Play With PDFs

Ishaan Kumar, Arnab Ghosh, TV Prabhakar

Indian Institute Of Technology, Kanpur

Abstract

In today's world PDF has become the document of choice with Presentations, Research Papers, Resumes, entire books have been presented in the form of PDFs. The rise of popularity of this format is due to its uniformity in displaying content across a wide variety of Browsers, Devices and Operating Systems. Its platform independent viewing across the proprietary software such as Adobe Reader and Open Source Software such as Evince Document Viewer readers as well gives a much needed uniformity. Such has been the impact of PDFs on software is that most of the softwares which allow creation of content such as Powerpoint, Word , Excel etc provides us the option of converting into PDF format.

Driven by the immense popularity of PDFs and the need for a software that can create custom PDF's from a repository of PDFs for future reference, we created an ubuntu application that would automatically tag the individual pages of the PDF based on the textual content of the pages of the PDF. Results were fetched according to the search query run on individual pages of in general larger PDFs. We provided the users an option of selecting the given pages that came up with the result of the query and merging several of the pages of various PDFs across several of the queries based on the tags that we generated. We showed the user automatically generated tags and provided the user an option to change and edit the tags in case he was not satisfied with the automatically generated tag of the given page of the PDF.

Several times not only the page containing the given tag but the adjacent pages would be important as well for the current task at hand so we have provided an option of opening the entire pdf from which the page came from and have provided another mode whereby we can manually mixmatch several pages from different PDFs based on the ranges that are required from the specified PDFs.

It can be used to create a single presentation, taking relevant slides from several different presentations to create a single presentation. It could be useful especially among the **academia** to keep all relevant information across a wide variety of resources bundled together as a single PDF. It can be used by the students to make notes out of several PDFs that they come across dealing with a single topic and create a single document for future reference at an apt time.

It can be used by lawyers to mark important parts of a textbook that is relevant for the defense and attack with respect to the particular case.

Keywords: PDF, Tagging, PyGTK, Wordnet

1. Introduction

After deciding upon what we wanted to build, we first went on to build the backend of the application. MongoDB was decided as the database engine for its high scalability, in case we wanted to make the application a server-side application. For the automatic tagging we decided to use Latent Semantic Indexing. We had the option of selecting the dictionary on which the Latent Semantic Indexing would work but we thought that if we choose the tags only from the predefined dictionary then we might miss out on people's names as the tag. For example a paper by Professor TV Prabhakar must have Prabhakar as part of its tags hence we decided on a dynamic model whereby we add the items to the dictionary as they come along with PDFs and based upon the users choice of PDF domains they choose whether the tagging was found to be better .

