Detecting ELL Writing

Shayne Miel
SUNet ID: smiel
smiel@stanford.edu

Abstract

Determining whether an author is writing in their native language (L1) or a second language (L2) is a problem that lies at the intersection of three traditional NLP tasks: native language identification, similar language identification, and detecting translationese. In general, the goal of the language learner is to improve their proficiency until their writing is indistinguishable from that of a native speaker. By being able to automatically and reliably determine whether a section of text looks like L1 or L2 text, areas of writing that still need improvement can be brought to the learner's attention. Additionally, the state of the art for correcting grammatical errors involves using machine translation to translate from errorful text to corrected text[2] and there is interesting work being done in generating new training examples by using machine translation to go from error-free text to errorful text.[4] Both approaches could be enhanced by a system that can tell how close the translation is to an L1 or L2 target.

I present a supervised deep learning method for determining the probability that an essay was written by an English Language Learner (ELL) or a native English writer, using document-level labels while presenting only sliding windows of text to the model. A bidirectional LSTM acts as a convolution filter for the text and a pooling layer provides the final document-level prediction. My model improves upon a simple baseline by 0.2 AUC, but is unfortunately still picking up on confounds due to the data collection process.

1 Introduction

One of the largest challenges for non-native English students is learning to write in a way that reads like native English writing. Why blah

Examples

An automated means of providing feedback on areas of text that appear less like native English writing would be useful to students learning the language.

This paper introduces a relatively simple model blah

Section 2.1 provides examples of similar work wrt the task and section 2.2 wrt the approach. Section 3 describes the network in detail as well as several modifications that were tested. Section 4 gives the particulars of the experiment, including how the corpus was split into train/dev/test splits, the metric used, and the hyperparamters tested. Finally, section 5 analyzes the results and provides some next steps for future research.

2 Related work

2.1 Task

The task of ELL writing detection has not been studied extensively. Tomokiyo, et al. attempt this task using the TODO corpus and a TODO. [**?**] However, their work focuses mostly on text transcribed from spoken recordings, rather than writing generated by the ELL writer.

A similar task, native language identification, has received more attention. Malmasi, et al, established the ETS TOEFL-11 corpus, which has been used by a number of researchers. [5] Unfortunately for the purposes in this paper, native English writers don't tend to take the TOEFL and as such are not represented in this work. [**?**]

Classifying writing from similar language pairs (for instance TODO and TODO) is also related to this task, in that the same words are being used but the grammatical constructs are generally not followed in the same way. Gouette explores this task using a TODO . [3]

Finally, detecting translationese is also quite similar to this work in that beginning ELL students tend to conceive the structure and content of their essays in their primary language and then translate it to English when setting it down in writing. Baroni has studied this problem with TODO [1]

2.2 Model

One of the existing challenges for using RNNs and semantic vectors in general is figuring out how best to encode long documents. The approach taken in this paper is to use an LSTM as a convolution layer over sliding windows of text. This approach bears some similarities to prior work. Tang, et al. work with both a CNN and an LSTM to compose word vectors into sentence vectors, and then compose the sentence vectors into document vectors using a gated recurrent network.[**?**] Yang, et al. use a similar word-layer/sentence-layer architecture, but also provide an attention mechanism at each layer to enhance the quality of the output vectors.[**?**] The model in this paper differs from both of these in that the second level vectors contain overlapping information, due to the overlapping nature of sliding windows.

In [**?**], Mikolov describes a way to generate document vectors by adding a document token to the context of every word in the document in what is otherwise a standard word2vec model. While this approach is better than simply averaging all of the word vectors in the document, it loses important information about the order of tokens that are crucial to detecting ungrammatical English.

Collobert, et al. use a convolutional layer with a max pooling layer to generate window vectors around each word in a part of speech tagging task.[**?**] This is similar to the model presented here, except that I use an LSTM *as* the convolution.

