continuity. Under the circumstances, there is an urgent need to mine the vast volumes of data generated online to develop applications and systems which would better support instructors and students in the learning process.

1.2 Research focus

The bulk of online learning data is unstructured text in the form of blogs, discussion threads, online writing assignments, Q and A forums, chats, lecture notes, wiki pages, book PDFs, etc. Video lectures are the only noteworthy exception, but have received negligible attention for analysis tasks (Ferreira-Mello, André, Pinheiro, Costa, & Romero, 2019). This was one of the motivations to focus on analysing video content.

Although videos are a very popular reference for students, it can be discouraging for them to have to sit through long videos only to realize that it is not relevant for their personal learning needs. Here, it becomes important to provide proper tags for videos which can make the search engine results more tuned to an individual's unique learning requirements.

In addition, one can safely assume that the video title provided by the content creator will be broad and indicative in nature and cannot cover the breadth of concepts actually discussed within the video. However, it is quite impractical to manually annotate every video transcript to bring out every concept/theme discussed in the lecture/tutorial. Consequently, one has to depend on machine learning techniques like the topic modelling algorithms of text mining to accomplish this task at scale.

The quality of the model's output, the learned topics, merits due consideration. There are many ways to evaluate models and improve their performance. One is to run a suite of algorithms and choose the best for the task at hand. Another is to tweak the model architecture to look for the ideal model settings. Feature engineering is yet another means to improve model performance. Feature engineering essentially involves looking for the optimum representation of the input data which is fed into the machine learning model.

The 'engineering' can either be an alternative representation of the input features, or adding additional relevant features, or even removing extraneous features. In this work, feature engineering will be done by adding relevant features and then the model will be checked for improvement in performance.

1.3 Research problem

The main focus of this work is to address the following research question:

"To what extent does feature augmentation with bigram features impact the similarity between an LDA topic model's algorithm-generated topics and the corresponding human-labelled titles, when applied on a text corpus consisting of transcripts of YouTube videos on machine learning?"

This would also entail looking into the following smaller problems at various stages of the experiment:

Sub-Question A − What is the optimum number of topics in the given dataset?

Sub-Question B − How to represent the topic words to be able to use similarity measures?

Sub-Question C − How to arrive at the optimum number of bigrams?

Sub-Question D − What statistical tests can be applied to measure the impact?

The research question is restated in the following section as an experimental hypothesis.

1.4 Research Objectives

On the basis of the *research focus* section which highlights the specific area of research, and the *research problem* section which encapsulates the research focus as a research question, the Null and Alternative Hypotheses for this thesis are as follows:

Null Hypothesis: For a text corpus of transcripts of youtube lecture videos on machine learning, using additional features in the form of bigram word tuples does not change the mean cosine similarity between the LDA model's generated topics and the actual video titles, compared to that of the baseline LDA model without the additonal features.

Alternative Hypothesis: For a text corpus of transcripts of youtube lecture videos on machine learning, using additional features in the form of bigram tuples increases the mean cosine similarity between the LDA model topics and the actual video titles, compared to that of the baseline LDA model without additional features.

1.5 Research methodology

The research conducted in this project is *secondary* as it is based on a text corpus scraped from the youtube website using a Python programming script. The nature of this research is that it is *quantitative*, because the text data will be converted into a numerical variable for analysis. The research type is *empirical* as the hypothesis is articulated beforehand and empirically tested by a suitable experiment. The reasoning is *deductive* as it starts with a hypothesis and goes on to prove or disprove it on the basis of experimental evidence.

The research work broadly follows the Cross Industry Standard Process for Data Mining (CRISP−DM) framework. The different chapters of this thesis can be fairly mapped to the CRISP−DM framework as follows: The 'literature review' in Chapter 2 is like the *Business Understanding* phase of the CRISP−DM framework. Chapter 3 on 'Experiment Design and Methodology' is like the *Data Understanding* and *Data Preparation* phase. Chapter 4 on 'Results, Evaluation, and Discussion' is equivalent to the framework's *Data Modeling* and *Model evaluation* phases. The sections 'Contributions and Impact' and 'Future work and Recommendations' of Chapter 5 'Conclusions' can be considered the *Model Deployment* phase of CRISP−DM.

1.6 Scope and Limitations

The experiment scope is defined by the area of research focus in terms of the dataset, the topic model chosen, the evaluation metric chosen, and the deployment of the model.

For the current work, the dataset is from a specific source - lecture video on youtube. Also, the videos are on a particular field of study - Machine Learning. The topic model algorithm is restricted to the Latent Dirichlet allocation (LDA) which gives out topics in a form that is human understandable. The evaluation metric considered for the study is the Cosine Similarity distance measure. There are other distance measures which can used in any future work. For the current study, the 'deployment' is restricted to the analysis stage. It is not integrated into any education related app or

software. The task of integrating it into a real-life personalised learning application which uses a topic−model based recommender−system can be the subject of a future project.

A possible limitation of the experiment could be the size of the corpus − in this case 179 videos from two machine learning playlists on youtube. It remains to be seen how the model behaves for very large datasets, especially when considering additional features for such datasets.

Moreover, the corpus for the current study is drawn from playlists meant to cover a typical undergraduate module in Machine Learning where the instructor usually has to rush through many topics. So, an assumption was made that many unrelated topic themes could be obtained . This assumption is acceptable for a relatively smaller corpus like the one used for this study. This may not hold true for larger corpuses which have higher word co−occurence counts and a lot more contexts in which the words can co−occur. As a result, the model may generate many topics which are not clearly distinguishable. An extension of the LDA, called the Correlated Topic Model (CTM) may be more appropriate.

1.7 Document Outline

Presented below is a chapter−wise outline of the content covered in the rest of the thesis document.

Chapter 2 − Review of existing literature: This chapter offers a comprehensive look into the state−of−the−art in topic models, be it in terms of its working, model variants, applications in education and otherwise, successes, evaluation metrics, and strengths and limitations. Being fully based on the research literature, it can serve as an authentic introduction to topic models and inspire readers to pursue further research in this area.

Chapter 3 − Experiment Design and Methodology: Pertinent details like experiment design, stages, work−flow can be found here. Various aspects like corpus creation, data pre−processing, feature engineering are elaborated upon. Also the

methodologies adopted for evaluating and comparing model performance are discussed in detail.

Chapter 4 – Results, Evaluation, and Discussion: This chapter describes the actual carrying out of the experiment. It focuses on model training, tuning, and model performance and documents the results obtained after implementation. Here, the baseline and enhanced features model are compared for evaluating performance improvement, if any. The results are analysed and the strengths and limitations of the findings discussed.

Chapter 5 – Conclusion: This chapter gives a brief summary of the thesis, while highlighting the uniqueness of this work and how it contributes to the body of knowledge. Furthermore, suggestions for possible directions of future work are offered.

Chapter 2

Review of existing literature

This chapter gives an overview of the literature survey carried out. Text mining, as applied to the education domain is first considered. Topic models and their variants, both traditional and modern deep learning based, are discussed. Various applications of topic modeling are briefly reviewed, followed by a survey of the the model evaluation methods recommended in the research literature. The successes and limitations of these methods are brought out , leading to a justification for the present work.

2.1 Text Mining in Education

With the advent of the internet, online search engines and social media, online education was often viewed as a viable alternative to conventional classroom education in terms of reach and scale. There exists a wide range of platforms to support online education, such as Adaptive and Intelligent Educational Systems(AIES), web-based Intelligent Tutoring System(WITS), Learning Management System(LMS), Massive Open Online Courses(MOOCs). With the COVID−19 global pandemic since early 2020, widespread adoption of these technologies has become an unavoidable reality.

The large volume of structured and unstructured data produced in these platforms requires expert management and analysis. The data comes from different sources and in different formats, such as discussion forums, chats, Wikipedia, blogs, open Q and A, videos, etc. According to (Valjataga, T; Poldoja, 2011), processing and using this

information in a manner than aids instructors and students in the learning process is a challenge. Educational Data Mining(EDM) and Learning Analytics(LA) techniques have been recently employed to address this issue(Romero & Ventura, 2017)

Although effective, EDM and LA, for the most part, do not directly explore the educational resources available. Instead, they are focused on validating pedagogical theories by analyzing student demographics, course engagement, and performance. Text mining techniques, if adopted, can fill in the need because they are suitable for extracting valuable insights from the educational material itself. Hence, the new generation of online platforms could benefit from text mining techniques such as Natural Language Processing(NLP), text classification and clustering, information retrieval, and text summarization (Ferreira-Mello et al., 2019). Topic models are a branch of text mining which have an essential, but often understated, role to play in each of these text mining techniques.

2.2 Topic Modeling

According to (Ignatow & Mihalcea, 2017), Topic Modeling(TM) involves automated procedures for encoding a collection of texts in terms of meaningful categories that represent the main topics being discussed in the text corpus. True to its name, a topic model gives out "topics", collections of words that make sense together. The common TM techniques are Latent Dirichlet Allocation(LDA), Latent Semantic Analysis(LSA), and Non-negative Matrix Factorization(NMF)(Boyd-Graber, Hu, & Mimno, 2017; Pauca, V. P., Shahnaz, F., Berry, M.W., Plemmons, 2004)

LSA is derived from linear algebra (Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., Harshman, 1990) and it aims to find the best low rank approximation of a document−term matrix using Singular Value Decomposition(SVD). The document-term matrix is a way to represent the corpus wherein the rows and columns represent the individual documents and the unique terms(vocabulary) of the corpus respectively, and the cell values are the term weights for each document viewed row wise. Figure (2.1) shows a way of representing this document−term matrix as matrix

product of document−topic and topic−term matrices. The output of the LSA topic model is a set of principle factors in the topic space and does not translate into human recognizable topics.The LSA model is deemed to have an unsatisfactory statistical foundation and suffers from computational complexity (Karypis & Han, 2000; Wei, 2007)

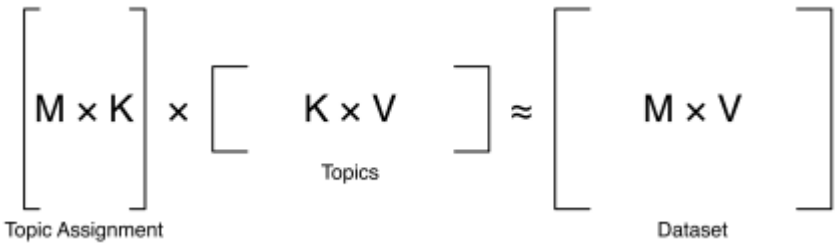

Figure 2.1: Matrix formulation of finding K topics for a dataset with M documents and V words

LDA is a generative, probabilistic topic model which is being widely used ever since it was first introduced by (Blei, D. M., Ng, A. Y., Jordan, 2003).These topics are also known as 'latent variables'. They are called so because they are not directly observable in the data sample available for learning, yet are intuitively believed to influence the sample. LDA treats the text as a 'bag of words'(BOW) which wholly disregards syntax, context, and location. The model is generative, as it imagines any given document as manifesting from a generative process supposedly followed by the document's real−life author, the steps of which are listed below:

1. First decide on the length N of the document
2. Then set the document's topic proportions (assuming that the K topics are found in varying proportions in each document. This is quite natural, given that a fixed set of themes/topic usually permeate the entire corpus)
3. For each of the N words in the document, the author follows the steps below:
    - Choose a topic from the K topics.
    - Then choose a word from the Vocabulary V

Keeping with the bag of words assumption, the number of words allowed for any given topic is decided by the topic's allocated proportion within the document, and how frequent a particular word occurs is determined by the strength of its association with that topic. The objective of the LDA model is to infer the parameters of the LDA algorithm that has "generated" the actual final text (the LDA process is the mathematical equivalent of a real−life author). The output of this inference process is a set of K topic−word distributions and a set of M document−topic distributions.

2.3 LDA model and its variants

(Hofmann, 2001) introduced the probabilistic topic approach to document modeling in his Probabilistic Latent Semantic Indexing method (pLSI; also known as the aspect model). The pLSI model does not make any assumptions about how the document's topic allocation weights θ are generated, making it difficult to test the generalizability of the model to new documents.(Steyvers & Griffiths, 2010). Possessing fully generative semantics, LDA overcomes the drawbacks of previous topic models such as pLSI.(Wei & Croft, 2006) and is able to identify new topics when applied to additional documents (Blei, 2012)

2.3.1 Conventional LDA and its extensions

LDA, as previously mentioned, is a probabilistic generative model.The fundamental assumptions at the core of this algorithm are:

- A document is a normalized multinomial distribution over topics.
- A topic is a normalized multinomial distribution over words.

The normalized multinomial distribution refers to a vector of values which sum to 1.The Multinomial distributions are sampled from a Dirichlet distribution, analogous to sampling of the binomial distribution from a Beta distribution. LDA uses an iterative Bayesian *posterior* probabilistic inference algorithm to converge to stable model parameter values. Bayesian learning requires a *prior distribution* and the Dirichlet

distribution is very convenient for the purpose − it is in the exponential family and is conjugate to the multinomial distribution.

The plate notation of the LDA shown below (figure 2.2) is the standard model architecture.(Blei & Lafferty, 2009) The nodes represent random variables; the edges denote dependence between random variables. Shaded node is the observed random variable while the remaining are hidden. The plates (boxes in the figure) indicate repeated sampling with the number of repetitions given by the variable in the bottom.