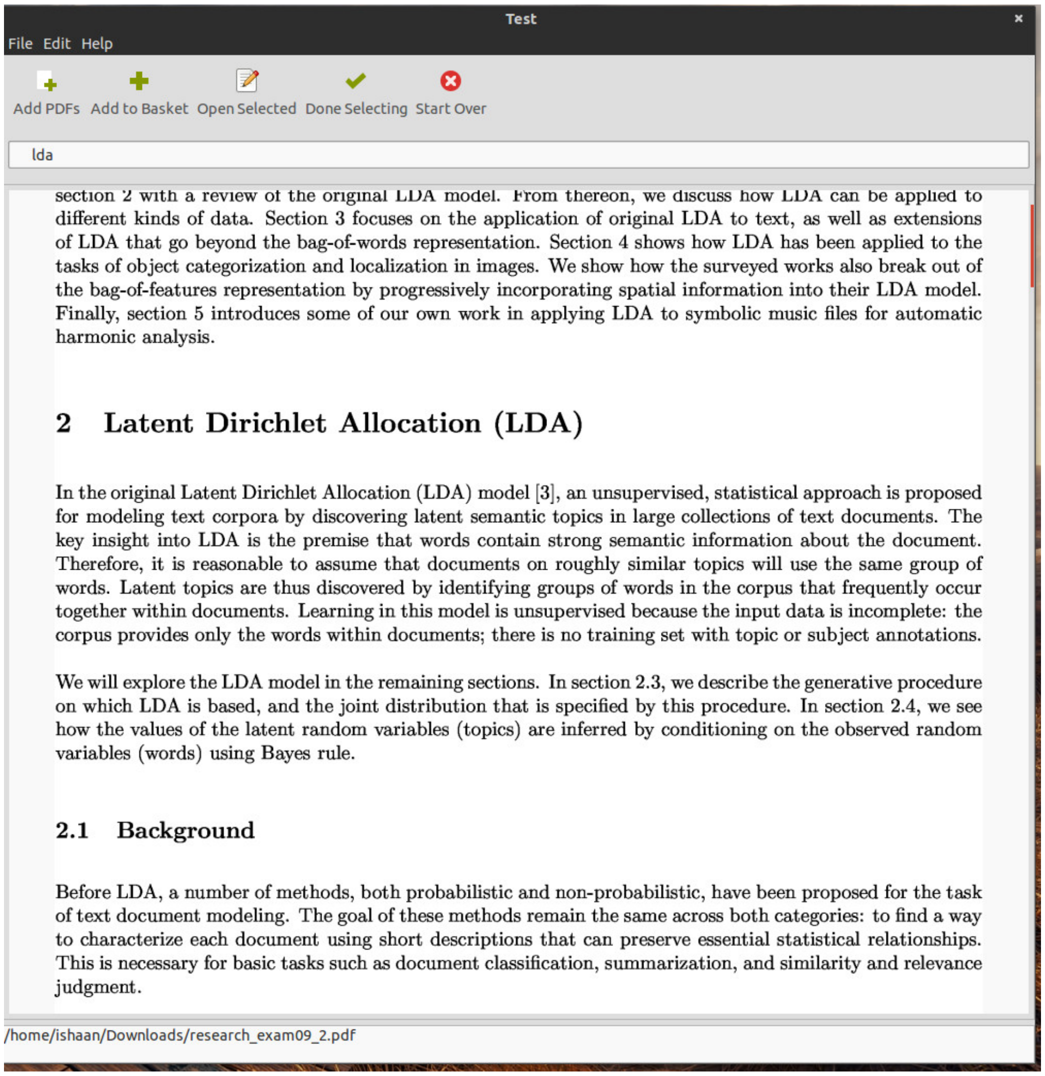
We have used the Wordnet to find semantically related words and added these as well to the database for the tags because we don't want to penalize the user for a slightly different search query for example we want the tag "love" to be also associated with "affection" and like words.

After completing the tagging we moved on to the scripts for the separation of PDFs and merging them given the ranges and the PDFs they had been taken from .

After completion of the backend , we explored on the possible choices for the front end of the app, the prospect of learning PyGTK to make cross platform apps charmed us and we decided that we would go ahead with PyGTK. Currently the app can be bundled as a .deb file and can be installed on any debian based machine. We would be extending its usability by building a mobile and tablet (Android and iOS) compatible app for wide usage of our software for the aid of one and all .

2. User Interface

The user interface of our application is made using PyGTK and targets the GTK+3.4 systems. The design of the interface is kept minimal and intuitive. The main problem we faced while working with the interface was the proper rendering of relevant pages after running the database query. Since a full fledge in-application pdf renderer was not available, we came up with the solution of using screenshots of pages, as Gtk provides native support for image rendering. So whenever a page needs to be shown in the result view, its high resolution image is loaded instead.



section 2 with a review of the original LDA model. From thereon, we discuss how LDA can be applied to different kinds of data. Section 3 focuses on the application of original LDA to text, as well as extensions of LDA that go beyond the bag-of-words representation. Section 4 shows how LDA has been applied to the tasks of object categorization and localization in images. We show how the surveyed works also break out of the bag-of-features representation by progressively incorporating spatial information into their LDA model. Finally, section 5 introduces some of our own work in applying LDA to symbolic music files for automatic harmonic analysis.

