Information Extraction from Textbooks for Physics Knowledge

Arman Bolat
ECE MS '15
abolat@andrew.cmu.edu

Bryan Tan
ECE MS '15
bstan@andrew.cmu.edu

Abstract

The goal of this project is to build a classifier that is able to extract physics knowledge from textbooks. Information extraction of textbooks is a useful problem to tackle because of the many applications that the extracted information could be used for. In the course of this project, textbooks in the PDF will be converted to a text format, and classifiers of different complexity will be tested for accuracy. Training data sets will be created by extracting laws, theorems, definitions, and other knowledge in the text format textbook, and attaching any relevant material, such as context or metadata.

1 Introduction

As more and more information is becoming digitalized, the importance of information extraction through machine learning methods has increased. The goal of this project is to extend machine learning methods to extract physics knowledge from textbooks. In particular, this project will focus on extracting theorems, laws, and definitions relevant to newtonian physics within a textbook. The resulting knowledge will be presented as a glossary, corresponding words and phrases with their respective definitions.

Information extraction from textbooks has a number of practical applications. For one, a problem from a textbook can be analyzed, and the relevant laws, theorems, and equations can be determined by using the knowledge extracted from textbooks, and potentially even a step by step solution for the problem could be provided. The extracted physics knowledge from a given textbook can also show a quick summary of the material covered in the textbook and allow for a classification of the textbook based on depth and breadth of material covered. Even more advanced would to be build a customized textbook through extracted information based on personalized needs.

The goal of this project is to extract physics knowledge from a set of textbooks. The rest of this report is organized as follows: related works to this project will be covered in section 2, the proposed method of creating and training the classifier will be covered in section 3, and preliminary results will be covered in section 4.

2 Related Works

The employment of information extraction and machine learning techniques for textbooks has not seen much work. However, there has been research into the application of knowledge extraction, such as the implementation of some of the example applications mentioned in the introduction. It is possible that the ease of access to resources that provide the information and data necessary (without the use of textbooks specifically) for such research is an explanation for the lack of research into information extraction for textbooks.

[Min Joo Soon Diagram Understanding] explores a method of automatically solving geometry questions. To accomplish this, both the diagram and the accompanying text must be understood. A visual definition of key concepts and terms must be learned, instead of a textual definition as in this project.

[Comprehension burden] attempts to create a tool that is able to assess the comprehension burden, or the difficulty in understanding the textbook material. A property of well-written textbooks is that concepts are presented in sequential order, such that each concept is explained before it occurs in examples or other concepts. A textbook with high comprehension burden thus does not present concepts in sequential order. In order for this to be determined, the occurrences and definition of concepts must be found.

[Textbook enrichment through images] enriches the learning experience by automatically finding relevant images for each chapter. In order to do so, key nouns, phrases, and ideas must be found, as well as their corresponding relevant images. This is similar to this project in that key ideas must be determined. [Identifying enrichment] also applies similar techniques to find key concept phrases.

[Concept hierarchy extraction] aims to create a concept hierarchy of a textbook. Concepts are identified and extracted by using Wikipedia as a resource. This is very similar to this project - the intermediate goal of extracting concepts is shared, albeit without the use of an external source as a direct reference. The end goal is more complex in the hierarchy extraction paper, as the relationship between concepts must be determined. [BBookX], a system designed at Penn State, extends upon research done in concept hierarchy extraction to automatically create open versions of textbooks by updating the textbook content through online resources such as Wikipedia.

3 Proposed Method

3.1 Training Data Extraction

It is important to generate the training data with the appropriate set of features that would be analyzed by the classifier. The training data used for this project will be extracted from the physics textbooks from The National Council of Educational Research and Training website (http://www.ncert.nic.in/index.html), and specifically the chapters on the Newtonian physics. For each sentence in the textbooks, a single training data point will be created, which will be represented as a sample object. The sample object will include all the features that could be used by the classifier.

