October 2020

Automatic Generation of Training Corpus for Natural Language Processing Tasks

Shruti Gupta

Anmol Gulati

Jayakumar Hoskere

Automatic Generation of Training Corpus for Natural Language Processing Tasks

ABSTRACT

Machine learning models that perform grammar error correction (GEC) suffer from insufficient training data. This disclosure describes techniques that automatically generate a large corpus of training data for GEC and other natural language processing tasks. With specific user permission, the techniques leverage the edit histories of documents by identifying changes to documents attributable to grammatical corrections by users. The training set for the GEC machine learning model is automatically augmented by sentences known to be ungrammatical (e.g., original text, before revision by user) or grammatical (e.g., text after revision by user), and labeled as such. The techniques enable the provision of a very large corpus of training data for grammar error-correcting or other natural language processing ML models.

KEYWORDS

- Grammar error correction (GEC)
- Natural language processing (NLP)
- Training set
- Training corpus
- Corpus curation
- Revision history
- Edit history
- Grammaticality filtering
- Machine learning model

BACKGROUND

Grammar error-correcting (GEC) machine translation models generally require millions of pairs of training data of the form (original ungrammatical-segment, revised grammatical-segment). Various sources of such training data are available, such as:

- High-quality academic datasets, manually annotated by linguists and language experts. These datasets number about 10,000 to 100,000 sentences.
- GEC websites comprising semi-manually annotated datasets of moderate quality. These number about two-to-ten million sentences.
- Edit histories of online, crowd-sourced encyclopedias, which number about two hundred million and are mostly noisy.

It is thus seen that currently available training datasets are either too small or of low quality. The problems of training grammar error correction (GEC) machine learning models and related natural language processing tasks thus suffer from insufficient, large-scale training data.

Online document editors typically store documents as a time sequence of edits. The edits, e.g., revision history, can be surfaced to users to enable them to revert to or look up a previous version. Documents and their edit histories potentially represent training example pairs numbering in the billions. Moreover, data from documents and their edit-histories are domain-wise more suited as these likely capture natural mistakes made by everyday users. As opposed to, e.g., online, crowd-sourced encyclopedias, which tend to have a formal style, third-person narrative, and extensive use of past tense, online user-generated documents encompass a much wider variety of writing.

Attempts have been made to synthesize sources like revision histories, where users revise the original content to a more polished version by rectifying mistakes that include grammar

errors. However, existing techniques do not effectively mine training data relevant to the NLP task, e.g., GEC; consequently, such techniques are unable to extract a useful training corpus from noisy raw content.

DESCRIPTION

This disclosure describes techniques that, with specific user permission, leverage the recorded edit history of online documents to auto-generate a training corpus for GEC machine learning models. Grammatical changes are filtered out of the edit history sequence of documents, which comprise (original-text, revised-text) pairs, to build a training corpus comprising likely (original-ungrammatical-text, revised-grammatical-text) pairs. This corpus is used to train GEC machine translation models.

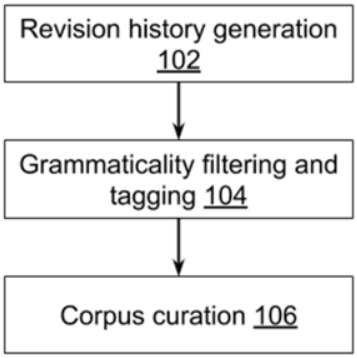

Fig. 1: Automatic generation of training corpus for GEC machine learning models

Fig. 1 illustrates automatic generation of training corpus for grammar error-correcting machine learning models, per techniques of this disclosure. The corpus is generated in three steps, explained below in greater detail.

Revision history generation (102)

As mentioned before, online document editors store documents in the form of a time sequence of edits made potentially by multiple users in a collaborative setup. Per the techniques

of this disclosure, a revision history for the document is generated by starting from an empty document and replaying edits one by one to create document versions at different points in time. In a multi-user editing scenario, parallel edits are first processed to obtain the net change. For example, for the word "cat," if userA deletes 'c' from the beginning and userB inserts 'e' at the end, the net resulting change is "cat" → "ate".