2 Latent Dirichlet Allocation (LDA)

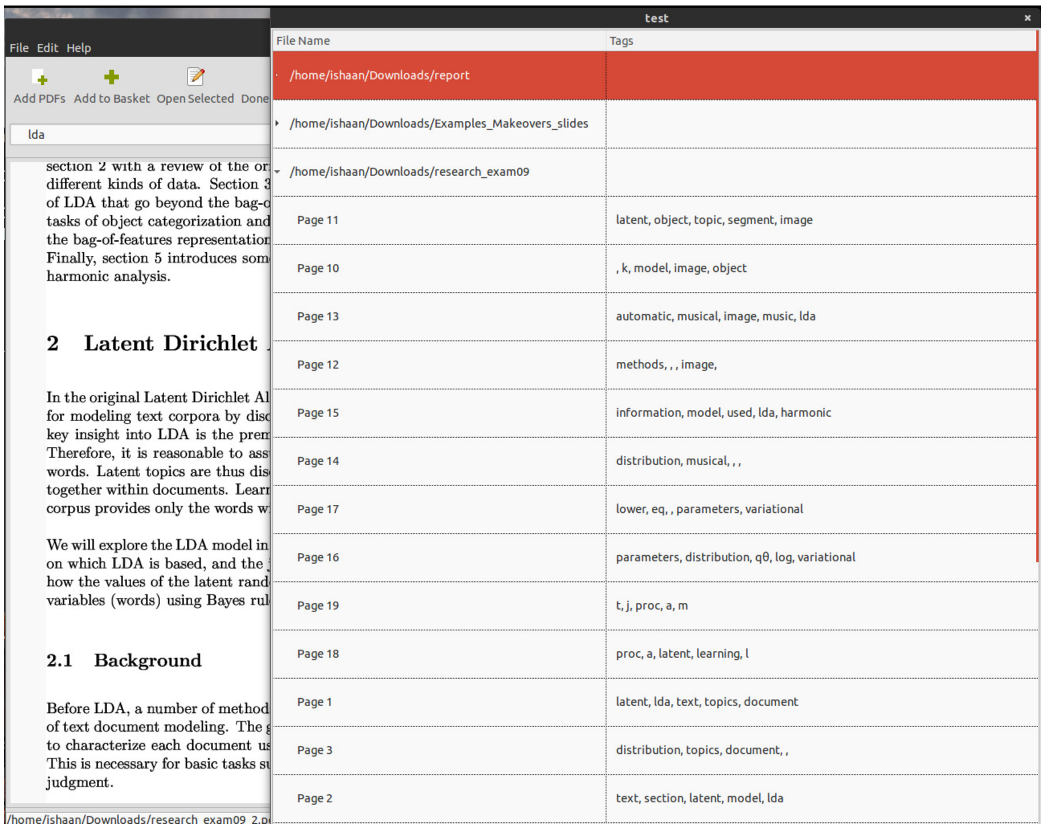
In the original Latent Dirichlet Allocation (LDA) model [3], an unsupervised, statistical approach is proposed for modeling text corpora by discovering latent semantic topics in large collections of text documents. The key insight into LDA is the premise that words contain strong semantic information about the document. Therefore, it is reasonable to assume that documents on roughly similar topics will use the same group of words. Latent topics are thus discovered by identifying groups of words in the corpus that frequently occur together within documents. Learning in this model is unsupervised because the input data is incomplete: the corpus provides only the words within documents; there is no training set with topic or subject annotations.

We will explore the LDA model in the remaining sections. In section 2.3, we describe the generative procedure on which LDA is based, and the joint distribution that is specified by this procedure. In section 2.4, we see how the values of the latent random variables (topics) are inferred by conditioning on the observed random variables (words) using Bayes rule.

2.1 Background

Before LDA, a number of methods, both probabilistic and non-probabilistic, have been proposed for the task of text document modeling. The goal of these methods remain the same across both categories: to find a way to characterize each document using short descriptions that can preserve essential statistical relationships. This is necessary for basic tasks such as document classification, summarization, and similarity and relevance judgment.