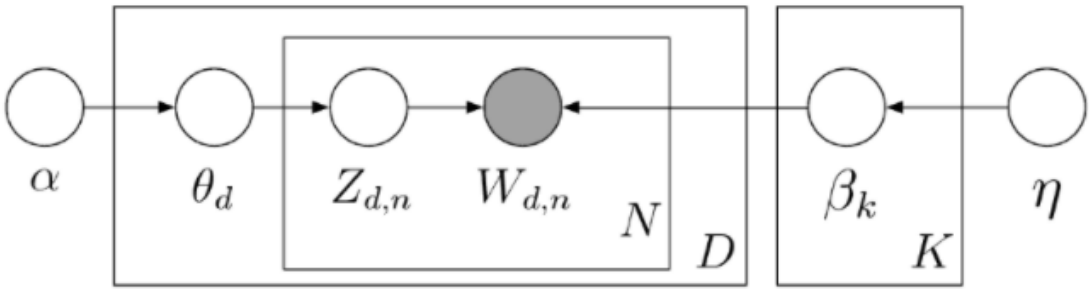

Figure 2.2: A graphical model representation of the LDA

$\alpha \rightarrow$ Concentration parameter of a Dirichlet distribution. A K−dimensional vector.
$\theta_d \rightarrow$ A K−dimensional vector representing topic allocations for document d. It is a normalized multinomial distribution sampled from the Dirichlet having parameter 'α'.
$Z_{d,n} \rightarrow$ Topic assigned to the n^{th} word of the document d of the corpus D. $Z_{d,n} \in (1, 2, 3, ..., K)$.
$\beta_k \rightarrow$ A topic−term normalized multinomial distribution for topic k $\in (1, 2, 3, ..., K)$. It is a V−dimensional vector drawn from a Dirichlet having parameter 'η'.
$W_{d,n} \rightarrow$ The n^{th} word of the document d.

The inference process for the LDA model is math heavy. It involves Gibbs sampling, a Markov Chain Monte Carlo(MCMC) method (Andrieu, Christophe; De Freitas, Nando; Doucet, Arnaud; Jordan, 2003) which greatly simplifies the iterative process and makes the otherwise intractable calculations elegant and intuitive (Hardisty & Resnik, 2010). A simple sequence of steps explaining the LDA algorithm without the math is given below:

1. Randomly initialize the necessary parameters

2. For each document, randomly assign to each word, one of the K topics. After this is done for all the words, run the following iterative process till the topic assignments stabilize.

3. For each document D:

   a. For each word W in document:

      (i) For each topic T:

      - Compute P(T|D), which is proportion of words in D assigned to topic T.
      - Compute P(W|T), which is proportion of assignments to topic T over all documents containing the word W
      - Word W is now associated with topic T with probability P(T|D) * P(W|T).

      (ii) After obtaining the K−dimensional vector of probability values from the 'for' loop in (i), sample the vector in order to re−assign a topic to the word W. This is done by normalizing the obtained vector, creating a 1−D array of K cumulative probability values, and sampling from the array.

Correlated topic model (CTM) is an extension of the LDA designed to deal with the LDA's inability to directly model the correlation between the occurrence of topics. It is quite common to have correlated latent topics, especially when the corpus is from a specific domain, say biology. A biology corpus is unlikely to have topic themes based on nuclear physics. Other than the normalization constraint that the topic proportions for a document sums to one, the components of the document−topic vector are largely independent. This *independence* seen in the LDA is a drawback for modeling corpuses where topics are expected to be correlated. The CTM process is identical to the LDA's generative process except that the topic proportions are drawn from a logistic normal distribution instead of a Dirichlet distribution.The logistic normal distribution allows

for a general pattern of variability between the components (Aitchison, 1982). The CTM uses the mean field variational inference algorithm for inference and it has been shown to perform better on held out data (Blei & Lafferty, 2007). However, the added flexibility and better fit on test data comes at a price−the inference process is slower than that of the LDA.

Dynamic Topic Model (DTM) is an extension of the LDA and CTM meant for capturing the evolution of topics in a sequentially organized corpus of documents. The LDA and CTM are order−agnostic for both the word order within the document and the document order within the corpus. This word and document *exhangeability* allowed in LDA and CTM make them inappropriate tools for topic modeling in many corpuses which are time−ordered and contain changing vocabulary and emphasis for the same topic over time. For e.g., the term 'independent variable' commonly used by 20^{th} century statisticians in the context of Multiple Linear Regression(MLR) may no longer be in vogue, and the term 'features' generally used by Machine Learning practitioners may be more popular instead. Thus, 'features' now acquires a technical connotation beyond its dictionary meaning and hence, it has an increased association with the same topic 'MLR'. In short, the topics and associated words evolve over time, and the DTM is used to uncover the dynamic nature of the underlying topics. In the DTM, the data is divided into time slices and each slice is modelled. Here too, the logistic normal distribution is used, but for modelling the variability of the time−series topics. This is an extension of the logistic normal to time−series simplex data (West & Harrison, 2006).

Bigram Topic Model (BTM) first proposed by (Wallach, 2006) attempts to take topic models beyond the 'bag of words'(BOW) approach. It incorporates $n-$gram statistics and latent topic variables by extending the standard unigram LDA topic model to include properties of a hierarchical Dirichlet bigram language model. When applied to 2 datasets of 150 documents each, one consisting of paper abstracts and the other of newsgroup postings, the BTM exhibited better predictive accuracy than either a hierarchical Dirichlet bigram language model or a unigram LDA topic model.

Assuming T topics and W words in the corpus, each topic T in the BTM is now

a set of W distributions. Hence, the BTM topic−term matrix is of size WT^2 as against the LDA's WT. Evidently, that makes the BTM model's inference process computationally expensive (Lau, Baldwin, & Newman, 2013).

2.3.2 Neural network based LDA variants

(Cao, Li, Liu, Li, & Ji, 2015) introduce a novel neural topic model(NTM) which is an ingenious representation of the standard LDA model. A supervised extension called sNTM is also explained which can tackle supervised tasks like labelled classification. The NTM seeks to overcome some of the inherent limitations of the standard LDA like imprecise prior knowledge and restricted unigram representation.

For a document d and a word w in d, topic models compute the conditional probability $p(w|d)$ as the combination of word−topic and topic−document distribution:

$$p(w|d) = \sum_{1}^{K} p(w|t_i)p(t_i|d) \tag{2.1}$$

where t_i is a latent topic and K is a pre−defined topic number. Let $\phi(w) = [p(w|t_1), ..., p(w|t_K)]$ and $\theta(d) = [p(t_1|d), ..., p(t_i|d)]$, where ϕ is shared among the corpus and θ is document−specific. Then Eq. 2.1 can be represented as the vector form:

$$p(w|d) = \phi(w) * \theta^T(d) \tag{2.2}$$

This formulation allows viewing the topic model as a neural network, where $\phi(w)$ functions as the look−up neural layer for words with the sigmoid activation function, and $\theta(d)$ works as a look−up neural layer for documents with the softmax activation function. The network output is calculated as the dot product of $\phi(w)$ and $\theta(d)$.

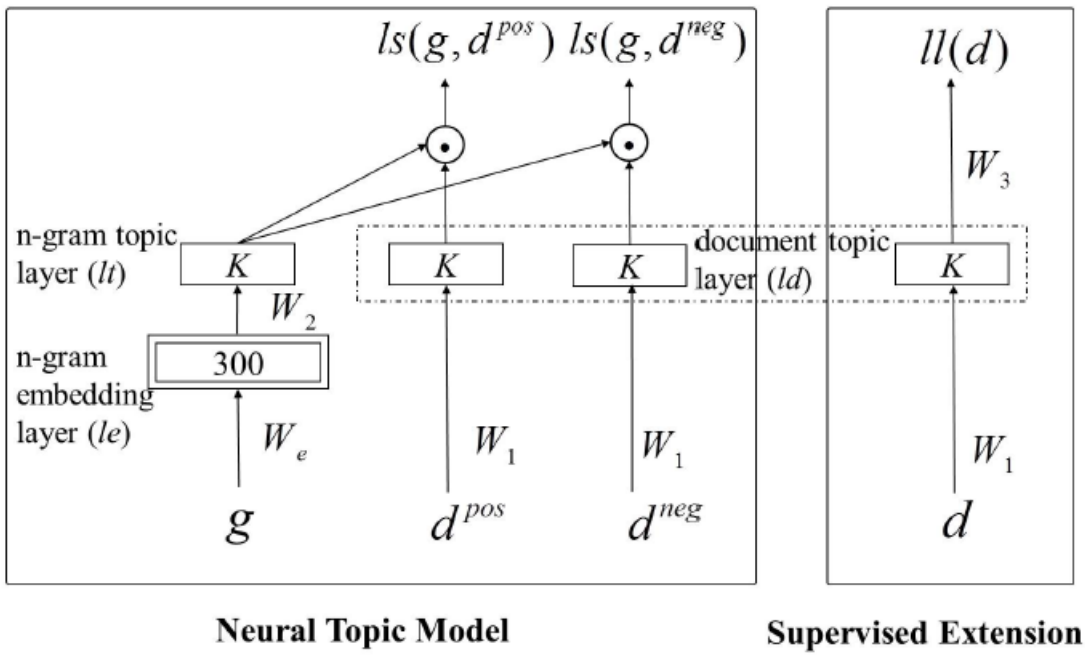

Figure 2.3: Unsupervised Neural Topic Model(NTM) and its Extension(sNTM)

The above figure shows the neural network architecture.The model accepts inputs (g,d), where g is an n-gram and d is the document ID. Using a pretrained word2vec model (trained on 100 billion words, word2vec provides 300-dimensional embedding for about 3 millions words or phrases), the n-grams can be represented as either a 300-dim embedding directly (if g is in the word2vec trained vocabulary) or as a sum of the n individual word embeddings. (Mikolov, Tomas; Sutskever, Ilya; Chen, Kai; Corrado, Greg S; Dean, 2013).

The weights W_e are not updated during the network training but kept fixed. W_2 is the weight matrix between the embedding layer and the topic layer. As mentioned before, the topic layer (lt) is the look up neural layer for word$-$topic distribution $\phi(w)$ and it has a sigmoid activation function. The document topic layer (ld) is the look up neural layer for the document$-$topic distribution $\theta(d)$ having the softmax activation function to enforce the probabilistic constraint. W_1 is a $|D| \times K$ look up matrix to convert document d to a suitable vector representation on which the softmax activation is applied. The scoring layer (ls) is obtained by the dot products of $lt(g)$ and $ld(d)$. It represents $p(g|d)$ of Eqn. 2.2 (g in place of w).

The training is done by the usual back−propagation (BP) algorithm wherein the weights W_1 and W_2 are randomly initialized and updated via stochastic gradient descent. As the network is intended to mimic a topic model, the objective/loss function is chosen accordingly. The equation for the cost function which is sought to be minimized is:

$$c(g, d^{pos}, d^{neg}) = max(0, \Omega - ls(g, d^{pos}) + ls(g, d^{neg})) \tag{2.3}$$

d^{pos} is a document containing g, and d^{neg} is a randomly sampled document not containing g. The idea behind the choice of the cost function is to achieve a margin of atleast Ω (experience value is 0.5) between the scores $ls(g, d^{pos})$ and $ls(g, d^{neg})$. This is in line with the intuitive expectation that a model which has learned well will score considerably higher for a document which actually contains the n-gram g vis−a−vis a document which does not.

For the supervised variant sNTM, the architecture includes a label−layer(ll) on top of the topic−document layer, parallel to the scoring layer. The output $ll(d)$ is computed as follows:

$$ll(d) = f(ld(d) \times W_3) \tag{2.4}$$

where the matrix W_3 denotes the weights of each topic contributing to the label score. $f(.)$ indicates a suitable activation function depending on the label property. In the sNMT, the BP algorithm updates W_1 and W_3 to minimize the label error.

On applying the sNMT, NMT, and LDA in a multiclass classification problem using the 20 Newsgroups dataset which contain more than 20,000 organized in 20 classes, the sNMT model performed consistently better, achieving an average accuracy of 0.76 over all classes for 100 topics and higher.

When applied to a multilabel classification task on the Wiki10+ dataset having 25 most frequent social tags and more than 17,000 documents with 2.6 tags per document on average, the sNTM attained Macro−F_1 and Micro−F_1 scores (Lewis, Yang, Rose, & Li, 2004) of 0.65 and 0.66 respectively, again outperforming the other models. The results for both tasks show that supervised topic models show an explicit advantage over the unsupervised ones.

2.4 Applications in Education

(Basu, Subhasree; Yu, Yi; Singh, Vivek K; Zimmermann, 2016) describe the design of a system *Videopedia* for recommending lecture videos for educational blogs using LDA topic modeling. Topic models are used to map videos (closed caption video transcripts) and blogs in the common semantic space of topics. After matching the videos and blog, the videos with high similarity values are recommended for the blog. The *Videopedia* system uses initial pruning by matching video and blog metadata, choosing top ranked matches and only then applying LDA. The dataset consisted of 3000 videos on 8 STEM related categories from the youtube channel for National Programme on Technology enhanced learning (NPTEL) and 1000 Wikipedia pages to which videos are to be recommended real-time.The ground truth of whether a video is relevant or not was created by checking if the wiki category of the Wiki page matches with the subject in the video's metadata. The results show that their system did better than the direct LDA (without initial pruning), and other algorithms like Vector space model(VSM − which uses TFIDF for matching) and probabilistic Latent Semantic Analysis (pLSA) on metrics like precision, recall and F−score. The *Videopedia* system performs best if initial pruning selects only the top 10% ranked videos.

(Distante, Cerulo, Visaggio, & Leone, 2014) apply LDA topic models, in conjunction with formal concept analysis, on online discussion forums of a Moodle Learning Manage-ment system (LMS) to determine the discussion topics and the hierarchical relationships between them. The individual forum messages and discussion threads are then matched with the obtained hierarchy of topics and are ranked by similarity scores and suitably tagged. This topic−driven navigation structure was seen to improve information search tasks of the LMS search function. Two of the four researchers involved in this study created a list of 11 specific search tasks with predefined search goals, i.e., the number of posts the user wants to retrieve. The effectiveness of the new navigation system with respect to the baseline full text search system was checked by the other two researchers. The criteria were (i) the number of items (forum posts) the user had to inspect to satisfy the information need and (ii) the time spent to accom-

plish the task. The topic-driven navigation system on average needed a lesser number of inspections compared to the full text search (9 vs 14) and it also took lesser time to complete the task (133 vs 170 seconds).

2.5 Other applications

Topic modeling has been applied in a wide range of disciplines to obtain insights. The field of digital humanities is dominated by topic modeling methods. It all started in 2010 with a widely circulated blog post on topic modeling. Since then, humanities scholars have used topic models in studies of themes in $19^{th}-$century literature (Jockers & Mimno, 2013), the history of literary scholarship (Goldstone & Underwood, 2012), and many other subjects.

