Consistency scores in text data [*]

Rohan Alexander *University of Toronto*
Ke-Li Chiu *University of Toronto*

In this paper we introduce a process to clean the text extracted from PDFs using various methods from natural language processing. Our approach compares originally extracted text with the text generated from, or expected by, these methods using earlier text as stimulus. To guide this process, we introduce the notion of a consistency score, which refers to the proportion of text that is unchanged by the model. This is used to monitor changes during the cleaning process, and to compare the messiness of different texts. The methods that we consider are: n-grams, [Google's Word2vec, Standford's GloVe], GPT-2, and GPT-3. We illustrate our process on text from the Canadian Hansard and introduce both a Shiny application and an R package to make our process easier for others to adopt.

Keywords: text-as-data; natural language processing; GPT-3; quantitative analysis.

Introduction

When we think of quantitative analysis, we may like to think that our job is to 'let the data speak'. But this is rarely the case in practise. Datasets can have errors, be biased, incomplete, or messy. In any case, it is the underlying statistical process of which the dataset is an artifact that is typically of interest. In order to use statistical models to understand that process, we typically need to clean and prepare the dataset in some way. This is especially the case when we work with text data. But this cleaning and preparation requires us to make many decisions. To what extent should we correct obvious errors? What about slightly-less-obvious errors? Although cleaning and preparation is a necessary step, we may be concerned about the extent to which have we introduced new errors, and the possibility that we have made decisions that have affected, or even driven, our results.

In this paper we introduce the concept of consistency in a text corpus. Consistency refers to the proportion of words that are able to be forecast by a statistical model, based on preceeding words and surrounding context. Further, we define internal consistency as when the model is trained on the corpus itself, and external consistency as when the model is trained on a more general corpus. Together, these concepts provide a guide to the cleanliness and consistency of a text dataset. This can be important when deciding whether a dataset is fit for purpose; when carrying out data cleaning and preparation tasks; and as a comparison between datasets.

To provide an example, consider the sentence, 'the cat in the...'. A child who has read this book could tell you that the next word should be 'hat'. Hence if the sentence

[*]Thank you to X, Y and Z for helpful comments. A Shiny application for interactive and small-scale examples is available: https://kelichiu.shinyapps.io/Arianna/. An R package for larger scale applications is available: **HERE**. Our code and datasets are available: https://github.com/RohanAlexander/consistency_scores_in_text_datasets. Comments on the 24 July 2020 version of this paper are welcome at: rohan.alexander@utoronto.ca.

was actually 'the cat in the bat', then that child would know something was likely wrong. The consistency score would likely be lower than if the sentence were 'the cat in the hat'. After we correct this error, the consistency score would likely increase. By examining how the consistency scores evolve in response to changes made to the text during the data preparation and cleaning stages we can better understand the effect of the changes. Including consistency scores when text corpuses are shared allows researchers to be more transparent about their corpus. And finally, the use of consistency scores allows for an increased level of automation in the cleaning process.

We consider various natural language processing methods. These include n-grams, [Google's Word2vec, Standford's GloVe], as well as GPT2 and GPT-3. The n-gram approach involves identifying two, three, or more, words that are commonly found together. Here think of a two-gram involving the word 'good' 'good morning'. At scale these can identify missing or unusual words, and work quickly, but they lack nuance. For instance an equally reasonable two-gram involving the word 'good' is 'good work'. For that reason we consider pre-trained word embedding models. These include [Google's Word2vec, Standford's GloVe], which works [representing words as dense vectors that capture their semantic relationships with each other]**SOMEHOW**. We also consider pre-trained generative models such as GPT-2 and GPT-3 from OpenAI. GPT-2 and GPT-3 are pre-trained generative unsupervised language models. They differ in the number of parameters. Generative Pretrained Transformer (GPT)-2 has X parameters and GPT-3 has more than 175 billions parameters. While GPT-2 is more limited, it can be run on smaller machines, whereas GPT-3 cannot.