/home/ishaan/Downloads/research_exam09_2.pdf



section 2 with a review of the or
different kinds of data. Section 3
of LDA that go beyond the bag-o
tasks of object categorization and
the bag-of-features representation
Finally, section 5 introduces som
harmonic analysis.

2 Latent Dirichlet

In the original Latent Dirichlet Al
for modeling text corpora by disc
key insight into LDA is the prem
Therefore, it is reasonable to ass
words. Latent topics are thus dis
together within documents. Learn
corpus provides only the words w

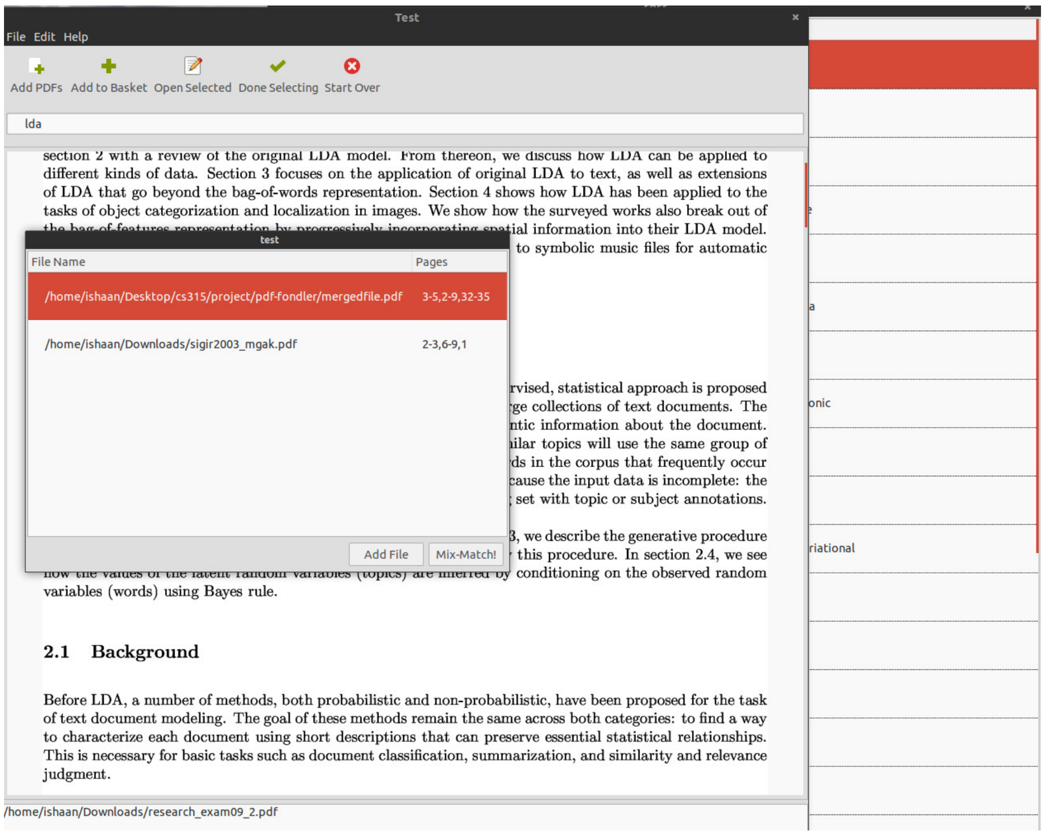
We will explore the LDA model in
on which LDA is based, and the
how the values of the latent rand
variables (words) using Bayes rul

2.1 Background

Before LDA, a number of method
of text document modeling. The
to characterize each document us
This is necessary for basic tasks su
judgment.