Specifically, each sample object will include:

- the sentence itself
- the sentence before the current sentence
- the sentence following the current sentence
- the boolean value representing whether the sentence is bold
- the boolean value representing whether the sentence contains a mathematical formula or expression
- the type or label of the sample object (theory/law, definition, or none)

The classifiers will be using not only the content of the sentence but also the context of where the sentence is located as well as the metadata of the sentence. By including a greater number of features, a more general training data set for different classifiers can be created. Since different classifiers might be considering different set of features, it is better to include all possible features which decide if a particular sentence is a piece of physics knowledge or just a regular sentence.

3.2 Naive Bayes Implementations

This project will begin with implementing a simple Naive Bayes classifier and compare the accuracy by gradually increasing the number of features the classifier considers. Specifically, a classifier implementing a bag-of-words model which only looks at the words used in the sentence will be created. After the completion of the simple classifier, the implementation will be modified to account

for the sentence metadata as well. The final Naive Bayes implementation will take into account all features of a sentence in order to classify it.

4 Preliminary Results

4.1 Simple Bag-of-words Approach

The initial approach of using a bag-of-words model seemed fairly successful with a training data accuracy of 81.70% and a testing data accuracy of 71.72%. However, upon inspection, it was clear that this initial approach was failing: for the testing data, the accuracy for general (non-knowledge) sentences was 95.58%, but the accuracy for knowledge sentences was only 3.17%. This indicated that the bag-of-words model had a feature set that was too large to effectively classify the sentences, which all had a relatively small number of features, or words, compared to other problems solved by a bag-of-words approach.

The poor results highlighted areas of interest worth tackling to improve the classification accuracy: adding more weight to important and distinguishing phrases or words, and taking into account the other features of the text (namely, the existence of bolded words). The script used to split textbook chapters up into sentences was also further improved to handle edge cases more effectively.

4.2 Bag-of-words Approach with Weighting

After implementing the changes to fix problems pointed out in the initial approach, the new classifier improved in training data knowledge sentence classification accuracy, going from 17.83% to 21.99%, but regressed in testing data knowledge sentence classification accuracy, going from 3.17% to 1.35%. The weighting did not improve the classification accuracy, as the size of the vector of indicator features was still too large in comparison to the size of the sample sentences.

4.3 Naive Bayes with Key Phrases

The results of the second iteration showed that a bag-of-words approach would not work, even after adding weights and accounting for different features. Thus, instead of using a bag-of-words approach, a few key phrases such as ”is defined” and ”is called” were selected and used as the indicator features, along with the presence of bolded words. The result of the simpler classifiers were substantially better: 35.29% classification accuracy for testing data knowledge sentences. After looking through the misclassified examples, it became clear that the training data itself actually had many misclassified samples. After going through the data set to look for any missed definitions, the classification accuracy rose to 76.47%.

Training Data Set Classification Accuracies			
Classifier	Knowledge Sentence	General Sentence	Overall
Simple bag-of-words	17.83%	98.40%	81.70%
Weighted bag-of-words	21.99%	98.01%	86.69%
Naive Bayes with Key Phrases	43.06%	96.68%	95.66%
Naive Bayes, fixed data set	86.11%	96.55%	96.45%

Testing Data Set Classification Accuracies			
Classifier	Knowledge Sentence	General Sentence	Overall
Simple bag-of-words	3.17%	95.58%	71.72%
Weighted bag-of-words	1.35%	95.41%	81.15%
Naive Bayes with Key Phrases	35.29%	97.80%	96.18%
Naive Bayes, fixed data set	76.47%	97.80%	97.25%

5 Citations, figures, tables, references

These instructions apply to everyone, regardless of the formatter being used.

