An NLP-based automated editorial aid

ABSTRACT

This article discusses ProWritingAid - an automated editorial aid, that improves English manuscript spelling, grammar and readability using machine learning, handcrafted rules, and n-gram models. It provides client applications and plugins for major operating systems and editing software. ProWritingAid generates a series of reports and actionable text correction suggestions aiming to improve writing for professional and academic writers, students, second language learners.

1 INTRODUCTION

Writing good text is both art and science. It should be concise, informative and easy to understand. In professional and scientific publishing this involves significant amount of editorial work. An editor directs the focus of the manuscript, cuts out nonessential text, verifies facts. He checks the correct usage of language – not just grammar, but also style and ease of perception. Automated editorial aid tools (ETs) can provide cost and time savings by simplifying editorial work and moving more of it to authors. Moreover, ETs are beneficial for students and language learners - over a billion people are using or learning English as a second or foreign language [1].

Major problems ETs try to solve are (1) spelling, (2) grammar (e.g. punctuation, word agreement) correction [2,3] and (3) improving text readability. There are many difficulties in all of these tasks. For example, a naïve approach to spelling correction might look up each manuscript word in a dictionary and flag the unknown ones. However, there may be many candidate corrections for a given string. Consider the string *ater*. Without context, we can equally likely substitute it with *after, later, alter, water*. Furthermore, such dictionary is bound to lack neologisms and domain-specific terminology. Finally, correct spelling and usage of a word may depend on context. E.g. the phrase *see you in five minuets* could be correct or not, depending on whether the author uses minutes or dances to measure time. [4]

English text readability has a long history of assessment. Some of the well-known metrics include Flesch-Kinkaid Score and the Dale-Chall Formula [5, 6]. However, these metrics are insufficient. While they can be used for recommending the user to reduce average syllable count, such suggestions are too broad and may be unsuitable for some kinds of text. Providing specific and actionable recommendations requires detecting clichés and inappropriate slang, pinpointing overly complex grammatical constructs among many other tasks.

A considerable amount of research exists on the different aspects of text correction. Approaches progressed from simple dictionary lookups to contextual spelling techniques analyzing adjacent words when generating suggestions. Popular methods for correcting grammar errors include machine learning classifiers, n-gram models, statistical machine translation models and usage of handcrafted rules [2]. Availability of publicly accessible datasets, such as Google N-gram corpus [7], served as a major enabler for using statistical and machine learning methods in this domain.

In this paper, we introduce ProWritingAid [8] – an ET that addresses the following areas:

1. Grammar and spelling error correction;
2. Readability improvement;
3. Cross-platform text editing via standalone applications and integration with popular third-party software.

By combining diverse NLP techniques into an integrated platform, ProWritingAid (PWA) is aiming to simplify work for writers, publishers, academics and English learners.

2 System description

2.1 Architecture

Figure 1 shows the high-level PWA architecture. The core of PWA is the text analysis engine, including multiple modules, such as spelling correction.

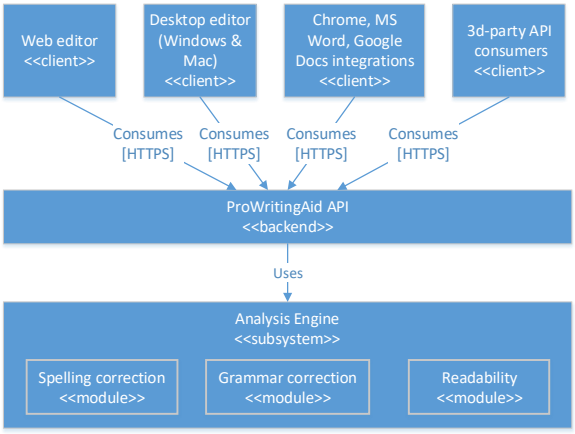

Figure 1: High-level ProWritingAid architecture

These modules are consumed by the backend service to provide a unified API for client applications. Finally, there are multiple client applications, including standalone clients and plugins for Google Chrome, Google Docs and Microsoft Word. The system also supports API access from third-party applications using HTTPS and JSON.

Our text analysis engine is built using the principles of the Unstructured Information Management Architecture (Apache UIMA) [9]. The key concepts of this architecture are:

1. **Annotator** – used to perfume a particular type of analysis on the text, such as Part-of-Speech (POS) tagging;
2. **Annotation** – The output of an annotator, such as a particular POS tag on a certain token;

3. **Common Analysis Structure (CAS)** – allows sharing and representation of annotator results.

Relations between these concepts are shown in Figure 2.