Journalism researchers (Günther & Quandt, 2016) laid out a road map of text mining techniques available for journalism research. This includes rule−based approaches, dictionaries, supervised machine learning, document clustering, and topic models. (Jacobi, Van Atteveldt, & Welbers, 2016) used LDA topic modeling to study the *New York Times* coverage of nuclear technology between the years 1945 (end of WWII) to 2016, replicating a previous study. They demonstrated the utility of LDA for quickly analyzing news content in large digital news archives.

Political scientists, who are intrigued by different political phenomena, have found the topic models very handy for analysis. For instance, (Quinn, Monroe, Colaresi, Crespin, & Radev, 2010) used the R programming language to analyze topics in Senate floor speeches made between 1997−2004. The corpus consisted of more than 118,000 speeches got from the *Congressional Record*. They were able to estimate the topic substance, associated keywords, and the nesting hierarchy of topics. (Gerrish & Blei, 2012) have developed several predictive models linking legislative sentiments to legislative texts and have used these models to predict voting patterns with high accuracy. (Soroka, Stecula, & Wlezien, 2015), working on a dataset drawn from 30,000 news stories in the US spanning three decades, used topic model to explore the relationship between the media, economy, and public opinion. The results indicate that

media coverage reflects change in the future economy, and this influences and is influenced by public opinion. The patterns of their results give a plausible explanation for the surprising finding of positive coverage and public assessment seen even in the midst of a massive nation−wide crisis like the Great Recession.

Topic models have also been put to good use in other studies. (Gurcan & Cagiltay, 2019) used LDA topic modeling on a dataset comprising online job advertisements posted on an online employment site. The purpose of the study was to create a systematic competency map of the essential knowledge, skill set, tools, and capabilities sought by recruiters in the Big Data Software Engineering(BDSE) field. A total of 2,638 ad posts made between May−July 2018 were scraped using an API developed by *indeed.com*. The results showed that Java, Python, and Scala are the most sought−after programming languages; Jenkins, Maven, and Spring MVC the most demanded coding tools; Hive, SQL, Hbase, NoSQL are the most demanded databases; Hadoop and Spark are the big data tools the most in demand; and finally, Java + Python + Hive, Java + Python + Scala, Hadoop + Spark + Hive are the most popular tool combinations. The study also showed some important industry trends like a) shift from software engineering to data science roles b) dominance of cloud-based services and applications c) transition from databases to datawarehouses d) shift to real−time and scalable architectures.

(Zhu, Shyu, & Wang, 2013) in their paper propose a system called *VideoTopic*: A content−based video recommendation using LDA topic models. This system considers both the text and visual features of the videos for the LDA model. The text features are from the video transcripts. The visual features are the "visual words" taken from the raw video key frames. A screen grab from the video has multiple visual patterns (Fei-Fei & Perona, 2005; Sivic, Russell, Zisserman, Freeman, & Efros, 2008), akin to a document having multiple topics. The average topic distribution of the videos watched by a user is calculated and, assuming that the user would like to watch videos consistent with his or her interests, a suitable video with topic distribution closest to the user's interest will be recommended. The MovieLens 1M dataset and 3,475 movie trailers downloaded from youtube were used for their experiments.

(Wang & Xu, 2018) used LDA topic models to leverage deep learning for detecting automobile insurance fraud. The authors obtained a labelled dataset of 37,082 claims from an auto insurance company, of which 415 were fraudulent. The topic models were used to create additional useful features hidden in the text descriptions of the accidents given in the claims.The LDA added Deep neural network(DNN) did better than the DNN−only model (which in turn outdid the other classifiers like SVM and RF) on all metrics applied−Accuracy, Precision, TP rate, F1 score. The topical text features extracted by LDA also improved the understanding of auto insurance fraud. This is especially important for the claim processors who are not auto experts.

This gives a glimpse into the wide−ranging applications of topic modeling and makes its role in important text analysis tasks like information retrieval(IR), document clustering, document classification, natural language processing(NLP) etc. amply clear.

2.6 Evaluation Approaches

The LDA model is an unsupervised model, unlike the common classifiers like Logistic Regression, Naive Bayes, Random Forest, etc. Therefore, it requires its own criteria for evaluation. (Heinrich, 2005) discusses the evaluation of the cluster quality of a trained topic model. The trained LDA, by chugging out the document−topic distribution, is actually a 'soft clustering' of the documents. Here, distance metrics like the *symmetric* KL−divergence can be used for similarity ranking and a subjective judgement of the similarities can be carried out.

In case its an unseen document, the document-topic distributions can be derived by running the inference algorithm (Gibbs posterior sampling) on the query document, but with the trained model values for the topic−term distributions β_K and hyperparameter α (see page 11). The document−topic distribution obtained for the query document can then be used for calculating similarity as before.

As there is an element of subjectivity in judging the similarity scores, a more objective evaluation is the comparison of the estimated LDA model to an apriori

clustering/categorization of the corpus as a reference. Of course, this too is only as objective as the reference. Here, the Variation of Information distance(VI−distance) proposed by (Meila, 2003) can be used to calculate the distance between the 'soft clustering' of the LDA and the 'hard clustering' of the reference cluster.

Another common criterion of cluster quality not requiring apriori categorization is held-out log-likelihood (log converts the joint probability into a sum which is easier to deal with) i.e. $\log\ p(\vec{w}_{\tilde{m}}|M)$ where M refers to the trained model, $\vec{w}_{\tilde{m}}$ is the vector of words in the test document $\tilde{m}$. These log−likelihood values are usually large negative number, so another measure 'perplexity' is used. 'Perplexity' is the reciprocal geometric mean of the word likelihoods in the test corpus. It can be understood as the expected vocabulary size (under a uniform word distribution) that the model needs to generate any test token. In more layman terms, how 'confounded' is the model when it has to generate a test token? Lower values indicate better model quality.

The above evaluation measures are useful for evaluating LDA for predictive and information retrieval tasks, but they do not address the more exploratory goals of topic modeling like getting insights into the themes/concepts in the text collection through the LDA topic outputs. (Chang, Boyd-Graber, Gerrish, Wang, & Blei, 2009) present a method for measuring the interpretability of a topic model. They also demonstrate that models which achieve better predictive perplexity have less meaningful topics. The proposed human evaluation tasks are what is known as *word intrusion* and *topic intrusion*.

Word intrusion measures how semantically cohesive the topics inferred by a model are. This it does by measuring how often or how easily humans are able to spot 'intruders' − high-probability words from other topics randomly inserted into the topic of interest. *Topic intrusion* measures how well an LDA model's decomposition of a document as a mixture of topics agrees with human associations of topics with a document. Here humans are shown the document's title and a small snippet of it. They are then shown a set of LDA topics of which all except one, are strongly associated with the document as per the trained LDA. The 'intruder' is chosen randomly from the other low−probability topics in the model.

Another aspect of topic model interpretation is to be able to ascribe a significance rank to the model topics in terms of how well the associated words factually correspond to genuine themes of the corpus domain and are not just either 'noise' or representing unimportant themes. (AlSumait, Barbará, Gentle, & Domeniconi, 2009) offers 3 definitions of 'junk' topics using a variety of measures, by which it becomes possible to compute the topic significance using a heuristic decision making strategy called the "Weighted Linear combination" (WLC) approach (Bouyssou, Marchant, Pirlot, Tsoukias, & Vincke, 2006).

2.7 Limitations and gaps in the literature

As seen through the literature survey, there are very few studies using educational lecture videos as a data source for any sort of text analysis, what to speak of topic model analysis. Most of the work pertaining to educational data mining has been on text resources like discussion forums, online assignments, and essays with the objective of either analysing student performance or for providing backend student support (Ferreira-Mello et al., 2019). Yet, many studies have shown the importance of online educational videos in achieving learning outcomes and enhancing the learning experience.(Kay & Kletskin, 2012; Long, Logan, & Waugh, 2016; Guy, Retta and Marquis, 2016; Nagy, 2018).

On the other hand, there are many legitimate criticisms of Massive Online Open courses(MOOCs), one of which is the lack of adaptive learning. All learners have to go through the same course material (primarily lecture videos) in the same sequence and are also assessed uniformly. With a heterogeneous student cohort, this lack of adaption to individual learning needs can lead to frustration and exacerbate the already prevalent issue of drop-outs in MOOCs (Romero & Ventura, 2017). Tools offering automated personalised recommendations play a vital role in such a situation.

The machine learning algorithms behind these tools not only need to have higher accuracy but should also be interpretable as that would also enable the educators, who are subject matter experts, to contribute towards improving algorithmic performance.

'Black box' algorithms are not really suited for dealing with domain peculiarities. As seen in the survey, topic models are not only suited for all kinds of text mining tasks, but also retain the element of interpretability in their model outputs. Hence, topic models on lecture video transcripts have been considered for this thesis with the larger objective of empowering educators to participate in the creation of machine learning pipelines for recommender systems in personalised learning.

Chapter 3

Experiment design and methodology

In this chapter, a detailed description of the overall structure of the experiment has been provided along with a rationale for the specific design choices. Various aspects of the experiment preliminaries like corpus creation, data pre−processing, and feature engineering are in particular focus. The evaluation metrics and comparison tests are also explained. The aim of the study is to measure an improvement in topic model interpretability, if any, on a text corpus of lecture videos on machine learning, after introducing additional features in the form of bigrams. Interpretablity, in this study, implies the 'closeness' of the topic model's outputs and the original titles of the videos (human−labeled topics). The cosine similarity between the model's dominant topic for a particular video and the original title of the same video is taken as a proxy for interpretability. It is measured for both the baseline model and the bigrams−added experimental model and compared.

3.1 Design Summary

A step−wise description of the experiment, which summarises all the tasks, is given below. Every step will be elaborated upon at the appropriate place.

I. **Corpus creation:** The dataset, consisting of youtube video transcripts, is cre-

ated by scraping youtube website. The corpus for this experiment has 179 documents corresponding to 179 lecture videos on machine learning. The video's original title supplied by the video uploader/content creator is taken to be the 'labeled topic' for this experiment.

II. **Baseline model:**

- The corpus is pre−processed and made ready for topic modelling.
- The feature space consists of only unigram tokens.
- The LDA algorithm gives the corpus' topic−term distribution matrix for each of the preset number (K) of topics.
- After applying LDA, the following is done for each document 'd' in the corpus:
  (a) The document's dominant topic is obtained from the model output.
  (b) Only the top 'n' terms from the dominant topic are considered.
  (c) The Word2Vec word embedding algorithm, trained on the corpus, is used to get the vector representation of each of the top 'n' terms obtained in (b).
  (d) The trained Word2Vec is again fitted to obtain the vector representation of each of the 't' words in the video's original title (labeled topic).
  (e) Cosine similarity values for the $n * t$ pairs of vectors got in (c) and (d) are calculated
  (f) The average of the values got in (e) is the representative cosine similarity value for document 'd'. It is the cosine similarity between the document's LDA generated dominant topic and the original title (labeled topic) of the corresponding video.
- Thus, the baseline model now consists of a column/list of 179 values showing the "closeness/similarity" between the baseline−LDA generated dominant topic for each document and the original title (labeled topic) of the respective video.

III. **Experimental (Added features) model:**

- This has the unigram tokens as before, but the feature space includes bi−gram tokens.
- The same sequence of steps given for the baseline model is followed here too. The final output will be a column/list of 179 values representing the "closeness/similarity" between the experimental−LDA generated dominant topic for each document and the original title (labeled topic) of the respective video.

IV. **Statistical comparison:** The two columns of values obtained at the end of steps II. and III. will be compared using statistical tests to see if there is any significant difference in the means of the two columns. Depending upon the test results, the research hypothesis that use of bigrams increases LDA model topic's 'closeness' to human−labeled topics will accordingly be accepted/ rejected.

3.2 Design prerequisites

3.2.1 Corpus Creation

The text corpus for the experiment is crafted out of two youtube video playlists consisting of lectures on Machine Learning given by Prof.Andrew Ng of Stanford University, USA and Prof.Arti Ramesh of SUNY Binghamton, USA respectively. Prof.Andrew's machine learning playlist has 112 videos and Prof.Arti's playlist has 67, totalling to 179. Their video playlists are available for viewing at the following links:

- Prof.Andrew Machine Learning Playlist
- Prof.Arti Machine Learning Playlist

The thesis Appendix contains screenshots of the complete python script to automatically scrape the URLs and get the video transcripts and video titles into Python *pandas* dataframe format.

3.2.2 Data Pre-processing

The data pre-processing phase has two parts - data cleaning and data preparation.

3.2.2.1 Data cleaning

The two dataframes (see Appendix figures A.9 and A.10) have three columns each − *video_ID, transcripts*, and *video title*. The video ID columns serves as an index column, but is not a part of any of the experiment steps and is therefore kept untouched. For the video titles column, the following cleaning is done.

- Removing non−alphanumeric characters
- Removing repetitive phrases in the video titles. For e.g. every video title in the Prof.Andrew dataframe begins with the lecture number ('Lecture 18.1' etc.) and ends with the phrase [Machine Learning — Andrew Ng]. These phrases don't add any useful information to the video titles. Moreover, with the word embedding vector representations of each word in the video title being used for cosine similarity calculations, these phrases can also lead to wrong results.

For the transcripts column , the cleaning involves:

- Removing non−alphanumeric characters
- Expanding contractions like "don't" "isn't".

Python regular expression library *re* is used for pattern matching and substitution and *contraction* is used for expanding the contractions.

3.2.2.2 Data preparation

So far, the dataframes weren't merged into one as the cleaning requirement was unique to each dataframe. Now, the two frames are separately shuffled, merged, and re−shuffled so that the order of the documents has no undue influence on the LDA's model's topic learning process. The *random_state* argument in python pandas' *sample* method is called upon to ensure reproducibility.

The *data preparation* phase directly precedes the LDA model building stage which makes use of the Python *gensim* package. The LDA model uses the 'bag of words' (BOW) approach which only considers the word frequency and disregards sentence syntax, context, and location within text. A series of steps were undertaken to get the transcripts into a bag of words (BOW) feature representation ready for model building.

- The transcript string (a single string spanning an entire video transcript) was made into individual word tokens (smallest lexical units). Here, word token refers to a sequence of characters separated from the adjacent sequences by empty space.

- Stopwords (highly common english words) were removed from the text. If not removed, the stopwords, because of their ubiquity, will appear in the LDA model output topics without giving any useful information about the topic.