5.1 Citations within the text

Citations within the text should be numbered consecutively. The corresponding number is to appear enclosed in square brackets, such as [1] or [2]-[5]. The corresponding references are to be listed in the same order at the end of the paper, in the **References** section. (Note: the standard BIBTEX style `unsrt` produces this.) As to the format of the references themselves, any style is acceptable as long as it is used consistently.

As submission is double blind, refer to your own published work in the third person. That is, use "In the previous work of Jones et al. [4]", not "In our previous work [4]". If you cite your other papers that are not widely available (e.g. a journal paper under review), use anonymous author names in the citation, e.g. an author of the form "A. Anonymous".

5.2 Footnotes

Indicate footnotes with a number[1] in the text. Place the footnotes at the bottom of the page on which they appear. Precede the footnote with a horizontal rule of 2 inches (12 picas).[2]

5.3 Figures

All artwork must be neat, clean, and legible. Lines should be dark enough for purposes of reproduction; art work should not be hand-drawn. The figure number and caption always appear after the figure. Place one line space before the figure caption, and one line space after the figure. The figure caption is lower case (except for first word and proper nouns); figures are numbered consecutively.

Make sure the figure caption does not get separated from the figure. Leave sufficient space to avoid splitting the figure and figure caption.

You may use color figures. However, it is best for the figure captions and the paper body to make sense if the paper is printed either in black/white or in color.

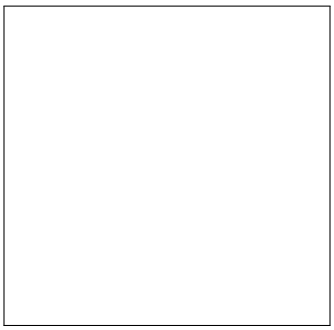

Figure 1: Sample figure caption.

5.4 Tables

All tables must be centered, neat, clean and legible. Do not use hand-drawn tables. The table number and title always appear before the table. See Table 1.

Place one line space before the table title, one line space after the table title, and one line space after the table. The table title must be lower case (except for first word and proper nouns); tables are numbered consecutively.

[1] Sample of the first footnote

[2] Sample of the second footnote

Table 1: Sample table title

PART	DESCRIPTION
Dendrite	Input terminal
Axon	Output terminal
Soma	Cell body (contains cell nucleus)

6 Final instructions

Do not change any aspects of the formatting parameters in the style files. In particular, do not modify the width or length of the rectangle the text should fit into, and do not change font sizes (except perhaps in the **References** section; see below). Please note that pages should be numbered.

7 Preparing PostScript or PDF files

Please prepare PostScript or PDF files with paper size "US Letter", and not, for example, "A4". The -t letter option on dvips will produce US Letter files.

Fonts were the main cause of problems in the past years. Your PDF file must only contain Type 1 or Embedded TrueType fonts. Here are a few instructions to achieve this.

- You can check which fonts a PDF files uses. In Acrobat Reader, select the menu Files>Document Properties>Fonts and select Show All Fonts. You can also use the program `pdffonts` which comes with `xpdf` and is available out-of-the-box on most Linux machines.
- The IEEE has recommendations for generating PDF files whose fonts are also acceptable for NIPS. Please see `http://www.emfield.org/icuwb2010/downloads/IEEE-PDF-SpecV32.pdf`
- LaTeX users:
  - Consider directly generating PDF files using `pdflatex` (especially if you are a MiKTeX user). PDF figures must be substituted for EPS figures, however.
  - Otherwise, please generate your PostScript and PDF files with the following commands:

    ```
    dvips mypaper.dvi -t letter -Ppdf -G0 -o mypaper.ps
    ps2pdf mypaper.ps mypaper.pdf
    ```

    Check that the PDF files only contains Type 1 fonts.
  - xfig "patterned" shapes are implemented with bitmap fonts. Use "solid" shapes instead.
  - The `\bbold` package almost always uses bitmap fonts. You can try the equivalent AMS Fonts with command

    ```
    \usepackage[psamsfonts]{amssymb}
    ```

    or use the following workaround for reals, natural and complex:

