

GEC Review

Nuri Tas

August 2023

1 Grammatical Error Correction

Grammatical error correction (GEC) aims to systematize writing errors to correct grammar and spelling errors. However, one of the biggest bottlenecks in GEC is data sparsity [Google], namely the need for more diverse and large exemplary data. Although there have been many attempts to generate synthetic data based on various models, such as heuristic-based random words, such models failed to represent the proper distribution of grammatical errors in the wild [Google]. We will explain tagged corruption models by [stahlberg-kumar-2021-synthetic] that enables a more flexible distribution of synthetic data and produces state-of-the-art results on GEC baselines.

1.1 Tagged Corruption Models

This section summarizes the paper [stahlberg-kumar-2021-synthetic]

Tagged corruption models create an ungrammatical (corrupt) sentence from a clean sentence based on an error tag $\tau \in T$, where T denotes 25 error tags supported by ERRANT.

The model is trained using Seq2Edits [stahlberg-kumar-2020-seq2edits], which can predict the error tags together with the edits. Additionally, Finite State Transducers (FST) are utilized to force to generate a particular error tag. The following figure shows the FSTs for the *SPELL* tag.

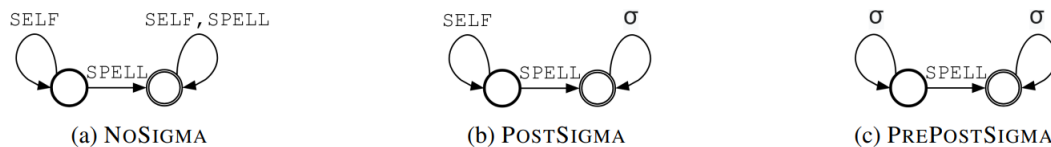


Figure 1: Image credit: [stahlberg-kumar-2021-synthetic]

In Fig 1, σ refers to any error tag and *SELF* denotes the source spans that are not modified, i.e., the change points where corruption is applied. In Fig 1a,

only includes *SELF* and *SPELL*, whereas Fig 1b can have any error tag after starting with *SELF* and *SPELL* tags. Fig 1c, however, can start and end with any error tags as long as including a *SPELL* tag in between.

1.1.1 Synthetic Data Generation with Tagged Corruption Models

Let x_n ($n \in [1, N]$) and $y_{t,n}$ denote the input sentence and the corresponding corrupted sentence for a given tag $\tau \in T$, respectively. One of the goals is to only apply one tag per sentence and create a new corrupted corpus from the input data such that the distribution of the error tags are equal:

$$\forall \tau \in T : P(\tau) \approx = \frac{|\{\tau_n = \tau | n \in [1, N]\}|}{N} \quad (1)$$

Three methods are compared: Offline-Optimal, Offline-Probabilistic, Online

Offline-Optimal The offline-optimal model searches for the tag with the highest log-probability under the constraint that the distribution of the tags matches the desired distribution:

$$\max_{\tau^*} \sum_{n=1}^N \log P(y_{\tau_n^*, n} | \tau_n^*, x_n) \quad (2)$$

It can be shown that this model can be solved with the minimum-cost solver.

Offline-Probabilistic In the offline-probabilistic model, the samples are created according to a given distribution, and then N sentences with the highest likelihood to contain a given tag is drawn:

$$P((x, y) | \tau) = \frac{P(\tau | (x, y)) P(x, y)}{\sum_{n=1}^N P(\tau | (x_n, y_n)) P(x_n, y_n)} \quad (3)$$

assuming each corrupted sentence sample for a given tag has equal probability, i.e., $P(x, y) = \frac{1}{N}$, we obtain

$$P((x, y) | \tau) = \frac{P(\tau | (x, y))}{\sum_{n=1}^N P(\tau | (x_n, y_n))} \quad (4)$$

Now,

$$P(\tau | (x, y)) = \frac{P(\tau, x, y)}{P(x, y)} = \frac{P(x) P(\tau | x) P(y | (\tau, x))}{P(x, y)} = \frac{\frac{1}{N} \frac{1}{|T|} P(y | (\tau, x))}{\frac{1}{N}} = \frac{1}{|T|} P(y | (\tau, x)) \quad (5)$$

Note that, since we draw the corrupted sentences with the highest probability to contain a given tag, the model may include corrupted sentences derived from the same sentence or exclude some sentences in the input data. This is not an issue in the offline-optimal model as the model already takes each input sentence as a supply node (in the minimum-cost problem terminology) and derive the optimal tag accordingly.