- The word token are lemmatized to convert inflectional forms of a word to its base, dictionary form. This helps reduce the dimensionality of the feature space.

- *Dictionary* module from Python *gensim* package is called to create a Dictionary (or vocabulary) of words. Yet another filter is applied, wherein very rare words (occurring in only one document) and common words (occurring in more than 40% of documents) are excluded from the Dictionary. [These are experience values used in the *Gensim* documentation examples]

- Finally, a bag of words vector space model of the transcripts is created. Here, every document (transcript) is represented by a row of frequency values for the corresponding Vocabulary token in the columns. The dataset now has 2863 unique tokens across 179 documents (179 * 2863 document−term matrix) ready for modeling.

The next section briefly lays the theoretical foundations of feature engineering and the evaluation metrics and provides the rationale for design choices made, before moving to baseline and experimental model implementation in the next chapter.

3.2.3 Feature Engineering

After data collection and data cleaning, the next step in any machine learning task is to transform the raw data into a form suitable for modeling. In text mining, each token is usually an input feature. The token, in this context can be a phrase, a word, or even a character. The feature value is typically a numerical measure indicating the importance (or weight) of that token. A vector of such values for a document is called the document's feature vector. The common forms of the feature vector are *Count Vectors*, *Term Frequency Inverse Document Frequency (TF−IDF) Vectors* and *Word Embeddings*.

3.2.3.1 Feature representation

Count Vectors: This is the simplest representation of text where the feature value is just the count of the token's occurrences within the document. Here, the corpus is represented as a document−term matrix where the documents of the corpus are the rows and the corpus vocabulary are the columns. This representation of text data is the mathematical equivalent of the Bag−of−Words (BOW) in which the document is treated as merely a collection of words and neglects order, context, and syntax. As the LDA model works on this BOW assumption, the count vectors representation of text has been employed for this experiment. The description of how to represent the current corpus in a BOW feature space has already been given under the *Data preparation* section of this chapter.

TF-IDF: This measure is created to overcome the inherent drawbacks of the count vectors feature representation. Words with higher frequency score higher in the count vectors case and will disproportionately skew the subsequent models. The TFIDF takes care of that by giving more importance only to those terms which have both high frequency within the document as well as a localized presence in the corpus. The TF component of this measure factors in the token's frequency within the document. The IDF component considers the token's relative importance in the corpus. Tokens which appear in fewer documents weigh more. Mathematically TFIDF is calculated

as shown below:

$$TF - IDF = f_{t,d} * \log \frac{N}{n_t} \tag{3.1}$$

$f_{t,d} \longrightarrow$ frequency of token t in document d

$N \longrightarrow$ number of documents in the corpus

$n_t \longrightarrow$ number of documents containing the token t

Word Embeddings: This is a dense representation of text where the tokens are represented as a vector of real numbers such that semantically similar tokens are located close to each other in the vector space. The underlying idea is a concept from linguistics known as the *distributional hypothesis* − that semantically similar tokens appear in similar contexts (words surrounding the given word) in text documents. This idea is exploited in the Word2Vec algorithm, one of the most popular Word Embedding models. The Word2Vec is a shallow neural network with hierarchical softmax activation. It uses either the Continuous bag-of-words (CBOW), or the Skip−Gram architecture to produce the distributed representation.

This experiment requires word embeddings at the evaluation and comparison stage when the baseline LDA model and the bigram features added experimental LDA model need to be compared (ref *Design Summary* section of this chapter). The reason for using a word2vec word embedding in the experiment is that it gives a semantically justifiable vector representation of words which can be used to measure distances using measures like cosine similarity. The measured distance values can then serve as a proxy for topic model interpretability. Higher mean cosine similarity indicates greater model interpretability.

The Google News 300 (word vectors of 300 dimensions trained on Google News dataset of 100 billion words) pre−trained Word2Vec model was attempted for this experiment but it led to some practical difficulties. Firstly, the model could not generate embeddings for words not in its vocabulary.For example, one of the lectures given by Prof.Arti was titled “Hoeffding’s inequality”. The pre-trained word2vec could not generate the embedding for the token ‘Hoeffding’.

Secondly, the pre−trained word2vec model, by design, depends on the vocabulary

and contexts in which the words appear in the pre−trained model's training data. In the context of some other studies, this resulted in perpetuating the training data's biases and cultural stereotypes related to gender and work (Llorens, 2018). To observe the impact on the current dataset, the pre−trained word2vec was tested on specific tokens which appear in the video titles.

In the normal course, an acronym like SVM (this acronym appears thrice in Prof. Arti playlist's video titles and once in Prof.Andrew playlist's video titles) would be readily understood as Support Vector Machine for someone familiar with Machine Learning. However, when the pre−trained Google News 300 model was used to find words most similar to SVM, it outputted many words and acronyms completely unheard of in Machine Learning. It became evident that the Google News pre−trained model, during its training phase, would have seen the acronym SVM in many other contexts unrelated to Machine Learning. Therefore, the acronym SVM's vector representation using the pre−trained model can no longer be assumed to carry discernible information about its semantic connection with other Machine Learning terminologies.

For the above reasons, the pre−trained word2vec model was dropped. Instead, a word2vec model was trained on the current corpus itself and used for generating embeddings (Relevant details are given in Chapter 4 under the section *Baseline Model*). This reinforces the view that human interpretability is very important at every stage of the machine learning pipeline and only someone familiar with the dataset's domain can run appropriate tests to check validity of outputs.

3.2.3.2 Feature selection

Another aspect of feature engineering is feature selection. The purpose is to select only the most informative features and leave out the trivial ones, as Very high dimensionality feature spaces increase model complexity, training time, and the costs associated with learning. (Dasgupta, Drineas, Harb, Josifovski, & Mahoney, 2007) discusses document frequency (DF), information gain(IG), chi-squared, mutual information and sampling as the standard methods to perform feature method.

For this experiment, the feature dimensionality is relatively small (2863 unique

tokens for baseline model), the features belong to a specific domain (all related to machine learning) and will be used for unsupervised learning (LDA topic model) rather than a supervised classification task. So feature selection is not of particular concern, especially for the baseline LDA model.

But for the bigrams added LDA, it does merit some attention (this is also related to one of the research sub−questions on choosing optimum bigrams). Bigrams need to be filtered suitably, else they could trigger an unwanted cycle − create many word-combinations not 'sufficiently observed' in the data, leading to more training data being required, which in turn makes more unseen bigram word combinations. This results in an 'explosion' of features.

Normalized pointwise mutual information (NPMI) based bigram filtering is very appropriate as it can extract those bigrams which are typical of the dataset's domain thus aiding interpretability, and yet be less sensitive to low occurrence frequency of bigram (Bouma, 2009). Mathematically,

$$NPMI = \frac{\ln \frac{p(x,y)}{p(x)p(y)}}{-\ln p(x,y)} \tag{3.2}$$

When the two words 'x' and 'y' only occur together, then $p(x, y) = p(x) = p(y)$ and NPMI equals 1. When the tokens are distributed as expected under 'independence', then NPMI equals 0 ; when they never occur together then NPMI equals -1. In the python *gensim* implementation of this experiment, NPMI was set to 0.5 so that the candidate bigrams have high degree of correlation and yet, are not fully dependent on each other . This filtered out many irrelevant bigrams which turned out to be common phrases used by the narrators/instructors in their videos, for example 'this video', 'next step', 'hello everyone' and similar such phrases. The bigram added experimental LDA model had 3430 unique tokens against the baseline LDA's 2863.

3.3 Comparison methods

For this experiment, there are two types of model comparison. One is intra−model and the other is inter−model.The former is to do with choosing the optimum model

settings for the baseline LDA and the experimental LDA model, and the latter is about comparing the optimum baseline and experimental LDA model.

3.3.1 Topic coherence

One of the research sub−question was determining the optimal number of topics for the model. Topic coherence is a metric recommended for this evaluation. Topic coherence measures the intrinsic quality of the topics produced by the model rather than judging the model based on extrinsic performance like model perplexity on a held-out test document. Topic coherence is a measure of how cohesive the high-probability words of a topic are. It was originally proposed by (Newman, Lau, Grieser, & Baldwin, 2010) and variations were developed such as the one by (Aletras & Stevenson, 2013).

3.3.2 Statistical tests

Another research sub−question had to do with choosing the appropriate statistical test to compare the baseline LDA and experimental LDA model. Since the test has to do with comparing the cosine similarity values obtained for the baseline and experimental model, the t−test is most suited for the purposes.

The t−test is of two kinds : the independent samples t−test and the paired sample t−test. The paired sample is used only when the measurements are on the same person or thing. For example, two blood pressure measurements on the same person using different equipment. It is possible to justify the use of either of these tests. The possible argument in favour of either is given below.

Argument for Independent samples t-test: Although the original corpus is the same, the feature set for the baseline and the experimental model are different. Moreover, the frequency values for the feature tokens would change and the trained word2vec embeddings of the LDA topic terms would also be different.

A hypothetical example will make this clear. Suppose the baseline model has terms like ‘machine’, ‘rate’, ‘supervised’, ‘learning’, and ’unsupervised’ as separate features with their own frequency counts and word2vec representations. The bigrams added fea-

tures model could probably have 'machine learning','learning rate', 'supervised learning','unsupervised learning' as bigram features in addition to retaining 'learning' and 'rate' as unigram features.

The example above shows how the feature set itself has changed. Naturally, the token frequency values (which impacts the BOW based LDA model output) and the word embeddings (which depend on the context words in the corpus) of the LDA topic terms would also change. Therefore, for all practical purposes, the cosine similarity values of the baseline and experimental model are from 'independent samples'.

Argument for Paired samples t-test: Here both the baseline and experimental model are subjected to the same learning algorithm, the LDA. Hence, their outputs and all further derivations from that should be treated as paired samples. So the paired samples t-test and its non-parametric equivalent, the wilcoxon signed-rank test needs to be done.

In this experiment, both kinds of t-tests will be used for comparing the two models, to cover for the arguments that could be in favour of either of them. To sum up, this chapter has given the experiment structure, provided a detailed description of all the necessary steps before implementation like data collection, data pre−processing and feature engineering, addressed the research sub−questions raised in Chapter 1, and discussed the methodologies adopted for design and evaluation.

Chapter 4

Results, evaluation and discussion

This chapter details the experiment's implementation and the results obtained thereof. This will be followed by a discussion on the findings in the light of the literature reviewed in Chapter 2. The strengths and limitations of the study will also be analysed.

4.1 Implementation

The implementation is in three stages. It begins with the *Baseline Model*, then the *Experimental Hypothesis model* and finally, the *Model Evaluation* stage where the two models are compared.

4.1.1 Baseline Model

The baseline model refers to the LDA topic model algorithm applied on the corpus having only unigram features. An important factor affecting topic model output is the number of desired topics K. This has to be decided in advance by the researcher as it is a necessary input parameter to run the model. If the value of K is too small, the model topics will be overly broad, while very large values of K will lead to many redundant, narrow topics. As discussed in Chapter 3, topic coherence is generally used to choose the appropriate K. The same measure will be applied here too. The model will be repeatedly run iterating over a range of values for K, and topic coherence will be used to select the optimum baseline model.

For running the baseline model the Python *gensim* library is used (Řehůřek & Sojka, 2010). Although this library contains a module *LdaModel* to directly create topic models, it is recommended instead to use the MALLET framework (available for download here) and the Python *gensim* wrapper *LdaMallet*. MALLET, which stands for MAchine Learning for LanguagE Toolkit, is a Java−based package for all kinds of text mining tasks (McCallum, 2002). The MALLET topic modeling toolkit contains a fast, memory−efficient, Gibbs sampling implementation of LDA.

The baseline model was run repeatedly, looping over K ranging from 2 to 40 with a step size of 1. The input data was a 179 by 2863 document−term matrix (see page 28 for steps to prepare input). On a personal laptop powered by a Intel(R)Core(TM)i3 CPU@2.00GHz and 8 GB RAM, the model took 42.74 seconds/iteration i.e. 27 minutes, 46 seconds in all to train.

Figure 4.1 tabulates the top 10 choices for K (number of topics) by decreasing order of Topic coherence value. As per this table, coherence score is highest for $K = 21$.

	Number of Topics	Coherence Score
19	21	0.5342
17	19	0.5303
21	23	0.5260
23	25	0.5163
22	24	0.5119
26	28	0.5111
28	30	0.5053
32	34	0.5050
20	22	0.5040
14	16	0.5012

Figure 4.1: Top 10 K values by descending order of coherence

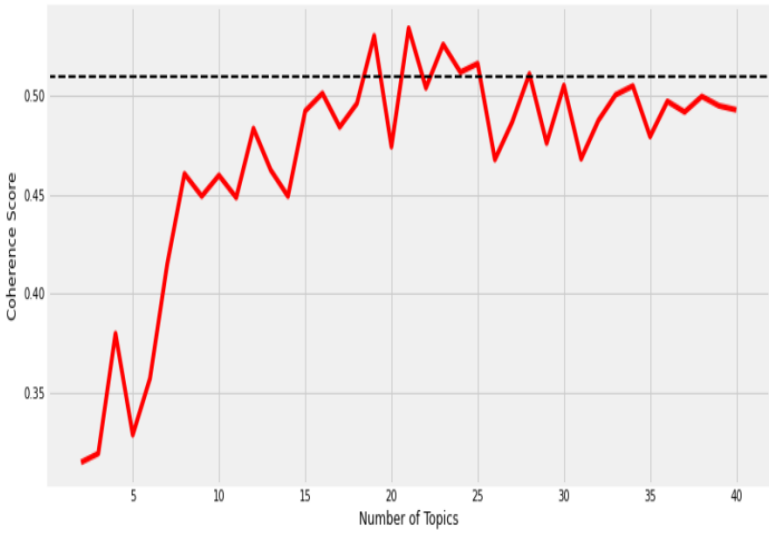

Figure 4.2: Plot of topic coherence vs number of topics

Figure 4.2 plots coherence scores against the number of topic (K). Choosing the optimum number of topics has an element of subjectivity to it, but a recommended rule of thumb is to look for K which arrests the trend before it. With this in mind, $K = 22$ is chosen, as the coherence value are consistently increasing with K before

that. The following table shows the output of the baseline topic model for the optimum number of topics K=22. For each topic, the top 20 terms associated with the topic in decreasing order of importance is given. It must be mentioned that the 22 topics are internally represented with the topic IDs 0-21, although the table below shows the topic numbers as 1-22.