Each edit tracks its position in the document and the actual change made. These details are used to maintain a history of changes made to a particular segment at any level of granularity. After processing the edits, a sequence of segment versions (version 1 → version 2 → … → version N → final version) is obtained.

For the task of GEC, version histories are considered at a sentence level. To isolate edits that are made to rectify errors, changes made while generating or framing a sentence are ignored. For example, if a sentence construction and revision happens thus: "This" → "This is" → "This is cat" → "This is a cat", only the revision "This is cat" → "This is a cat" is tracked.

Eventually, revision history generation results in pairs of sentences of the form (version X → version Y) where version Y is a revision of version X.

Grammaticality filtering and tagging (104)

The (version X → version Y) sentence revision pairs can capture a variety of changes, only a fraction of which are changes made to correct grammar. For the task of GEC, (version X, version Y) pairs that capture grammar error corrections are relevant, e.g., the original sentence (version X) is ungrammatical while the revised sentence (version Y) is grammatically correct. The revised sentence can still have grammar errors but is an improvement over the prior version. Herein are disclosed the following techniques to filter grammatical error corrections from the set of sentence-revision pairs.

- Grammaticality filtering using edit tagging via syntax annotations, and
- Grammaticality filtering using machine learning.

Each of these techniques is described in greater detail below.

Grammaticality filtering using edit tagging via syntax annotations

Edits that transform a sentence from original to revised are classified into, e.g., tagged with, a known edit type using rule-based heuristics. If the edit tag belongs to a known grammar-edit category, e.g., adding an article, changing verb form, changing the number of a noun, correcting a spelling, etc., it is retained as a grammatical edit. If the edit tag belongs to a semantic edit or rephrasing category, e.g., changing a proper noun, changing sentence structure, etc., then it is not considered a grammatical edit. Edit tagging not only serves the purpose of filtering sentence pairs for grammaticality but also creates a diverse and representative training corpus by having an appropriate representation of different edit types.