3 Approach

3.1 Data collection

One of the reasons that there are not many papers applying deep learning to ELL related problems is the lack of a sufficiently sized corpus. In order to collect enough data to make use of a deep neural network, I had to stitch together a number of freely available research corpora, as well as some privately held corpora provided by Turnitin (turnitin.com). To protect the privacy of the students, I will refer to the data sets from Turnitin as Private A, Private B, etc. They are all essays written online by students in 6th-12th grade, as well as early college. Details of all 14 corpora are described in Table 1. The L1 abbreviations are listed in Table 2.

Table 1: ELL and Native English Corpora

CORPUS	n ESSAYS	n PROMPTS	L1s
ICNALE	5,600	2	CHN, **ENG**, FIL, HKG, IND, JPN, KOR, PAK, SIN, THA, TWN
NUCLE	1,397	3	CHN
FCE	2,481	44	CAT, CHN, FRA, GER, GRC, ITA, JPN, KOR, NL, POL, PRT, RUS, SPA, SWE, THA, TUR
CEEAUS	1,008	2	CHN, **ENG**, JPN
MOECS	199	1	**ENG**, JPN
Gachon	15,831	20	KOR
TECCL	9,864	???	CHN
Private A	550	1	KOR
Private B	4,694	4	CHN
TOEFL-11	12,100	8	ARA, CHN, FRA, GER, IND, ITA, JPN, KOR, SPA, TEL, TUR
Private C	41,227	53	**ENG**
Private D	29,559	21	**ENG**
ASAP	17,677	8	**ENG**

Table 2: L1s

ABBREVIATION	LANGUAGE				
ARA	Arabic	GRC	Greek	PRT	Portuguese
BUL	Bulgarian	HKG	Hong Kong Cantonese	RUS	Russian
CAT	Catalan	IND	Indian languages	SIN	Singapore languages
CHN	Chinese	ITA	Italian	SPA	Spanish
CZE	Czech	JPN	Japanese	SWE	Swedish
ENG	English	KOR	Korean	TEL	Telugu
FIL	Filipino	NL	Dutch	THA	Thai
FIN	Finnish	NOR	Norwegian	TSW	Tswana
FRA	French	PAK	Urdu	TUR	Turkish
GER	German	POL	Polish	TWN	Taiwanese

3.2 Data cleaning

3.3 Neural network description

3.4 Modifications

4 Experiments

train, dev, test split metric baseline abbreviations hyperparameters

dev/test table

one essay per model

5 Conclusion

5.1 Analysis

Neural net is able to beat the simple baseline provided, but it does so by picking up on confounds in the data that are unrelated to the desired output of how similar is this section of text native writing

Table 3: Sample table title

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

5.2 Future steps

Language model trained on large corpus of english

Collecting large corpus of native/non-native writing on the same topic

Collecting proficiency scores on the writing, rather than using L1 as a proxy.

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.

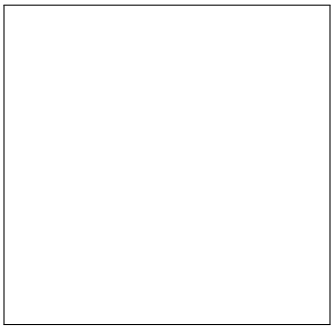

Figure 1: Sample figure caption.

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 3.

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.

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

[1] Baroni, Marco, and Silvia Bernardini. "A new approach to the study of translationese: Machine-learning the difference between original and translated text." *Literary and Linguistic Computing* 21.3 (2006): 259-274.

[2] Chollampatt, Shamil, Kaveh Taghipour, and Hwee Tou Ng. "Neural network translation models for grammatical error correction." *arXiv preprint* arXiv:1606.00189 (2016).

[3] Goutte, Cyril, et al. "Discriminating similar languages: Evaluations and explorations." *arXiv preprint* arXiv:1610.00031 (2016).

[4] Liu, Zhuoran, and Yang Liu. "Exploiting Unlabeled Data for Neural Grammatical Error Detection." *arXiv preprint* arXiv:1611.08987 (2016).

[5] Malmasi, Shervin, Joel Tetreault, and Mark Dras. "Oracle and human baselines for native language identification." *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications.* 2015.

[6] Tomokiyo, Laura Mayfield, and Rosie Jones. "You're not from 'round here, are you?: naive Bayes detection of non-native utterance text." Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies. Association for Computational Linguistics, 2001.