	Terms per Topic
Topic1	class, word, classification, probability, calculate, spam, document, count, positive, classifier, predict, bayes, prediction, calculation, final, article, naive, assumption, thousand, occurring
Topic2	kernel, large, negative, support, squared, boundary, transpose, greater, decision, linear, positive, hypothesis, svm, classifier, predict, 0.5, close, landmark, polynomial, call
Topic3	cluster, k-means, centroid, step, blue, red, randomly, random, move, clustering, t-shirt, running, assigned, run, purpose, assignment, initialize, distance, closer, shown
Topic4	octave, type, command, show, return, square, variable, put, file, comma, code, save, string, library, quickly, desktop, dot, magic, loop, hope
Topic5	email, type, supervised, people, learn, reason, word, unsupervised, spam, human, brain, process, sense, system, hard, quick, tool, task, group, label
Topic6	rule, true, space, multiple, probability, hypothesis, suppose, event, variable, combination, fit, counter, sample, key, computer, raise, world, real, outcome, learn
Topic7	anomaly, detection, model, probability, engine, large, test, aircraft, positive, epsilon, gaussian, label, anomalous, negative, normal, center, evaluate, capture, put, plot
Topic8	gradient, descent, derivative, update, step, partial, cost, rate, minimum, implementation, alpha, iteration, batch, implement, stochastic, epsilon, global, single, smaller, respect
Topic9	error, respect, sigmoid, net, differentiation, sigma, recall, precision, cancer, weight, output, patient, true, threshold, update, summation, predict, delta, setting, quantity
Topic10	cost, hypothesis, logistic, optimization, sum, minimize, linear, regularization, objective, squared, notation, corresponds, lambda, equation, written, earlier, denote, usual, previous, definition
Topic11	matrix, element, column, row, transpose, inverse, multiply, linear, dimensional, multiplication, algebra, identity, result, dimension, operation, add, product, real, taking, normal
Topic12	thousand, performance, hundred, 000, 100, run, test, choose, people, method, common, evaluation, result, option, accuracy, correct, ten, pick, long, earlier
Topic13	tree, instance, decision, attribute, humidity, high, outlook, sunny, node, split, play, path, entropy, normal, tennis, side, branch, leaf, model, answer
Topic14	sigma, gaussian, pca, dimensional, distribution, squared, formula, reduce, multivariate, project, original, projection, surface, variance, square, apply, dimension, standard, mathematical, product
Topic15	image, text, system, character, pixel, label, pipeline, patch, photo, pedestrian, car, individual, recognize, person, classifier, detection, idea, country, ocr, run
Topic16	size, plot, house, price, figure, predict, axis, range, tumor, fit, linear, straight, housing, malignant, hypothesis, fact, color, horizontal, benign, output
Topic17	alpha, solve, constraint, margin, optimization, equation, transpose, square, respect, dual, lagrange, sigma, solving, slack, optimize, part, version, differentiate, coefficient, remember
Topic18	error, high, fit, test, bias, model, variance, curve, hypothesis, polynomial, overfitting, cross-validation, low, validation, fitting, regularization, figure, higher, cross, close
Topic19	beta, prior, distribution, log, likelihood, maximum, estimate, estimation, curve, alpha, head, posterior, quantity, tail, raised, probability, coming, greater, form, 0.5
Topic20	network, unit, neural, layer, output, input, hidden, delta, node, propagation, neuron, activation, computing, complex, weight, computed, sum, forward, computation, concretely
Topic21	model, weight, form, prediction, space, logistic, functional, circle, greater, separable, learn, regularization, complex, essentially, linearly, linear, amount, dimension, classification, learned
Topic22	movie, user, rating, learn, website, phone, system, action, rated, recommender, predict, online, prediction, star, alice, rate, collaborative, love, filtering, eve

Table 4.1: Baseline LDA Topics

The next task is to get the dominant topic for each document. For this, the document-topic distribution for each document is first obtained, and the topicID is chosen which has maximum allocation proportion for that document. As an illus-

tration, the figure below shows the topicIDs for a randomly selected video sorted by decreasing order of topic proportions.

```
[(7, 0.47619614613720707),
 (11, 0.1465737929392153),
 (9, 0.07519199857117342),
 (15, 0.057946855589291746),
 (2, 0.040265126709134576),
 (17, 0.035026095929828 75),
 (4, 0.025421206167768 06),
 (16, 0.021491933083288686),
 (6, 0.01887241769363577),
 (21, 0.012105336270365741),
 (5, 0.010140699728126056),
 (14, 0.010140699728126056),
 (18, 0.009485820880712826),
 (12, 0.008830942033299599),
 (3, 0.007521184338473141),
 (8, 0.007521184338473141),
 (10, 0.007084598440197655),
 (19, 0.006648012541922 17),
 (13, 0.006429719592784427),
 (1, 0.005993133694508941),
 (20, 0.005993133694508941),
 (0, 0.005119961897957969)]
```

Figure 4.3: Topic allocations of randomly selected document

TopicID 7 (corresponding to Topic8 in Table 4.1) is the dominant- topic for this document. The dominant topics for each document are similarly obtained along with the corresponding top 20 terms. They are added to the dataframe containing the transcripts and video titles of all 179 documents. The dataframe is shown below in Figure 4.4. The last column *original title terms* contains the tokenized form of the column *Video Titles*.

	video_id	transcript	Video title	dominant topic number	topic terms	original title terms
0	ISBGFY-gBug	in this video I would like to tell y...	Advice For Applying Machine Learnin...	17	[error, high, fit, test, bias, model...	[advice, for, applying, machine, lea...
1	8DfXJUDjx64	in the last video we developed an an...	Anomaly Detection Developing And ...	6	[anomaly, detection, model, probabil...	[anomaly, detection, developing, and...
2	-Ia3q9d7AKQ	in this and the next few videos I wa...	Logistic Regression Classification	15	[size, plot, house, price, figure, p...	[logistic, regression, classification]
3	5aHWpIWEIcc	in the PCA algorithm we take n dimen...	Dimensionality Reduction Choosing...	13	[sigma, gaussian, pca, dimensional, ...	[dimensionality, reduction, choosing...
4	AasatTRGVhI	hello everyone welcome back in this ...	K Nearest Neighbors	20	[model, weight, form, prediction, sp...	[k, nearest, neighbors]
...	...	...	...	...	...	...
174	Ccje1EzrXBU	sometimes people talk about support ...	Support Vector Machines Large Mar...	1	[kernel, large, negative, support, s...	[support, vector, machines, large, m...
175	kHwlB_j7Hkc	our first learning algorithm will be...	Linear Regression With One Variable...	15	[size, plot, house, price, figure, p...	[linear, regression, with, one, vari...
176	I4ISUAcvHFs	in the previous video we talked abou...	Large Scale Machine Learning Mini...	7	[gradient, descent, derivative, upda...	[large, scale, machine, learning, mi...
177	0i9OhkbfNwE	in this video I would like to keep w...	Neural Networks Representation Ex...	19	[network, unit, neural, layer, outpu...	[neural, networks, representation, e...
178	yN-xTgxxNr4	in the last video we introduced Baye...	Point Estimation Bayesian Learning f...	18	[beta, prior, distribution, log, lik...	[point, estimation, bayesian, learni...

179 rows × 6 columns

Figure 4.4: Input dataframe with dominant topicID and topic terms added

Next, the word2vec representation (word embeddings) for the dominant topic's terms and the original title terms are required to make cosine similarity calculations. *word2vec* module from *gensim* library is utilized to create the word2vec model. The word2vec was trained on the current corpus with the context window size set to 20 words, the number of training iterations set to 50, and a desired vector dimension of ten. The trained model is saved and loaded to ensure reproducibility if the experiment is repeated in future. The word embeddings for all the necessary tokens are obtained and stored as Python *Numpy* arrays. Following that, python's *Scikit-Learn* is imported for calculating the representative cosine similarity value for each document (Chapter 3, section *Design Summary* can be looked up for the design steps of baseline model).

4.1.2 Experimental hypothesis model

The experimental model begins with the creation of bigram features which will be added to the input data. Python's *gensim* provides a *Phrases* module which generate unigram as well as bigram phrases of the text transcripts. As discussed in Chapter 3, NPMI is a very appropriate measure for bigram filtering, as the threshold value of NPMI can be conveniently adjusted to extract those bigrams which are more typical of the domain, thus aiding in the interpretability of model output. NPMI ranges from -1 to +1. -1 indicates that the bigram tokens never occur together, and +1 implies that the tokens always occur together. NPMI equalling zero implies that tokens are independent i.e. they occur together with a frequency expected under the independence assumption.

By choosing various values of NPMI ranging between $0.2 - 0.8$ with a step of 0.1, it was possible to manually inspect the generated phrases for randomly selected documents, and choose the NPMI which removed irrelevant bigrams the most. Admittedly, manual inspection is a luxury afforded only because of a relatively smaller corpus, but it allows for subject experts to contribute to the model building process by generating experience values of these parameters on smaller subsets of larger corpuses, and apply those values for the entire corpus. NPMI=0.5 gave best results and was hence chosen for bigram generation. Another unexpected benefit of this manual inspection was that

it revealed other common tokens which could be added to the stop word list for stop word removal.

After generating bigrams, the input data now had 3430 unique tokens as against the baseline's 2863 unique tokens. This point onwards, the same sequence will be followed as done for the baseline LDA model. The LDA model was run for this bigram-added corpus using MALLET, with K value ranging from 2-40 and 500 iterations over the input data for each K. The model took 47.14 seconds/iteration over K totalling 30 minutes and 38 seconds. The following figures show the top 10 values of K when sorted by topic coherence, and the plot of topic coherence over the range of K values.

	Number of Topics	Coherence Score
23	25	0.5606
21	23	0.5434
26	28	0.5421
25	27	0.5410
27	29	0.5393
32	34	0.5347
35	37	0.5288
18	20	0.5275
36	38	0.5247
30	32	0.5241

Figure 4.5: Top 10 K values by descending order of coherence

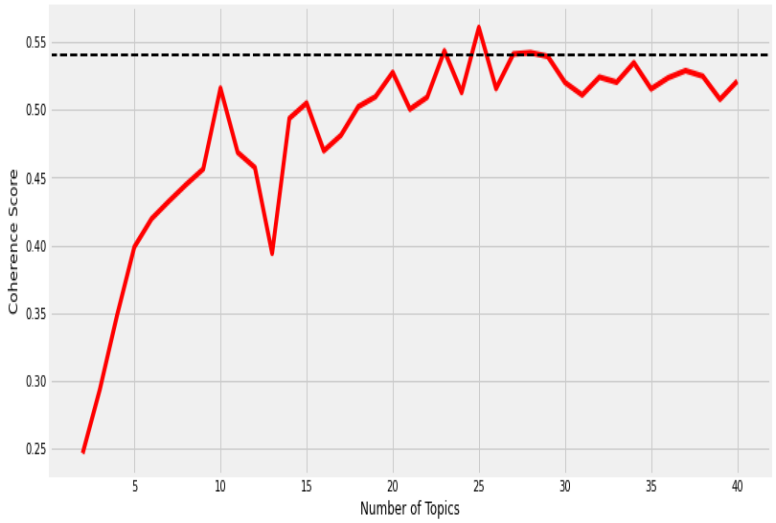

Figure 4.6: Plot of topic coherence vs number of topics

A cursory glance through figures 4.5 and 4.1 shows that the experimental hypothesis model has slightly higher topic coherence on average. Once again, there is a bit of ambiguity on the best choice of K. K=23 seems to be the starting point where the plot starts to plateau. Then onward, the plot seems to have a flat trend, save for the spike at K=25. Hence, K=23 was chosen as the optimum number of topics.

The model output for the experimental hypothesis model is shown below as a table of topics with their 20 most likely terms. As can be seen, some of the topics have bigram tokens. Another remarkable feature is that almost all bigram tokens in the model output are immediately recognizable word combinations in the context of machine

learning. Very few exception exist, like "when_differentiate" and "nothing_but" in Topic20. This vindicates the NPMI based approach to bigram extraction discussed in the beginning of this section.