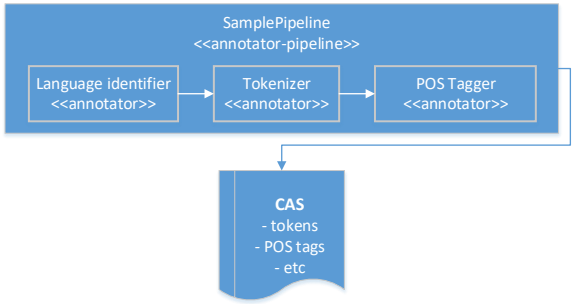

Figure 2: Key UIMA architectural concepts

This architecture allows easy analysis customization and algorithm prototyping because of the modularity and composability.

2.2 Text correction

The general approach to text correction used by PWA is as follows:

1. Generate the list of candidate corrections;
2. Filter corrections using manually created anti-false positive (anti-FP) patterns;
3. Confirm the likelihood of candidate corrections with the Google N-gram model [7], looking at the surrounding context.

We use a mixture of handcrafted rules and machine learning to generate candidate corrections. Where training data is available or can be easily created, machine learning is preferred. However, assembling a reliable learning corpus may sometimes be too laborious. Often manual rules are much faster to create and perform extremely well. Notably, a similar approach was used by winners of the CoNLL-2014 shared task [10].

When applying machine learning, we train individual models for specific error types (such as subject-verb agreement) using Conditional Random Fields (CRFs), Support Vector Machines (SVMs), confusion sets and other techniques.

For readability improvement, we rely on a set of manually created rules, heuristics and dictionaries to highlight clichés, overuse of glue words, inconsistent punctuation, excessive word repetition, and other issues. The concept of readability is domain-specific. For instance, colloquialisms like *raining cats and dogs* are acceptable in a novel, but not in a maths article. Therefore, we adapt rule sets to the user-defined writing style.

2.3 Reporting analysis results

Our web and desktop applications provide the user with a summary report and highlight individual issues in the text.

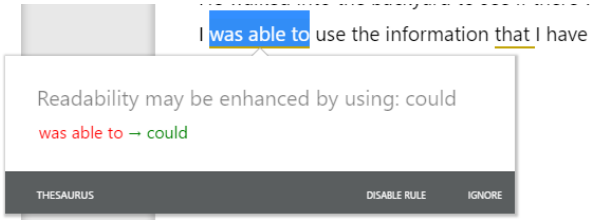

Figure 3: Fragment of analysis overview report

The summary report contains key readability metrics, top suggestions and result comparison to other PWA users. Apart from actionable recommendations, this provides users with a learning opportunity, because many suggestions contain guidance on correct word usage.

3 CONCLUSIONS

In this paper, we presented ProWritingAid - an ET that improves spelling, grammar and readability using machine learning, handcrafted rules, and n-gram models. PWA provides a standalone and a web editor, plugins for popular editing software, and an API for third parties. It presents text analysis results as a series of reports and text annotations. By including guidance on correct word construct usage into the generated annotations, PWA aims to provide the users with a learning opportunity.

An interesting direction for future research is exploring more general approaches to text correction using such deep learning techniques as Long Short-Term Memory (LSTM) neural networks.

ACKNOWLEDGMENTS

This work was made possible by Chris Banks, the creator and director of PWA; Viktor Pekar, the chief NLP engineer; and the rest of the PWA team who put their effort into building the system.

REFERENCES

[1] Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel R. Tetreault. 2014. Automated Grammatical Error Detection for Language Learners: Second Edition.

[2] Alla Rozovskaya and Dan Roth. 2014. Building a state-of-the-art grammatical error correction system. In *Transactions of the Association for Computational Linguistics*, 2(10):419–434

[3] Alla Rozovskaya and Dan Roth. 2014. Joint Learning and Inference for Grammatical Error Correction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*

[4] Karen Kukich. 1992. Techniques for Automatically Correcting Words in Text. In ACM Computing Surveys, Vol 24, No 4.

[5] Kincaid, J.P., Fishburne, R.P., Rogers, R.L., & Chissom, B.S. (1975). Derivation of new readability formulas (automated readability index, fog count, and Flesch reading ease formula) for Navy enlisted personnel. Research Branch Report 8–75. Chief of Naval Technical Training: Naval Air Station Memphis.

[6] Dale E, Chall J. 1948. "A Formula for Predicting Readability". Educational Research Bulletin. 27: 11–20+28.

[7] Google N-gram dataset. 2012. http://storage.googleapis.com/books/ngrams/books/datasetsv2.html. Accessed: 24/6/2017

[8] ProWritingAid. 2017. https://prowritingaid.com. Accessed:24/6/2017.

[9] Apache UIMA Overview. https://uima.apache.org/d/uimaj-2.4.0/overview_and_setup.html#ugr.ovv.conceptual.architecture_framework_sdk. Accessed: 24/06/2017

[10] Mariano Felice, Zheng Yuan, Øistein E. Andersen, Helen Yannakoudakis, and Ekaterina Kochmar. 2014. Grammatical error correction using hybrid systems and type filtering. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task.*