    ```
    \newcommand{\RR}{I\!\!R} %real numbers
    \newcommand{\Nat}{I\!\!N} %natural numbers
    \newcommand{\CC}{I\!\!\!\!C} %complex numbers
    ```
  - Sometimes the problematic fonts are used in figures included in LaTeX files. The ghostscript program `eps2eps` is the simplest way to clean such figures. For black and white figures, slightly better results can be achieved with program `potrace`.
- MSWord and Windows users (via PDF file):
  - Install the Microsoft Save as PDF Office 2007 Add-in from `http://www.microsoft.com/downloads/details.aspx?displaylang=en&familyid=4d951911-3e7e-4ae6-b059-a2e79ed87041`

  - Select "Save or Publish to PDF" from the Office or File menu
- MSWord and Mac OS X users (via PDF file):
  - From the print menu, click the PDF drop-down box, and select "Save as PDF..."
- MSWord and Windows users (via PS file):
  - To create a new printer on your computer, install the AdobePS printer driver and the Adobe Distiller PPD file from `http://www.adobe.com/support/downloads/detail.jsp?ftpID=204` *Note:* You must reboot your PC after installing the AdobePS driver for it to take effect.
  - To produce the ps file, select "Print" from the MS app, choose the installed AdobePS printer, click on "Properties", click on "Advanced."
  - Set "TrueType Font" to be "Download as Softfont"
  - Open the "PostScript Options" folder
  - Select "PostScript Output Option" to be "Optimize for Portability"
  - Select "TrueType Font Download Option" to be "Outline"
  - Select "Send PostScript Error Handler" to be "No"
  - Click "OK" three times, print your file.
  - Now, use Adobe Acrobat Distiller or ps2pdf to create a PDF file from the PS file. In Acrobat, check the option "Embed all fonts" if applicable.

If your file contains Type 3 fonts or non embedded TrueType fonts, we will ask you to fix it.

7.1 Margins in LaTeX

Most of the margin problems come from figures positioned by hand using `\special` or other commands. We suggest using the command `\includegraphics` from the graphicx package. Always specify the figure width as a multiple of the line width as in the example below using .eps graphics

```
\usepackage[dvips]{graphicx} ...
\includegraphics[width=0.8\linewidth]{myfile.eps}
```

or

```
\usepackage[pdftex]{graphicx} ...
\includegraphics[width=0.8\linewidth]{myfile.pdf}
```

for .pdf graphics. See section 4.4 in the graphics bundle documentation (`http://www.ctan.org/tex-archive/macros/latex/required/graphics/grfguide.ps`)

A number of width problems arise when LaTeX cannot properly hyphenate a line. Please give LaTeX hyphenation hints using the `\-` command.

Acknowledgments

Use unnumbered third level headings for the acknowledgments. All acknowledgments go at the end of the paper. Do not include acknowledgments in the anonymized submission, only in the final paper.

References

References follow the acknowledgments. Use unnumbered third level heading for the references. Any choice of citation style is acceptable as long as you are consistent. It is permissible to reduce the font size to 'small' (9-point) when listing the references. **Remember that this year you can use a ninth page as long as it contains *only* cited references.**

[1] Alexander, J.A. & Mozer, M.C. (1995) Template-based algorithms for connectionist rule extraction. In G. Tesauro, D. S. Touretzky and T.K. Leen (eds.), *Advances in Neural Information Processing Systems 7*, pp. 609-616. Cambridge, MA: MIT Press.

[2] Bower, J.M. & Beeman, D. (1995) *The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System.* New York: TELOS/Springer-Verlag.

[3] Hasselmo, M.E., Schnell, E. & Barkai, E. (1995) Dynamics of learning and recall at excitatory recurrent synapses and cholinergic modulation in rat hippocampal region CA3. *Journal of Neuroscience* **15**(7):5249-5262.