	Terms per Topic
Topic1	k-means, clusters, blue, unsupervised_learning, people, step, cluster_centroid, randomly, clustering, points, red, cluster, cluster_centroids, group, random, assigned, method, move, number_clusters, turns
Topic2	training_set, size, machine_learning, problems, learning_algorithm, learning_algorithms, videos, house, price, computer, feature, single, multiple, neural_networks, brain, algorithms, 100, idea, main, additional
Topic3	true, space, test, rule, performance, learn, learning, based, rules, model, suppose, lots, counter, key, real, important, models, big, bunch, wrong
Topic4	gradient_descent, update, descent, step, alpha, learning_rate, cost_function, derivative, gradient, partial_derivative, steps, implement, epsilon, respect, algorithms, loop, initial, stochastic_gradient, advanced_optimization, code
Topic5	neural_network, output, delta, network, layer, input, propagation, unit, units, outputs, inputs, node, neural_networks, terms, hidden_layer, computing, hidden_units, activation, nodes, computed
Topic6	octave, type, command, algorithms, quickly, show, comma, put, code, file, return, desktop, variable, save, learning_algorithms, define, returns, commands, generate, colon
Topic7	matrix, vector, matrices, multiply, elements, column, element, turns, row, inverse, vectors, transpose, dimension, linear_algebra, result, written, taking, columns, identity_matrix, multiplication
Topic8	run, implement, thousand, 000, hundred, software, 100, transpose, method, training_examples, talked, ten, long, algorithms, concretely, idea, details, feature, worry, reason
Topic9	instances, tree, humidity, attribute, split, class, decision, entropy, high, calculate, play_tennis, probability, decision_tree, outlook, normal, path, sunny, attributes, outlook_sunny, branch
Topic10	sigma, model, anomaly_detection, feature, probability, gaussian, distribution, formula, multivariate_gaussian, gaussian_distribution, normal, square, anomalous, epsilon, high, parameter, sigma_squared, center, anomaly, aircraft_engines
Topic11	cost_function, hypothesis, term, linear_regression, logistic_regression, minimize, training_set, terms, cost, parameter, notation, turns, squared, corresponds, definition, apply, define, equation, fit, comma
Topic12	model, points, line, linear, feature, curve, prediction, complex, red, linearly_separable, simple, weights, circle, functional_form, bias, axis, models, blue, test, classification
Topic13	predict, 0.5, cancer, predicting, classifier, tumor, positive, threshold, learning_algorithm, actual, high, higher, precision, precision_recall, malignant, recall, predicted, call, classification, decide
Topic14	user, movies, movie, users, learn, rating, website, feature, parameter_vector, action, ratings, learning, phone, predict, alice, predicted, collaborative_filtering, rate, love, phones
Topic15	support_vector, machine, transpose, squared, vector, kernel, decision_boundary, svm, machines, norm, length, kernels, similarity, choose, positive, gaussian_kernel, close, inner_product, term, functions
Topic16	fit, error, regularization, training_set, hypothesis, test_set, model, overfitting, test, fitting, learning_algorithm, high, sets, lambda, train, cross-validation_set, cross_validation, order_polynomial, training_examples, cross-validation_error
Topic17	machine_learning, image, system, supervised_learning, people, text, applications, working, images, pipeline, build, application, label, performance, videos, labeled, run, learning_algorithm, learning, 000
Topic18	prior, probability, beta, log, estimate, estimation, parameter, alpha, distribution, heads, posterior, likelihood, curve, maximum_likelihood, raised, tails, coin, event, difference, beta_distribution
Topic19	class, words, spam, word, email, feature, calculate, probability, emails, positive, predict, classification, classes, document, count, prediction, types, naive_bayes, assumption, occurring
Topic20	term, sigma, respect, nothing_but, error, sigmoid, net, weight, weights, differentiation, step, form, summation, remember, quantity, when_differentiate, terms, perceptrons, coefficient, guy
Topic21	pca, dimensional, vectors, line, reduce, choose, project, original, representation, surface, numbers, feature, represent, vector, dimension, space, plane, call, dimensions, input
Topic22	alpha, margin, solve, points, line, square, constraints, transpose, constraint, dual, equation, solving, optimize, svm, optimization_problem, primal, slack, version, distance, lagrange
Topic23	plot, figure, average, smaller, corresponds, larger, fact, range_values, slightly, size, curve, color, show, change, contrast, part, horizontal_axis, clear, kind, purpose

Table 4.2: Experimental Hypothesis LDA Topics

Just as the baseline model, here too, the document's dominant topic is determined. The *pandas* dataframe for this experimental model, like the one for baseline model in page 38, is shown below.

	video_id	transcript	Video title	dominant topic number	topic terms	original title terms
0	ISBGFY-gBug	in this video I would like to tell y...	Advice For Applying Machine Learnin...	15	[fit, error, regularization, trainin...	[advice, for, applying, machine, lea...
1	8DfXJUDjx64	in the last video we developed an an...	Anomaly Detection Developing And ...	9	[sigma, model, anomaly_detection, fe...	[anomaly, detection, developing, and...
2	-la3q9d7AKQ	in this and the next few videos I wa...	Logistic Regression Classification	12	[predict, 0.5, cancer, predicting, c...	[logistic, regression, classification]
3	5aHWpIWEIcc	in the PCA algorithm we take n dimen...	Dimensionality Reduction Choosing...	20	[pca, dimensional, vectors, line, re...	[dimensionality, reduction, choosing...
4	AasatTRGVhI	hello everyone welcome back in this ...	K Nearest Neighbors	11	[model, points, line, linear, featur...	[k, nearest, neighbors]
...	...	...	...	...	...	...
174	Ccje1EzrXBU	sometimes people talk about support ...	Support Vector Machines Large Mar...	14	[support_vector, machine, transpose,...	[support, vector, machines, large, m...
175	kHwlB_j7Hkc	our first learning algorithm will be...	Linear Regression With One Variable...	1	[training_set, size, machine_learnin...	[linear, regression, with, one, vari...
176	I4ISUAcvHFs	in the previous video we talked abou...	Large Scale Machine Learning Mini...	3	[gradient_descent, update, descent, ...	[large, scale, machine, learning, mi...
177	0i9OhkbfNwE	in this video I would like to keep w...	Neural Networks Representation Ex...	4	[neural_network, output, delta, netw...	[neural, networks, representation, e...
178	yN-xTgoxNr4	in the last video we introduced Baye...	Point Estimation Bayesian Learning f...	17	[prior, probability, beta, log, esti...	[point, estimation, bayesian, learni...

179 rows × 6 columns

Figure 4.7: Experimental model dataframe with columns for dominant topicID and its topic terms included

Furthermore, a word2vec was trained on the bigrams added corpus and, for each document, word embeddings were generated for the dominant topic's terms and the original video title's terms. Finally, the representative cosine similarity value for each document under the experimental hypothesis model was calculated and saved. The next section outlines the statistical test results of the comparison between the cosine similarity values of the two models.

4.1.3 Model evaluation

Having obtained the representative cosine similarity value for each document, both for the baseline as well as the experimental model, they need to be statistically compared to evaluate which model performed better in terms of 'interpretability'. 'Interpretability' in the context of this study, is the closeness between the model's dominant topic for any given document and the corresponding video title. This is measured by the representative cosine similarity value for each document. If it is seen that the mean of these values for the experimental model is higher than that for the baseline model,

then that becomes the empirical evidence for the hypothesis that the bigrams added LDA model produces topics closer to human-labeled topics, when compared to the vanilla LDA model. The figure below shows the *pandas* dataframe consisting of the representative cosine similarity values for each document, for the baseline as well as bigrams-added experimental model.

```
     bigramsLDA_model_similarity_values  baseline_model_similarity_values
0                              0.244054                          0.086127
1                              0.195746                          0.268857
2                              0.311845                          0.026704
3                              0.194403                          0.202126
4                              0.346184                          0.202575
..                                  ...                               ...
174                            0.269977                          0.169448
175                            0.176268                          0.066915
176                            0.317638                          0.313013
177                            0.134748                          0.266578
178                            0.257296                          0.295815

[179 rows x 2 columns]
```

Figure 4.8: Cosine similarity values for the baseline and the experimental LDA models

There are two kinds of statistical t-tests, independent samples and paired samples. As previously mentioned in Chapter 3, both the test will be done for this study.The figure below shows the output of the independent samples t-test.

	Independent t-test	results
0	Difference (bigramsLDA_model_similarity_values...	0.0508
1	Degrees of freedom =	356.0000
2	t =	3.8217
3	Two side test p value =	0.0002
4	Difference < 0 p value =	0.9999
5	Difference > 0 p value =	0.0001
6	Cohen's d =	0.4040
7	Hedge's g =	0.4031
8	Glass's delta =	0.4265
9	r =	0.1985

Figure 4.9: Results of Independent Samples t-test

As can be seen in figure(4.9), the difference in means between the columns of cosine similarity values is statistically significant (N=356,t=3.8217, $p < 0.001$). Cohen's D at 0.4040 indicate medium effect size.

The figures(4.10) and (4.11) below shows the results for the paired-sample t-test and its non-parametric equivalent Wilcoxon signed-rank test.

	Paired samples t-test	results
0	Difference (bigramsLDA_model_similarity_values...	0.0508
1	Degrees of freedom =	178.0000
2	t =	5.5381
3	Two side test p value =	0.0000
4	Difference < 0 p value =	1.0000
5	Difference > 0 p value =	0.0000
6	Cohen's d =	0.4139
7	Hedge's g =	0.4131
8	Glass's delta =	0.4265
9	r =	0.3834

Figure 4.10: Paired sample t-test output

	Wilcoxon signed-rank test	results
0	Mean for bigramsLDA_model_similarity_values =	0.276977
1	Mean for baseline_model_similarity_values =	0.226177
2	T value =	4568.000000
3	Z value =	-5.022800
4	Two sided p value =	0.000000
5	r =	-0.265500

Figure 4.11: Wilcoxon signed rank test results

The results for the paired test and its non-parametric equivalent show a statistically significant difference between the cosine similarity values of the two models. Both have the two-sided p-value equalling zero. This is a further confirmation of the hypothesis for this study.

4.2 Results summary and discussion

This section contains a summary of the results got and it is followed by a discussion on the strengths and limitations of the findings.

4.2.1 Summary of results

The two tables below are a snapshot of the implementation capturing the important numbers for the two models.

	Baseline Model	**Experimental Hypothesis Model**
Token characteristic	Unigrams only	Unigrams and Bigrams
Unique token count	2863	3430
Training time	27 min, 46 sec	30 min, 38 sec
Maximum topic coherence	0.5342 @ K=21	0.5606 @ K=25
Optimum number of topics(K)	22	23

Table 4.3: Summary of model characteristics

	Independent t-test	**Paired t-test**	**Wilcoxon test**
Hypothesis mean	0.277	0.277	0.277
Baseline mean	0.226	0.226	0.226
Hypothesis 95% Conf. Interval	0.259 - 0.294	0.259 - 0.294	N.A.
Baseline 95% Conf. Interval	0.206 - 0.245	0.206 - 0.245	N.A.
Test statistic	t=3.8217	t=5.5381	Z=-5.022
2-sided p-value	0.0002	0.0000	0.00000
Cohen's d	0.404	0.4139	N.A.

Table 4.4: Results of statistical tests on cosine similarity values for the two models

4.2.2 Strength of findings

The finding is significant as the experiment design is conceptually sound and founded on valid methods employed in text analysis. The design is also robust as it can be applied for all corpus sizes, big or small.

The results clearly demonstrate that bigrams do impact topic models and hence, it also provides a starting point for further research in this area. Another highlight is

that all intermediate stages and outputs in the experiment have an everyday, intuitive meaning which makes the entire analysis accessible to text mining practitioners from different backgrounds.

The results of this study are in also line with findings seen elsewhere. (Lau et al., 2013) and (Nokel & Loukachevitch, 2015) showed that introducing bigrams in topic modeling improved topic coherence scores. This was also seen in the current study.

4.2.3 Limitations of findings

The study was restricted to one specific corpus, transcripts of lecture video on machine learning. The same experiment needs to be conducted on datasets from many other fields in order to draw more definite conclusions. The size of the dataset is another limiting factor. At 179 documents, the corpus could be labeled as 'small' and hence, some would say, the findings are not generalizable to all corpus sizes. While there were some real constraints for this study like paucity of time, it does pave the way for future in-depth studies on a variety of datasets, both in terms of size and the domain.

Chapter 5

Conclusion

This chapter is a brief account of the study with an emphasis on the learnings, the novelty aspects, and recommendations for future lines of research. There is a concise overview of the thesis in the sections *Research Overview*, *Problem Definition*, and *Design, Evaluation, and Results*. The sections *Contributions and impact* and *Future work and recommendations* highlight the uniqueness and innovations in this study, and avenues for further research.

5.1 Research Overview

This research sought to provide empirical evidence for improvement in topic modeling performance, when applied on a text corpus consisting of transcripts of lecture videos on machine learning, by introducing bigram features in the input dataset. The hypothesis was grounded on the background knowledge about topic modeling gleaned from the various research studies which used the LDA algorithm to analyze texts.

The baseline LDA model had only the unigram tokens as input features while the experimental hypothesis model included bigram tokens. In both cases, applying the LDA model outputs document-topic and topic-term distributions. From this, the dominant topic associated with each document is determined for the respective model.They need to be separately compared with the actual video titles to see which of the two models gives topics closer to human-labeled topics. The word2vec model was

applied to generate word embeddings for the pertinent terms in the dominant topic and video titles. Cosine similarity distance measure was used to find the 'closeness' between the dominant topic and the original title for both models. Statistical tests were then carried out to see which model had higher mean cosine similarity values, thus indicating closeness of that model's output topic with human-labeled titles.

5.2 Problem Definition

The research problem was framed in the form of a question: *"Can the LDA topic modeling algorithm, applied on a text corpus of transcripts of youtube lecture videos on Machine Learning having added features in the form of bigram word tuples, generate topics of higher mean cosine similarity with the actual video title, compared to applying the algorithm on the same corpus but without the added features?"*
The research question necessitated looking into 4 sub-questions:
What is the optimum number of topics in the given dataset?
How to represent the topic words to be able to use similarity measures?
How to arrive at the optimum number of bigrams?
What statistical tests can be applied to measure differences in mean values?

The research problem was formally restated as the following null and alternative hypotheses:

Null Hypothesis: For a text corpus of transcripts of youtube lecture videos on machine learning, using additional features in the form of bigram word tuples does not change the mean cosine similarity between the LDA model's generated topics and the actual video titles, compared to that of the baseline LDA model without the additonal features.

Alternative Hypothesis: For a text corpus of transcripts of youtube lecture videos on machine learning, using additional features in the form of bigram tuples increases the mean cosine similarity between the LDA model topics and the actual video titles, compared to that of the baseline LDA model without additional features.

5.3 Design, Evaluation and Results

Barring inclusion of bigram features, the design steps for the baseline and the experimental hypothesis steps are almost identical. Hence, the following summary of the steps for the baseline model can be easily extended to the experimental model.

The dataset for this study was created by scraping youtube playlists of lectures on machine learning. Each document in the text corpus is a transcript of a lecture video. After corpus creation, it was cleaned and prepared for topic modeling using the LDA algorithm. The output of the LDA algorithm gives the multinomial document-topic and topic-term distributions. The dominant topic for each document was derived from the output. A word2vec model trained on the text corpus was used to create 10-dimensional word embeddings of the terms in the original title and the dominant topic for each document.

Assuming t terms in the document title and n terms in the document's dominant topic, a $n * t$ matrix of cosine similarity values is generated for each document by the considering the word vectors pairwise. Then the mean of the matrix values is taken as the representative cosine similarity value for each document. It represents the 'closeness' between the document's LDA generated dominant topic and the document's original title provided by the video uploader. For each of the 179 documents in the corpus, this value is calculated, thus giving a column of cosine similarity values for the baseline model.