We apply our approach to the Canadian Hansard. The Canadian Hansard was digitised by Beelen et al. (2017). This was a process by which PDFs were scanned, put through optical character recognition (OCR), and then corrected to a reasonable extent. The dataset is extensive, fit-for-purpose, and, appropriately, it has been used considerably, for instance Rheault and Cochrane (2020). However, there are many issues with the dataset in terms of reading it. We focus on one particular year - **(Rohan pick one)** - and show that our approach can be used to relatively quickly improve the quality of the dataset in a reproducible and consistent manner.

The remainder of our paper is structured as follows: **(Rohan add structure)**. Additionally, we construct a Shiny app available at: **(Rohan add link)**. That app computes internal and external consistency scores for corpus excerpts, and have developed an R Package that allows our approach to be used on larger datasets, which is available at: **(Rohan add link)**.

Background

PDF text extraction and n-grams

PDFs contain a lot of information, but typically that information is not able to be analysed directly by statistical models. It needs to first be extracted from the PDF, often by optical character recognition (OCR), and then cleaned, prepared and made into some type of tabular dataset.

One issue in this process is that OCR is not perfect. Hence there is a need to correct the output before it can be analysed. Traditionally, word-correction techniques evaluate errors one by one without considering the context of the surrounding words. This was no longer the case in modern correction techniques as statistical language models (SLMs) and feature-based methods have been used for context-sensitive correction (citation). Without exception, all human languages have some words that co-occur more frequently with others. Under this assumption, we can regard the production of English text as a set of conditional probabilities, written as $\Pr(w_k \mid w_1^{k-1})$, where k is the number of words in a sentence, w_k is the predicted word, and w_1^{k-1} is the history of the word occurring in a sequence (Brown et al. (1992)). In other word, the generation of prediction w_k is based on the history w_1^{k-1}. This conditional probability is the foundation of an n-gram language model.

An n-gram model is a probabilistic language model that predicts the next word in a sequence of words. The n in n-gram represents the number of words in a sequence. Taking an incomplete sentence from the Jane Eyre excerpt as an example:

“We had been wandering,”

“We” is a uni-gram, “We had” is a bi-gram, “We had been” is a tri-gram, and “We had been wandering” is a 4-gram. On the other hand, the tri-grams of the excerpt are “We had been,” and “had been wandering.” The overlapping chaining of words reveals the statistical behaviour in human language. Moreover, using n-gram models then enable us to assign a probability to the occurrence of a sequence of words or the likelihood of a word occurring next. Consider the two sentences: “We had been wandering,” and “We had been wangling.” The former is likely to be more frequently encountered in a training corpus. Thus, the n-gram would assign a higher probability to “wandering” than to “wangling.”

To predict which word appears next, we have to take the sequence of preceeding words into account, which requires knowing the probability of the sequence of words. The probability of a sequence appearing in a corpus follows the chain rule:

$$\mathrm{P}(\text{We,had,been,wandering}) = \mathrm{P}(\text{We})\cdot\mathrm{P}(\text{had} \mid \text{We})\cdot\mathrm{P}(\text{been} \mid \text{We,had})\cdot\mathrm{P}(\text{wandering} \mid \text{We,had,been})$$

However, the likelihood that more and more words will occur next to each other in an identical sequence becomes smaller and smaller, making the prediction difficult. Alternatively, we can approximate the probability of a word depending on only the previous word; this is known as Markov assumption. Instead, the Markov assumption permits us to approximate the probability using only the last n words (Brown et al. (1992)). More specifically:

$$\mathrm{P}(\text{We,had,been,wandering}) \approx \mathrm{P}(\text{We})\cdot\mathrm{P}(\text{had} \mid \text{We})\cdot\mathrm{P}(\text{been} \mid \text{had})\cdot\mathrm{P}(\text{wandering} \mid \text{been})$$

Because the bi-gram model looks only the immediate preceding word, an n-gram model under Markov assumption can then be reduced into a bi-gram model with n being any number. The general equation is:

$$\mathrm{P}(w_n|w_1^{n-1}) \approx \mathrm{P}(w_n|w_{n-N+1}^{n-1}))$$

The application of n-gram is versatile. N-gram models are widely applied in text prediction, spelling-correction, machine translation **(Keli add citation)**. In our application, we use a tri-gram based model to detect both real-word and non-word errors and correct them with the candidate word that has the highest probability. [add more about how we use n-gram once the functions in the app are in place]

Article: Class-Based n-gram Models of Natural Language (1992)

Syntax-based (grammartical) and semantic-based (sensible) word classifications in n-gram models.