Online Both offline-optimal and offline-probabilistic models have to create $N * |T|$ corrupted sentences to find the optimal solutions, and hence each have $\Omega(N|T|)$ time-complexity. This issue can be optimized by assigning tags on the fly according to a given distribution and creating a model with $\Omega(N)$ time complexity.

1.2 ERRANT for Turkish

As a first attempt, we give corresponding grammatical errors according to ERRANT error tags. Note that some tags don't have a Turkish example, i.e., CONTR (n't \rightarrow not) or DET (the \rightarrow a). The following table is taken from [UCAM-CL-TR-938] and adjusted for Turkish.

References

- [UCAM-CL-TR-938] Christopher Bryant. *Automatic annotation of error types for grammatical error correction*. Tech. rep. UCAM-CL-TR-938. University of Cambridge, Computer Laboratory, June 2019. DOI: 10.48456/tr-938. URL: <https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-938.pdf>.
- [stahlberg-kumar-2020-seq2edits] Felix Stahlberg and Shankar Kumar. "Seq2Edits: Sequence Transduction Using Span-level Edit Operations". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 5147–5159. DOI: 10.18653/v1/2020.emnlp-main.418. URL: <https://aclanthology.org/2020.emnlp-main.418>.
- [stahlberg-kumar-2021-synthetic] Felix Stahlberg and Shankar Kumar. "Synthetic Data Generation for Grammatical Error Correction with Tagged Corruption Models". In: *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*. Online: Association for Computational Linguistics, Apr. 2021, pp. 37–47. URL: <https://aclanthology.org/2021.bea-1.4>.

[Google]

Felix Stahlberg and Shankar Kumar. *The C4_200M Synthetic Dataset for Grammatical Error Correction*. URL: <https://ai.googleblog.com/2021/08/the-c4200m-synthetic-dataset-for.html>. (accessed: 08.07.2023).

| Code | Meaning | Description\Example |
|------------|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ADJ | Adjective | büyük adam oldun → koca adam oldun |
| ADJ:FORM | Adjective Form | → Comparative or superlative adjective errors. Since the corresponding Turkish comparative and superlative adjectives (daha/en) are strict, it is unlikely to encounter errors here. |
| ADV | Adverb | çabukca → çabucak |
| CONJ | Conjunction | ve → ama |
| CONTR | Contraction | - |
| DET | Determiner | - |
| MORPH | Morphology | This can be tricky in Turkish. For instance, both hızlı (adj) and hızlıca (adv) can be used interchangeably: hızlı geldin ~ hızlıca geldin |
| NOUN | Noun | kafam ağrıyor → başım ağrıyor |
| NOUN:INFL | Noun Inflection | We adjust the original definition to the correct use of "ler" or "lar" for plural words. Defterlar → defterler |
| NOUN:NUM | Noun Number | iki kişiydi → iki kişilerdi |
| NOUN:POSS | Noun Possessive | - |
| ORTH | Orthography | birşey → bir şey |
| OTHER | Other | Unclassified errors; e.g. paraphrases. |
| PART | Particle | - |
| PREP | Preposition | sen de kalalım → sende kalalım |
| PRON | Pronoun | bizimki çocuklar → bizim çocuklar |
| PUNCT | Punctuation | ; → , |
| SPELL | Spelling | entellektüel → entelektüel |
| UNK | Unknown | Detected but not corrected errors |
| VERB | Verb | geldim → gittim |
| VERB:FORM | Verb Form | gitme için bekledik → gitmek için bekledik |
| VERB:INFL | Verb Inflection | Misapplication of tense morphology. geldem → geldim |
| VERB:SVA | Subject-Verb Agreement | (Ben) geldin → (Ben) geldim |
| VERB:TENSE | Verb Tense | |
| WO | Word Order | hemen çocuk geldi → çocuk hemen geldi. Note: Turkish is more malleable against the word order. |