The above sequence of steps is also followed for the experimental model to generate a similar column of values for this model.These 2 columns of cosine similarity values were compared using statistical tests(parametric and non-parametric) to check for any significant difference in means. Successful implementation of the design would address the research question talked about in the previous section *Problem Definition*. As an integral part of the study, it was also required to make deliberate design choices pertaining to the four research sub-question raised previously. They are briefly mentioned below.

- To choose the optimum number of topics, topic coherence was used as a metric.

- Similarity measures like cosine similarity require words to be represented as vectors. word2vec word embeddings are suited for this as they are learned representations of text which also carry semantic information.

- There are many possible criterion to decide on the number of bigrams. The normalized pointwise mutual information (NPMI) is a measure which allows the researcher to control bigram extraction to predominantly generate dataset-domain relevant bigrams.

- The statistical tests were chosen based on the objective, which was to compare the means of two columns of values. So t-tests and their non-parametric counterpart were used.

The detailed description of the rationale behind these design choices and how they helped work around design challenges encountered during the study is given in Chapter 3 of this thesis. Finally, the results of this study helped confirm the hypothesis that including bigram features to topic models improved topic modeling performance in terms of the generated topic's closeness to human-labeled topics.

5.4 Contributions and Impact

As pointed out in Chapter 2 *Literature Review*, education text mining has hardly focused on analyzing the educational material directly although there is a wealth of data waiting to be mined. Rather, the focus has been on either investigating the validity of pedagogical theories or assessing student performance and student engagement. Even within the instances of direct research on the educational content, lecture videos have hardly received attention. This makes the current study unique and one of its kind.

One of the motivations for research on topic models (discussed in the *research focus* section of Chapter 1) was to capture the range of concepts discussed in video lectures for building educational applications. As shown by this experiment, introducing bigrams is a simple but effective feature engineering technique to produce higher quality of topics and better document tags. Moreover, topic models create memory-

efficient, low-dimensional representation of documents (as a multinomial distribution over topics) which can be used for informational retrieval tasks like query matching (Heinrich, 2005). For these reasons, the combination of bigrams and topic models has the potential to be the algorithmic backbone of practical edu-apps like lecture video recommender systems and personalised learning paths for students.

Topic models solely built with unigram features often cannot account for subtle yet significant changes in themes when topic themes are explored at various granularities. For example, an LDA output topic on biology may have terms like *acids*,*fatty*,and *nucleic*. However, it actually brings together very distinct biological concepts *fatty acids* and *nucleic acids* (Mimno, Wallach, Talley, Leenders, & McCallum, 2011). Encoding *fatty_acids* and *nucleic_acids* as single bigram tokens would likely have averted the creation of mixed-concept topics. This too was a reason behind employing bigrams in this study, and true to expectations, this study provided experimental evidence for enhancement in topic modelling.

Another objective of this study was to empower educators to contribute to the model building process by bringing their expertise to the table. Insofar as lecture videos are concerned, the educators are the content creators and are natural stakeholders in the process of learning. Hence any educational application must be able to incorporate their valuable inputs. This study showed that using bag of words based models (like the LDA) and introducing bigram features allows for them to weigh in with their inputs, especially when it comes to choosing the appropriate features (unigram and bigram tokens).

On the other hand, feature engineering techniques like PCA transform the feature space to a set of indecipherable principal components. This is also an issue with neural network based models whose internal learned parameters don't offer any intuitive meaning.However, keeping in mind the many benefits of neural network based learning, this study does apply the word2vec algorithm (which is actually a neural network), but only on the LDA output. This way, the input features remain intact and human-understandable.

The study has some important limitations too. Some of it has been discussed in

the final section of the previous chapter. To establish the findings on more concrete grounds, further work is required. Few of the possible directions for future work are given in the following section.

5.5 Future Work and recommendations

The current study provides empirical evidence for the hypothesis that bigrams added topic models produce topics which are 'closer' to human-labeled topics. Although the study answers the *what* aspect of the question of the effect of bigrams on topic models, it has not sufficiently explored the *why* and *how*.

That requires deeper research which also takes into account the variety inherent in the bigrams themselves. Some bigrams are non-compositional - their components do not have a well-defined semantic interpretation in isolation for a compositional interpretation of the word combination to exist (Lau et al., 2013). These bigrams are usually multiword named entities , for example, phrases like *Los Angeles* or *Nelson Mandela*. Then there are the low compositional bigrams, for example, phrases like *Melting pot*, where the bigram is alluding to an environment where different ideas are assimilated rather than a physical *pot* in which ingredients are *melted*. Finally, there are the compositonal bigrams like *lung cancer* and *breast cancer* which are specific types of *cancer*.

If standardized collocations (bigrams which are a syntactic and semantic unit) specific to the document collection are created - this is another area where the educators/ subject experts can meaningfully contribute - the topic model performance for varying proportions of the above bigram types can be measured against that of the gold-standard set.

Such future work on bigrams can also consider other performance criteria like the Akaike information Criterion(AIC) or the Bayesian Information Criterion (BIC). AIC and BIC are log-likelihood scores which penalize the number of parameters. The parameters here are bigram count(W) and the number of topics(T). Including these measures in the research may help throw some light on the relationships between the

number of bigrams, the number of topics, and the bigram type.

Detailed investigations into the above mentioned aspects, backed by extensive studies, can help develop a convincing theoretical foundation for the role and impact of bigrams, and by extension n-grams, in 'explainable machine learning models' for text analytics. The answers to the *why* and *how* probably lay there.

References

Aitchison, J. (1982). The statistical analysis of compositional data. *Journal of the Royal Statistical Society: Series B (Methodological)*, *44*(2), 139-160.

Aletras, N., & Stevenson, M. (2013). Evaluating topic coherence using distributional semantics. *Proceedings of the 10th International Conference on Computational Semantics, IWCS 2013 - Long Papers*(2009), 13-22.

AlSumait, L., Barbará, D., Gentle, J., & Domeniconi, C. (2009). Topic significance ranking of LDA generative models. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *5781 LNAI*(PART 1). doi: 10.1007/978-3-642-04180-8_22

Andrieu, Christophe; De Freitas, Nando; Doucet, Arnaud; Jordan, M. I. (2003). An Introduction to MCMC for Machine Learning. *Machine learning*, *50*(1-2). Retrieved from `https://books.google.com/books?id=uQ-XeOYIRLQC{&}pgis=1` doi: 10.1023/A:1020281327116

Basu, Subhasree; Yu, Yi; Singh, Vivek K; Zimmermann, R. (2016). Videopedia: Lecture Video Recommendation for Educational Blogs Using Topic Modeling. *International Conference on Multimedia Modeling*, 238-250. doi: 10.1007/978-3-319-27671-7

Blei, D. M. (2012). Probabilistic topic models. *Communications of the ACM*, *55*(4), 77-84.

Blei, D. M., & Lafferty, J. D. (2007). Correlated topic models. *Advances in Neural Information Processing Systems*, 147.

Blei, D. M., & Lafferty, J. D. (2009). Topic models. *Text mining: classification, clustering, and applications*, *10*(71), 34.

Blei, D. M., Ng, A. Y., Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, *3*(Jan), 993-1022.

Bouma, G. (2009). Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL*, 31–40.

Bouyssou, D., Marchant, T., Pirlot, M., Tsoukias, A., & Vincke, P. (2006). *Evaluation and decision models with multiple criteria: Stepping stones for the analyst*. Springer Science \& Business Media.

Boyd-Graber, J., Hu, Y., & Mimno, D. (2017). Applications of topic models. *Foundations and Trends in Information Retrieval*, *11*(2-3). doi: 10.1561/1500000030

Cao, Z., Li, S., Liu, Y., Li, W., & Ji, H. (2015). A novel neural topic model and its supervised extension. *Proceedings of the National Conference on Artificial Intelligence*, *3*.

Chang, J., Boyd-Graber, J., Gerrish, S., Wang, C., & Blei, D. M. (2009). Reading tea leaves: How humans interpret topic models. *Advances in Neural Information Processing Systems 22 - Proceedings of the 2009 Conference*, 288-296.

Dasgupta, A., Drineas, P., Harb, B., Josifovski, V., & Mahoney, M. W. (2007). Feature selection methods for text classification. In *Proceedings of the 13th acm sigkdd international conference on knowledge discovery and data mining* (pp. 230–239).

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, *41*(6), 391-407.

Distante, D., Cerulo, L., Visaggio, A., & Leone, M. (2014). Enhancing online discussion forums with a topic-driven navigational paradigm: A plugin for the

moodle learning management system. *KDIR 2014 - Proceedings of the International Conference on Knowledge Discovery and Information Retrieval*, 97-106. doi: 10.5220/0005078600970106

Fei-Fei, L., & Perona, P. (2005). A bayesian hierarchical model for learning natural scene categories. *Proceedings 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, *2*, 524-531.

Ferreira-Mello, R., André, M., Pinheiro, A., Costa, E., & Romero, C. (2019). Text mining in education. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *9*(6). doi: 10.1002/widm.1332

Gerrish, S. M., & Blei, D. M. (2012). How they vote: Issue-adjusted models of legislative behavior. *Advances in Neural Information Processing Systems*, *4*, 2753-2761.

Goldstone, A., & Underwood, T. (2012). What can topic models of PMLA teach us about the history of literary scholarship. *Journal of Digital Humanities*, *2*(1).

Günther, E., & Quandt, T. (2016). Word counts and topic models: Automated text analysis methods for digital journalism research. *Digital Journalism*, *4*(1), 75-88. doi: 10.1080/21670811.2015.1093270

Gurcan, F., & Cagiltay, N. E. (2019). Big Data Software Engineering: Analysis of Knowledge Domains and Skill Sets Using LDA-Based Topic Modeling. *IEEE Access*, *7*. doi: 10.1109/ACCESS.2019.2924075

Guy, Retta and Marquis, G. (2016). The flipped classroom: A comparison of student performance using instructional videos and podcasts versus the lecture-based model of instruction. *Issues in Informing Science and Information Technology*, *13*(1), 1-13.

Hardisty, E., & Resnik, P. (2010). *Gibbs Sampling for the Uninitiated* (Tech. Rep.). Maryland Univ College Park Inst for Advanced Computer Studies.

Heinrich, G. (2005). *Parameter estimation for text analysis* (Tech. Rep.).

Hofmann, T. (2001). Unsupervised learning by probabilistic Latent Semantic Analysis. *Machine Learning*, *42*(1-2), 177-196. doi: 10.1023/A:1007617005950

Ignatow, G., & Mihalcea, R. (2017). *An introduction to text mining: Research design, data collection, and analysis*. SAGE Publications.

Jacobi, C., Van Atteveldt, W., & Welbers, K. (2016). Quantitative analysis of large amounts of journalistic texts using topic modelling. *Digital Journalism*, *4*(1), 89-106. Retrieved from `http://dx.doi.org/10.1080/21670811.2015.1093271` doi: 10.1080/21670811.2015.1093271

Jockers, M. L., & Mimno, D. (2013). Significant themes in 19th-century literature. *Poetics*, *41*(6), 750-769.

Karypis, G., & Han, E.-H. S. (2000). Fast supervised dimensionality reduction algorithm with applications to document categorization & retrieval. *Proceedings of the ninth international conference on Information and knowledge management*, 12-19. doi: 10.1145/354756.354772

Kay, R., & Kletskin, I. (2012). Evaluating the use of problem-based video podcasts to teach mathematics in higher education. *Computers \& Education*, *59*(2), 619-627.

Lau, J. H., Baldwin, T., & Newman, D. (2013). On collocations and topic models. *ACM Transactions on Speech and Language Processing*, *10*(3), 1-14. doi: 10.1145/2483969.2483972

Lewis, D. D., Yang, Y., Rose, T. G., & Li, F. (2004). RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, *5*.

Llorens, M. (2018). Text Analytics Techniques in the Digital World: Word Embeddings and Bias. *Irish Communications Review*, *16*(1). Retrieved from `https://arrow.dit.ie/icr/vol16/iss1/` doi: 10.21427/D7TJ05

Long, T., Logan, J., & Waugh, M. (2016). Students' perceptions of the value of using videos as a pre-class learning experience in the flipped classroom. *Tech Trends*, *60*(3), 245-252.

McCallum, A. K. (2002). Mallet: A machine learning for language toolkit (2002).

Meila, M. (2003). Comparing clusterings: an information based distance. *Journal of multivariate analysis*, *98*(5), 873-895.

Mikolov, Tomas; Sutskever, Ilya; Chen, Kai; Corrado, Greg S; Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. *Advances in neural information processing systems*, 3111-3119. doi: 10.18653/v1/d16-1146

Mimno, D., Wallach, H., Talley, E., Leenders, M., & McCallum, A. (2011). Optimizing semantic coherence in topic models. In *Proceedings of the 2011 conference on empirical methods in natural language processing* (pp. 262–272).

Nagy, J. T. (2018). Evaluation of online video usage and learning satisfaction: An extension of the technology acceptance model. *International Review of Research in Open and Distributed Learning*, *19*(1).

Newman, D., Lau, J. H., Grieser, K., & Baldwin, T. (2010). Automatic evaluation of topic coherence. *NAACL HLT 2010 - Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Proceedings of the Main Conference*(June), 100-108.

Nokel, M., & Loukachevitch, N. (2015). A Method of Accounting Bigrams in Topic Models. , 1-9. doi: 10.3115/v1/w15-0901

Pauca, V. P., Shahnaz, F., Berry, M.W., Plemmons, R. (2004). Text mining using non-negative matrix factorizations. *Proceedings of the 2004 SIAM International Conference on Data Mining*, 452-456.

Quinn, K. M., Monroe, B. L., Colaresi, M., Crespin, M. H., & Radev, D. R. (2010). How to analyze political attention with minimal assumptions and costs. *American Journal of Political Science*, *54*(1), 209-228. doi: 10.1111/j.1540-5907.2009.00427.x

Řehůřek, R., & Sojka, P. (2010, May 22). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks* (pp. 45–50). Valletta, Malta: ELRA.

Romero, C., & Ventura, S. (2017). Educational data science in massive open online courses. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *7*(1). doi: 10.1002/widm.1187

Sivic, J., Russell, B. C., Zisserman, A., Freeman, W. T., & Efros, A. A. (2008). Unsupervised discovery of visual object class hierarchies. *Proceedings - 2008 IEEE Conference on Computer Vision and Pattern Recognition*, 1-8.