/home/ishaan/Downloads/research exam09 2.p

File Name	Tags
/home/ishaan/Downloads/report	
/home/ishaan/Downloads/Examples_Makeovers_slides	
/home/ishaan/Downloads/research_exam09	
Page 11	latent, object, topic, segment, image
Page 10	, k, model, image, object
Page 13	automatic, musical, image, music, lda
Page 12	methods, , , image,
Page 15	information, model, used, lda, harmonic
Page 14	distribution, musical, , ,
Page 17	lower, eq, , parameters, variational
Page 16	parameters, distribution, qθ, log, variational
Page 19	t, j, proc, a, m
Page 18	proc, a, latent, learning, l
Page 1	latent, lda, text, topics, document
Page 3	distribution, topics, document, ,
Page 2	text, section, latent, model, lda



section 2 with a review of the original LDA model. From thereon, we discuss how LDA can be applied to different kinds of data. Section 3 focuses on the application of original LDA to text, as well as extensions of LDA that go beyond the bag-of-words representation. Section 4 shows how LDA has been applied to the tasks of object categorization and localization in images. We show how the surveyed works also break out of the bag-of-features representation by progressively incorporating spatial information into their LDA model. … to symbolic music files for automatic

File Name	Pages
/home/ishaan/Desktop/cs315/project/pdf-fondler/mergedfile.pdf	3-5,2-9,32-35
/home/ishaan/Downloads/sigir2003_mgak.pdf	2-3,6-9,1

…rvised, statistical approach is proposed …ge collections of text documents. The …ntic information about the document. …ilar topics will use the same group of …ds in the corpus that frequently occur …cause the input data is incomplete: the … set with topic or subject annotations.

…3, we describe the generative procedure … this procedure. In section 2.4, we see how the values of the latent random variables (topics) are inferred by conditioning on the observed random variables (words) using Bayes rule.

2.1 Background

Before LDA, a number of methods, both probabilistic and non-probabilistic, have been proposed for the task of text document modeling. The goal of these methods remain the same across both categories: to find a way to characterize each document using short descriptions that can preserve essential statistical relationships. This is necessary for basic tasks such as document classification, summarization, and similarity and relevance judgment.

/home/ishaan/Downloads/research_exam09_2.pdf

3. Overview

The user types the query in the search and the list of appropriate pages is loaded in the result view. The user can then select the relevant pages and add them to the basket. Once all the required pages are added to the basket, the user can click 'Done selecting' to save the resultant file. To see the page displayed in its original context, the user can also open the respective pdfs of pages currently contained in the basket by clicking 'Open Selected'. The standard functionality of indexing and starting over is also provided.

3.1. Manual Mode

In manual mode, complete control is provided to the user so that he can see the files indexed so far, as well as modify tags of each page.

3.2. Mix-Match Mode

In this mode, user is supposed to add the files one by one and then is expected to specify the page numbers he wants in the resultant document, in the adjacent column.

4. What We Learnt

From this endeavor we learnt about Natural Language Processing and preprocessing of data to extract the relevant portions of text and to exclude stopwords so that we do not get spurious tags. We learnt about automatic tagging using Latent Semantic Indexing which is the state of the art tagger available at the time.

We learnt about Wordnet which is a Natural Language Processing tool represented as a directed graph with respect to the synonym sets that had been looked through the vast literature available. While inserting the tags into the database we also insert synonyms from the wordnet so that we don't miss out on a search for "affection" if the actual tag is "love"

We tried processing the query and also looked at papers for spelling correction and automatic prediction of the tag that is to be queried. We came across a dynamic programming setting in which it tries to predict the ensuing word using the previously entered tags.

We learnt about the PyGTK software development suite and learnt to integrate python scripts along with an user interface which is intuitive and self evident. We learnt about the lifecycle of a window in this format and how the control flows from one window to a dialog box. We learnt about creating good bundled software from the available tools.

5. Conclusion

The software that we have developed has been working pretty nicely with the PDFs that we have tested with. It works well with Academic Papers , Lecture Slides , Text intensive books . We hope to build a complete functioning product by the end of our summer vacation and distribute it as an Android/iOS app. The tagging seems to work well with the available text that we have tried our hands upon. The User Interface is pretty well commended by many who have used it. It is a pretty nice idea if we can perfect our product.

6. Acknowledgement

We express our gratitude towards Professor TV Prabhakar for providing an opportunity to work on such a good idea which is not currently provided by any available software at the current moment.

We thank all the Teaching Assistants Saurabh Srivastava, Sumit Kalra and Ashish Agarwal for their feedback .