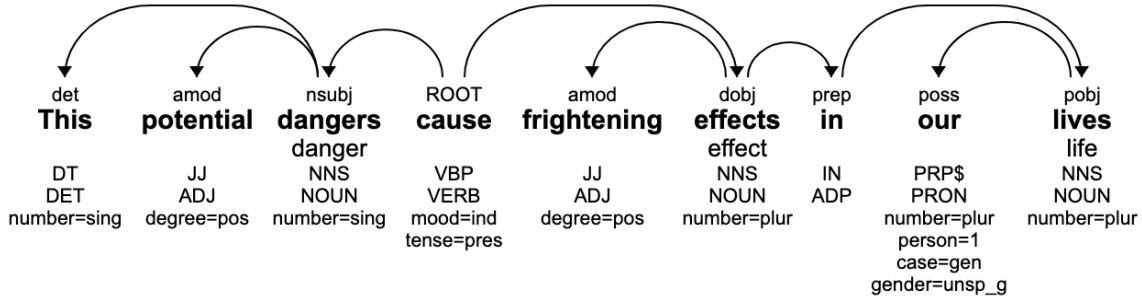

Fig. 2a: Dependency parse tree for the original sentence

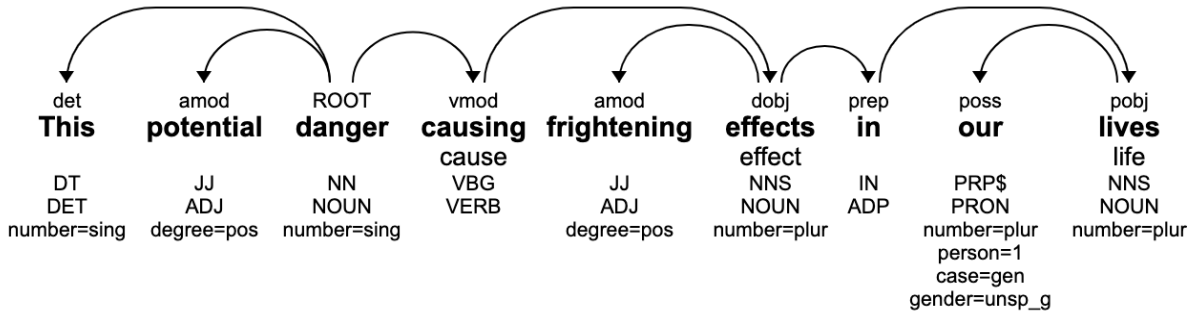

Fig. 2b: Dependency parse tree for the revised sentence

Grammatical edits are detected based on changes in syntactic annotations, e.g., part-of-speech tags, dependency parse tree labels, morphology, etc., and relationships between a word and its surrounding words/phrases. This is illustrated in Fig. 2, which shows the dependency parse tree for the original (Fig. 2a) and the revised (Fig. 2b) sentences along with syntactic annotations. Parts of speech are identified in coarse and fine forms. Coarse-form categories filter data for grammaticality, while fine-form categories build training-corpus diversity.

Examples of coarse-form parts of speech include, e.g., DET for determiner (this, that, etc.); NOUN for noun; ADJ for adjective; ADP for preposition; etc. Examples of fine-form parts of speech include NNS for plural noun; JJ for a specific type of adjective; VBP for the present form of a verb; VBG for the continuous form of a verb; etc. The nodes of the dependency parse tree are labeled. For example, as shown in Fig. 2a, the dependency parse tree is rooted on the word ‘cause,’ which is labeled as ROOT, and the children of ROOT are the subject noun (nsubj, ‘dangers’) and the object noun (dobj, ‘effects’).

As illustrated in Fig. 2b, the sentence revision “This potential dangers cause frightening effects in our lives” → “This potential danger causing frightening effects in our lives” has two edits: dangers → danger, and cause → causing. Considering the differences in syntactic annotations between the original (Fig. 2a) and revised (Fig. 2b) sentences, the NNS → NN change in part-of-speech tag indicates that the noun ‘danger’ has changed number, and the VBP → VBG change indicates that the verb ‘cause’ has changed tense.

Grammaticality filtering using machine learning

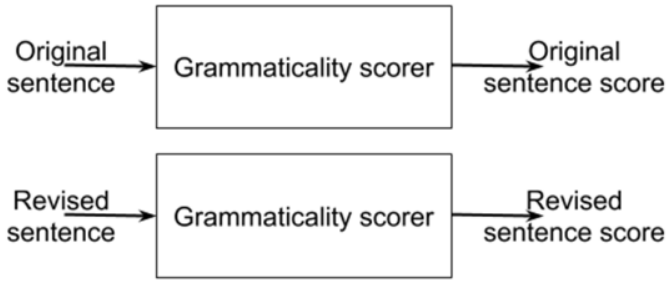

Fig. 3: Grammaticality filtering via machine learning

To filter grammatical error corrections from the set of sentence-revision pairs using machine learning, the original sentence and the revised sentence are input to a grammaticality scorer, which can be a deep-neural machine learning model that tests for grammaticality. The grammaticality scorer produces a grammar score for the original and the revised sentences. If the revised sentence score is substantially higher than the original sentence score, then the (original, revised) pair constitutes a grammar error correction.

Corpus curation via sampling (106)

Examples of grammatical errors that result from grammaticality filtering of document revision histories may not be directly usable for training GEC models. One reason for the lack of direct usability is that the distribution of error types in revision histories may be biased by a multitude of examples capturing the most common mistakes made by users. For example, omitting the determiner “the” before nouns is a common mistake made by ESL (English as a second language) writers. A corpus from revision history may be biased by overabundance of “the”-insertion corrections. Training GEC models on such a set can cause the models to learn the bias.