Snyder, T. D., De Brey, C., & Dillow, S. A. (2018). Digest of Education Statistics 2016, NCES 2017-094. *National Center for Education Statistics*.

Soroka, S. N., Stecula, D. A., & Wlezien, C. (2015). It's (Change in) the (Future) Economy, Stupid: Economic Indicators, the Media, and Public Opinion. *American Journal of Political Science*, *59*(2), 457-474. doi: 10.1111/ajps.12145

Steyvers, M., & Griffiths, T. (2010). Probabilistic Topic Models. *Latent Semantic Analysis: A Road To Meaning*, *3*(3).

Valjataga, T; Poldoja, H. L. M. (2011). Open online courses: Responding to design challenges. *Stanford University, H-STAR Institute, USA; to Associate Professor Jukka M. Laitam{\"a}ki, from New York University, USA, and to Professor Yngve Troye Nordkvelle from Lillehammer University*, *68*.

Wallach, H. M. (2006). Topic Modeling : Beyond Bag-of-Words. *Proceedings of the 23rd international conference on Machine learning*(1), 977-984.

Wang, Y., & Xu, W. (2018). Leveraging deep learning with LDA-based text analytics to detect automobile insurance fraud. *Decision Support Systems*, *105*, 87-95. Retrieved from `https://doi.org/10.1016/j.dss.2017.11.001` doi: 10.1016/j.dss.2017.11.001

Wei, X. (2007). *TOPIC MODELS IN INFORMATION RETRIEVAL* (Unpublished doctoral dissertation). MASSACHUSETTS UNIV AMHERST.

Wei, X., & Croft, W. B. (2006). LDA-based document models for ad-hoc retrieval. *Proceedings of the Twenty-Ninth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, *2006*, 178-185. doi: 10.1145/1148170.1148204

West, M., & Harrison, J. (2006). *Bayesian forecasting and dynamic models*. Springer Science \& Business Media.

Zhu, Q., Shyu, M. L., & Wang, H. (2013). VideoTopic: Content-based video recommendation using a topic model. *Proceedings - 2013 IEEE International Symposium on Multimedia, ISM 2013*, 219-222. doi: 10.1109/ISM.2013.41

Appendix A

Additional content

A.1 Python Script for corpus creation

```
#obtain andrew machine learning playlist video IDs

import googleapiclient.discovery
from urllib.parse import parse_qs, urlparse

#extract playlist id from url
url = 'https://www.youtube.com/playlist?list=PLLssT5z_DsK-h9vYZkQkYNWcItqhlRJLN'
query = parse_qs(urlparse(url).query, keep_blank_values=True)
playlist_id = query["list"][0]


youtube = googleapiclient.discovery.build("youtube", "v3", developerKey = "AIzaSyBW-ag7yd7ouIIT

request = youtube.playlistItems().list(
    part = "snippet",
    playlistId = playlist_id,
    maxResults = 50
)
response = request.execute()

playlist_items = []
while request is not None:
    response = request.execute()
    playlist_items += response["items"]
    request = youtube.playlistItems().list_next(request, response)


#print([
#    f'{t["snippet"]["resourceId"]["videoId"]}'
#    for t in playlist_items
#])
video_IDs_list = []
for t in playlist_items:
    video_IDs_list.append(t["snippet"]["resourceId"]["videoId"])

#print (video_IDs_list)
```

Figure A.1: Obtaining Prof.Andrew playlist Video IDs

```python
#get the andrew machine learning playlist transcripts using the video IDs got above

'''This is a multiline comment string.
In the following code block we will install the
youtube transcripts module of python and use it
to obtain video transcripts
'''
from youtube_transcript_api import YouTubeTranscriptApi

video_ids = video_IDs_list

out, _  = YouTubeTranscriptApi.get_transcripts(video_ids)

text_out = {}
for video_id in out:
    text_out[video_id] = ' '.join([x.get('text', '') for x in out[video_id]])

#print(text_out)

import pandas as pd
text_list = list(text_out.items())
df = pd.DataFrame(text_list,columns=['video_id','transcript'])
#print(df)

df.to_csv('andrew_ng_dataset.csv',index =False)
```

Figure A.2: Using Video IDs to get Prof.Andrew playlist's transcripts

```python
#get video IDs and transcripts for arti machine learning playlist

import googleapiclient.discovery
from urllib.parse import parse_qs, urlparse

#extract playlist id from url
url = 'https://www.youtube.com/playlist?list=PLUZjIBGiCHFfRJwflq6NqU3CuiPhAhSfi'
query = parse_qs(urlparse(url).query, keep_blank_values=True)
playlist_id = query["list"][0]


youtube = googleapiclient.discovery.build("youtube", "v3", developerKey = "AIzaSyBW-ag7yd7ouIIT

request = youtube.playlistItems().list(
    part = "snippet",
    playlistId = playlist_id,
    maxResults = 50
)
response = request.execute()

playlist_items = []
while request is not None:
    response = request.execute()
    playlist_items += response["items"]
    request = youtube.playlistItems().list_next(request, response)

#print(f"total: {len(playlist_items)}")
arti_video_IDs_list = []
for t in playlist_items:
    arti_video_IDs_list.append(t["snippet"]["resourceId"]["videoId"])

#print (arti_video_IDs_list)
```

Figure A.3: Obtaining Prof.Arti playlist's Video IDs

```
from youtube_transcript_api import YouTubeTranscriptApi

video_ids = arti_video_IDs_list

out, _  = YouTubeTranscriptApi.get_transcripts(video_ids)

text_out = {}
for video_id in out:
    text_out[video_id] = ' '.join([x.get('text', '') for x in out[video_id]])

#print(text_out)

import pandas as pd
text_list = list(text_out.items())
df = pd.DataFrame(text_list,columns=['video_id','transcript'])
#print(df)

df.to_csv('arti_ramesh_dataset.csv',index =False)
```

Figure A.4: Using Video IDs to get Prof.Arti playlist's transcripts

```
#scrape video titles from the youtube playlists of arti and andrew

from bs4 import BeautifulSoup as bs
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time

from selenium.webdriver.chrome.service import Service

service = Service('C:\\Users\\radhagopinath\\Anaconda3\\Lib\\site-packages\\selenium\\webdriver
service.start()
driver = webdriver.Remote(service.service_url)

#driver = webdriver.Chrome('C:\\Users\\radhagopinath\\Desktop')
url = "https://www.youtube.com/playlist?list=PLUZjIBGiCHFfRJwflq6NqU3CuiPhAhSfi"
driver.get(url)

elem = driver.find_element_by_tag_name('html')
elem.send_keys(Keys.END)
time.sleep(3)
elem.send_keys(Keys.END)

innerHTML = driver.execute_script("return document.body.innerHTML")

page_soup = bs(innerHTML, 'html.parser')
res = page_soup.find_all('span',{'class':'style-scope ytd-playlist-video-renderer'})

arti_playlist_titles = []
for video in res:
    if video.get('title') != None:
        arti_playlist_titles.append((video.get('title')))

driver.close()
```

Figure A.5: Web scraping Prof.Arti playlist's video titles

```
'''
from bs4 import BeautifulSoup as bs
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time

from selenium.webdriver.chrome.service import Service
'''
service = Service('C:\\Users\\radhagopinath\\Anaconda3\\Lib\\site-packages\\selenium\\webdriver
service.start()
driver = webdriver.Remote(service.service_url)

#driver = webdriver.Chrome('C:\\Users\\radhagopinath\\Desktop')
url = "https://www.youtube.com/playlist?list=PLLssT5z_DsK-h9vYZkQkYNWcItqhlRJLN"
driver.get(url)

elem = driver.find_element_by_tag_name('html')
elem.send_keys(Keys.END)
time.sleep(3)
elem.send_keys(Keys.END)

innerHTML = driver.execute_script("return document.body.innerHTML")

page_soup = bs(innerHTML, 'html.parser')
res = page_soup.find_all('span',{'class':'style-scope ytd-playlist-video-renderer'})

andrew_ng_playlist_titles = []
for video in res:
    if video.get('title') != None:
        andrew_ng_playlist_titles.append((video.get('title')))

driver.close()
```

Figure A.6: Web scraping Prof.Andrew playlist's video titles

```
In [7]: arti_playlist_titles

Out[7]: ['Sample spaces, events, random variables, and probability axioms',
         'Probability Mass Function and Sum Rule',
         'Probability Mass Function and Sum Rule Venn Diagram',
         'Introduction to Supervised Learning',
         'Understanding Supervised Learning using an Simple Dataset',
         'Supervised Learning Hypothesis Spaces',
         'Inductive Learning: Hypothesis Spaces',
         'Supervised Learning Steps',
         'Overfitting',
         'Decision Trees Hypothesis Spaces',
         'Information Gain in Decision Trees',
         'Decision Tree Algorithm',
         'Decision Trees: Entropy Simple Example',
         'Calculating Decision Tree Entropy (Simple Example)',
         'Decision Tree Information Gain Example 1',
         'How to Calculate Decision Tree Information Gain (Illustrative Example)',
         'Calculating Information Gain in Decision Trees',
         'Decision Trees Overfitting and Pruning',
         'Point Estimation: Maximum Likelihood Estimation',
         'Hoeffding's Inequality -- Part I',
         'Hoeffding's Inequality - Part II',
         'Bayesian Learning Introduction',
```

Figure A.7: A sample of video titles from Prof.Arti's playlist

```
andrew_ng_playlist_titles
```

```
['Lecture 1.1 - What Is Machine Learning - [ Machine Learning | Andrew Ng ]',
 'Lecture 1.2 - Supervised Learning - [ Machine Learning | Andrew Ng ]',
 'Lecture 1.3 - Unsupervised Learning - [ Machine Learning | Andrew Ng]',
 'Lecture 2.1 - Linear Regression With One Variable | Model Representation - Andrew Ng',
 'Lecture 2.2 - Linear Regression With One Variable | CostFunction - Andrew Ng',
 'Lecture 2.3 - Linear Regression With One Variable | Cost Function Intuition #1 | Andrew N
g',
 'Lecture 2.4 - Linear Regression With One Variable | Cost Function Intuition #2 | Andrew N
g',
 'Lecture 2.5 - Linear Regression With One Variable | Gradient Descent - [ Andrew Ng]',
 'Lecture 2.6 - Linear Regression With One Variable | Gradient Descent Intuition - [ Andrew
Ng]',
 'Lecture 2.7 - Linear Regression With One Variable | Gradient Descent For Linear Regressio
n',
 "Lecture 2.8 - What's Next - [ Machine Learning | Andrew Ng | Stanford University]",
 'Lecture 3.1 - Linear Algebra Review | Matrices And Vectors - [ Machine Learning | Andrew N
g]',
 'Lecture 3.2 - Linear Algebra Review | Addition And Scalar Multiplication - [Andrew Ng]',
 'Lecture 3.3 - Linear Algebra Review | Matrix Vector Multiplication - [ Machine Learning |
```

Figure A.8: A sample of video titles from Prof.Andrew's playlist

```
import pandas as pd
arti= pd.read_csv('arti_ramesh_dataset.csv')
andrew= pd.read_csv('andrew_ng_dataset.csv')

andrew['Video title'] = andrew_ng_playlist_titles
arti['Video title'] = arti_playlist_titles

arti.to_csv('arti_ramesh_dataset.csv',index =False)
andrew.to_csv('andrew_ng_dataset.csv',index =False)
```

```
arti
```

	video_id	transcript	Video title
0	hXMib_I7IkY	in this lecture we will be looking at some bas...	Sample spaces, events, random variables, and p...
1	CRhsbvifgsM	in this lecture we will introduce probability ...	Probability Mass Function and Sum Rule
2	Cg4prSGAtSw	let's go back to the Venn diagram to understan...	ProbabilityMass Function and Sum Rule Venn Di...
3	bvjBgC2riVc	hello everyone in this lecture we will be lear...	Introduction to Supervised Learning
4	MNdJhVD_fx0	alright continuing our previous discussion on ...	Understanding Supervised Learning using an Sim...
...	...	...	...
62	87wi7BA0Im8	hello everyone welcome back in this video we a...	Support Vector Machines: Illustrative Example
63	97J6OKMA7ws	hello everyone in this video we will consider ...	Example Problem explaining Support Vector Mac...
64	AasatTRGVhI	hello everyone welcome back in this video we a...	K Nearest Neighbors
65	TD_aDcH7Ki8	hello everyone welcome back in this video we a...	Bias Variance Tradeoff
66	h3JoGqyN6HQ	hello everyone welcome back in this video we a...	K nearest neighbors: Choosing k

67 rows × 3 columns

Figure A.9: Dataframe showing Prof.Arti playlist of 67 videos

andrew

	video_id	transcript	Video title
0	PPLop4L2eGk	What this machine learning? in this video, we ...	Lecture 1.1 — What Is Machine Learning — [Mac...
1	bQI5uDxrFfA	in this video I'm going to define whether it's...	Lecture 1.2 — Supervised Learning — [Machine ...
2	jAA2g9ItoAc	in this video we'll talk about the second majo...	Lecture 1.3 — Unsupervised Learning — [Machin...
3	kHwIB_j7Hkc	our first learning algorithm will be linear re...	Lecture 2.1 — Linear Regression With One Varia...
4	yuH4iRcggMw	in this video we'll define something called th...	Lecture 2.2 — Linear Regression With One Varia...
...	...	...	...
107	CykIW9hFK24	in this in the next few videos I wanted to tel...	Lecture 18.1 — Application Example Photo OCR |...
108	9tbxsbQ2Z0A	in the previous video we talked about the phot...	Lecture 18.2 — Application Example Photo OCR |...
109	kvtLGRthW4U	seen over and over that one of the most reliab...	Lecture 18.3 — Application Example Photo OCR |...
110	8fx1nSFQqkk	in earlier videos that said over and over that...	Lecture 18.4 — Application Example Photo OCR |...
111	m_0IDnW-WD0	welcome to the final video of this machine lea...	Lecture 19 — Conclusion Summary And ThankYou —...

112 rows × 3 columns

Figure A.10: Dataframe showing Prof.Andrew playlist of 112 videos