Article: A Statistical Approach to Automatic OCR Error Correction in Context (1996)

Context-sensitive correction system to both non-word and real-word errors based on n-gram models applied to OCR postprocessing.

Pre-trained language models

Draw on and cite: 'Language Models are Few-Shot Learners'

In recent years, the use of pre-trained language models has played a significant role in advancing natural language processing (NLP) tasks such as reading comprehension, text generation, and even creative writing **(Ke-Li add ref)**. Pre-trained models remove the need for researchers to prepare training datasets or allot computational resources to the stage of pre-training. Because of this, there has been substantial development in NLP research.

OpenAI's GPT-3 is a generative, pre-trained, task-agnostic, language model that was announced by OpenAI in June 2020 **(Ke-Li add ref)**. OpenAI is an AI research company who openly pursues Artificial General Intelligence—to have machines learn and understand any intellectual tasks as human do **(Ke-Li add ref)**. GPT-3 is the third Generative Pre-trained Transformer language model publicly released by OpenAI. It has '175 billion parameters' and can learn NLP tasks with just a few examples. OpenAI claims that the text generation by GPT-3 is 'difficult for humans to distinguish from authentic human content' (Brown et al. (2020)). Since the API of GPT-3 at the moment is only available to a few invited users, public applications of GPT-3 are limited. On the other hand, its predecessor, GPT-2, has enabled researchers made breakthroughs in the past years. Alt et al. (2019) extended GPT-2 to relation extraction, which is a task of identifying the relationship between concepts appeared in text. They had shown a milestone in predicting larger range of distinct relation types with higher confidence **(Ke-Li add ref)**. In the field of speech recognition, data scarcity is often an issue due to the difficulty to collect large amount of data from human for learning **(Ke-Li add ref)**. Bird et al. (2020) employed GPT-2 as part of the solution to generate augmented data for classification. GPT-2 also helped NLP researchers tackling the task of textual paraphrasing **(Ke-Li add ref)**. Hegde and Patil (2020) used GPT-2 to build an unsupervised paraphrasing model that is not restricted by domain shift within an article, which is a problem faced by the usual super-

vised models of paraphrasing.
(Ke-Li please find papers that use it, describe them, and add refs - look on arXiv).

GPT-3 is a Transformer-based model that has over 175 billions of parameters (Brown et al. (2020)). Proposed by Vaswani et al. (2017), the Transformer is a novel network architecture that outperforms RNN-based models in computational efficiency **(Keli add citation)**. The parameters refer to the weights of the connections that enable the neural network to learn **(Keli add citation)**. In the past few years, the growth of parameters of transformer language models expands rapidly. The first generation GPT has 110 million parameters. Released in 2018, GPT-2 has 1.6 billion parameters, and within two years, GPT-3 has grown the size to 175 billion, which is over 100 times than its predecessor . The more parameters there are the better the model can generalize to new tasks; therefore, the increasing amount of parameters accelerates the improvements in text synthesis and downstream NLP tasks **(Keli add citation)**. GPT-3 can learn NLP tasks with just a handful of example; OpenAI refers this ability as few-shot learning, or in-context learning. Few-shot learning means that the model **(Keli add citation)**. As a result, we only need to provide GPT-3 a few demonstrations for it to learn a new NLP task. **(Ke-Li add more detail about what it needs and what it outputs and how it works).**. For example, OpenAI team trained GPT-3 to generate "news articles". The prompt input is a plausible first sentence of a news story written by human, and the output is an entire news article generated by the model itself. Without the demonstrations, GPT-3 tends to treat the prompts as "tweets" and generate texts that serve as responses or follow-up tweets. The team fed a few previous news articles in the model's context as demonstrations, and the model learned to produce the outputs that adapt to the style and length of news stories (Brown et al. (2020)). This few-shot learning ability indicates that the model can learn like human do and would democratize the use of language models to broader fields. **(Ke-Li please find the language that they use for this and then we need to make ours consistent with theirs).**

Stylized example

(TODO: This stylized example need to be changed and updated to actually use GPT-3 at some point.)