Corpus curation is a technique to balance the distribution of error types such that they match error distributions of standard datasets created by linguists. By curating the corpus, the GEC model learns not only the most common errors but also error patterns that may otherwise be overshadowed by an overabundant supply of common errors. Corpus curation results in an improved training corpus.

The corpus is curated through a data sampling procedure. In particular, stratified random sampling is used to divide the (original, revised) sentence pairs into buckets based on type of grammatical error and to randomly select a certain number of examples from each of those buckets. The sampling probability, e.g., the probability of including an example from a particular bucket in the final corpus, is matched to the error distributions of standard training corpora. For GEC tasks, as mentioned before, there exist a few standard corpora in academia, but these are limited in size (~0.5M). By sampling the revision history data in proportions similar to a known corpus, a very large corpus (~100M) is generated that is similar to the known corpus.

In this manner, the techniques of this disclosure leverage the known edit history of documents recorded by online document editors to generate a very large training corpus for grammar error-correcting ML models. The use of online documents enables capturing a diverse set of errors made naturally by users. The techniques of grammaticality filtering and corpus curation via sampling, as described herein, make it viable to mine noisy datasets, e.g., edit histories of online documents, to arrive at a meaningful training corpus.

Further to the descriptions above, a user may be provided with controls allowing the user to make an election as to both if and when systems, programs or features described herein may enable collection of user information (e.g., information about a user's documents, a user's social network, social actions or activities, profession, a user's preferences, or a user's current

location), and if the user is sent content or communications from a server. In addition, certain data may be treated in one or more ways before it is stored or used, so that personally identifiable information is removed. For example, a user's identity may be treated so that no personally identifiable information can be determined for the user, or a user's geographic location may be generalized where location information is obtained (such as to a city, ZIP code, or state level), so that a particular location of a user cannot be determined. Thus, the user may have control over what information is collected about the user, how that information is used, and what information is provided to the user.

CONCLUSION

This disclosure describes techniques that automatically generate a large corpus of training data for GEC and other natural language processing tasks. With specific user permission, the techniques leverage the edit histories of documents by identifying changes to documents attributable to grammatical corrections by users. The training set for the GEC machine learning model is automatically augmented by sentences known to be ungrammatical (e.g., original text, before revision by user) or grammatical (e.g., text after revision by user), and labeled as such. The techniques enable the provision of a very large corpus of training data for grammar error-correcting or other natural language processing ML models.

REFERENCES

[1] Lichtarge, Jared, Chris Alberti, Shankar Kumar, Noam Shazeer, Niki Parmar, and Simon Tong. "Corpora generation for grammatical error correction." *arXiv preprint arXiv:1904.05780* (2019).

[2] Grundkiewicz, Roman, and Marcin Junczys-Dowmunt. "The WikEd error corpus: A corpus of corrective Wikipedia edits and its application to grammatical error correction." In

International Conference on Natural Language Processing, pp. 478-490. Springer, Cham, 2014. https://emjotde.github.io/publications/pdf/mjd.poltal2014.draft.pdf accessed on Feb. 17, 2020.

[3] "Syntax" https://en.wikipedia.org/wiki/Syntax accessed on Feb. 17, 2010.

[4] "Part of speech" https://en.wikipedia.org/wiki/Part_of_speech accessed on Feb. 17, 2010.

[5] "Dependency-based parse trees" https://en.wikipedia.org/wiki/Parse_tree#Dependency-based_parse_trees accessed on Feb. 17, 2010.

[6] "Morphology (linguistics)" https://en.wikipedia.org/wiki/Morphology_(linguistics) accessed on Feb. 17, 2010.

[7] "Sampling (statistics)" https://en.wikipedia.org/wiki/Sampling_(statistics) accessed on Feb. 17, 2010.

[8] "Stratified sampling" https://en.wikipedia.org/wiki/Stratified_Sampling accessed on Feb. 17, 2010.

[9] "Sampling probability" https://en.m.wikipedia.org/wiki/Sampling_probability accessed on Feb. 17, 2010.

[10] Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).