In our application, we deploy OpenAI GPT-3 to generate text and . . .

As a stylized example, let's consider the following actual paragraph from Jane Eyre, by Charlotte Bronte:

> There was no possibility of taking a walk that day. We had been wandering, indeed, in the leafless shrubbery an hour in the morning; but since dinner (Mrs. Reed, when there was no company, dined early) the cold winter wind had brought with it clouds so sombre, and a rain so penetrating, that further out-door exercise was now out of the question.

Let's pretend that this text had been created from optical character recognition and that it had the following errors: some 'h' were replaced with 'b'; and some 'd' have been replaced with 'al':

> There was no possibility of taking a walk that day. We had been wandering, indeed, in tbe leafless shrubbery an hour in tbe morning; but since dinner (Mrs. Reeal, when tbere was no company, dined early) the cold winter wind had brought with it clouds so sombre, and a rain so penetrating, tbat further out-door exercise was now out of the question.

Assume a model that is trained to perfectly forecast the next word in Jane Eyre. For this fragment there are 62 words, comprising 5 errors and 57 correct words. So the internal consistency score of this fragment would be: 57/62 = 0.919. When we recognise and correct the errors, this consistency score would increase to 1.

Similarly, assume a model that is trained on an external data source. This means that it will recognise the the cases where some 'h' were replaced with 'b', but not recognise that 'Reed' has become 'Reeal'. Hence, the external consistency score would be 58/62 = 0.935.

Data

Various data can be used. . . .

Application

Discussion

Internal validity vs external- one is their own words the other is a general set of words.

In the same way that precision and recall provide important measures. . .

Appendix

Literature summaries

(These are just here for now - we'll remove later.)

Article: Improving Language Understanding by Generative Pre-Training (2018)

Introduction of a semi-supervised approach for language understanding machine learning tasks. The approach is the combinaiton of unsupervised pre-training and supervised fine-tuning.

Article: Language Models are Few-Shot Learners (2020)

Official paper from OpenAI to introduce OpenAI GPT-3. Emphasizes on the "fewer-shots" and "task-agnostic" aspects of the latest langual model compared to its predecessor GPT-2.

Discussion of how GPT-3 works

(Ke-Li - it would be good for us to go through how it actually works in as much detail as we can manage (in our own words). This is mostly just to force us to learn how it works and may be taken out of the final version of the paper, but it's still an important exercise.)

References

Alt, C., Hübner, M., and Hennig, L. (2019). Fine-tuning pre-trained transformer language models to distantly supervised relation extraction. *arXiv preprint arXiv:1906.08646*.

Beelen, K., Thijm, T. A., Cochrane, C., Halvemaan, K., Hirst, G., Kimmins, M., Lijbrink, S., Marx, M., Naderi, N., Rheault, L., et al. (2017). Digitization of the canadian parliamentary debates. *Canadian Journal of Political Science/Revue canadienne de science politique*, 50(3):849–864.

Bird, J. J., Faria, D. R., Ekárt, A., Premebida, C., and Ayrosa, P. P. (2020). Lstm and gpt-2 synthetic speech transfer learning for speaker recognition to overcome data scarcity. *arXiv preprint arXiv:2007.00659*.

Brown, P. F., Della Pietra, V. J., Desouza, P. V., Lai, J. C., and Mercer, R. L. (1992). Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–480.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Hegde, C. and Patil, S. (2020). Unsupervised paraphrase generation using pre-trained language models. *arXiv preprint arXiv:2006.05477*.

Rheault, L. and Cochrane, C. (2020). Word embeddings for the analysis of ideological placement in parliamentary corpora. *Political Analysis*, 28(1):112–133.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.