Specifically, they had to select those errors which in their opinion are most common in Basque texts regardless of the register and domain.

In addition to their own judgment, these experts used as additional material the list of most common errors in the exams for the EGA title (*Euskararen Gaitasun Agiria*, or in English: Certificate of Proficiency in Basque).

From the intersection of the two sets of ten errors selected by both experts, six errors resulted (Table 1), of which two (erroneous use of suffixes in dates and times) were discarded because they are easily solved by rule-based techniques (Oronoz, 2009). Thus, the four errors selected for this paper are the following:

- E1: Wrong use of the verb tense or aspect. For example, the use of the verbal form of the present in a future context.
- E2: Misuse of the verbal paradigm. The verbal system in Basque consists of four paradigms: *nor* (monovalent intransitive), *nor-nork* (bivalent transitive), *nor-nori* (bivalent intransitive), *nor-nori-nork* (trivalent transitive). Each verb is conjugated according to its corresponding paradigm(s). It is a very common error to use the wrong paradigm.
- E3: Lack of concordance between the verb and the subject. Confusion of the declension suffix in the subject.
- E4: Misuse of the verbal suffix. Completive sentences in Basque are formed by adding the suffix (*-(e)la*) to the verb of the subordinate sentence. If the sentence is negative, the suffix should be *-(e)nik*.

| *Error-type* | *Examples* |
|---|---|
| E1: Verb tense | *Ziur bihar **jakiten*** (→jakingo) *dugula. (I'm sure we'll find out tomorrow)*<br>*Gustura egingo **nuen*** (→nuke) *orain. (I'd love to do it now)*<br>*Gauza bat **faltatzen*** *(*→falta*) zait esateko. (There's one more thing I have to say)* |
| E2: Verbal paradigm | *Afaltzera gonbidatu **zidan*** (→ninduen). *(He/She invited me to dinner)*<br>*Atzo kalean ikusi **nizun*** (→zintudan). *(I saw you yesterday on the street)*<br>*Utzi behar **dugu*** (→diogu) *negar **egitea*** (→egiteari). *(We have to stop crying)* |
| E3: Concordance verb-subject | ***Jon*** (→Jonek) *ez daki ezer. (Jon doesn't know anything)*<br>***Bidaiak*** (→Bidaiek) *atsedena hartzeko balio dute. (Travelling is good to rest)*<br>*Jende askok uste **dute*** (→du). *(A lot of people think).* |
| E4: Completive sentences | *Ez dut uste hori egia **dela*** (→denik). *(I don't think that's true)*<br>*Nire ustez, hori horrela **dela*** (→da). *(I think it's like that)*<br>*Badago beste kutsadura bat **dela*** (→dena) *nuklearra. (There's another contamination which is nuclear contamination)* |

Table 1: Selected errors and examples. In brackets, the English translation of the corrected example

## *3 Generation of synthetic datasets*

### 3.1 Generation of synthetic errors

The size of the different training datasets available for GEC is insufficient for training *seq2seq* neural models, so different techniques for the generation of additional synthetic training data have been proposed in the literature. Some authors (Grundkiewicz y Junczys-Dowmunt, 2014; Lichtarge et al., 2019) have proposed to extract training data from Wikipedia revision histories, a source from which a large number of examples can be extracted, especially for English. Other authors (Reiet al., 2017; Ge et al., 2018) generate synthetic training data following the *back-translation* strategy proposed for machine translation systems. An intermediate model is trained from the initial training corpus to be applied to a corpus of correct sentences, and thus sentences with grammatical errors are automatically generated. Another alternative proposed in the literature to generate synthetic training corpora is to introduce "noise" in a corpus of correct texts. The "noise" or grammatical errors are introduced by means of

linguistic rules (Yuan and Felice, 2013), or more generic operations of token replacement, elimination, insertion or reordering (Zhao et al., 2019).

In our case, to generate the synthetic training corpus we will follow an approach based on linguistic rules, in line with the strategy adopted by Yuan and Felice (2013). However, unlike Yuan and Felice (2013), we will not use an initial annotated corpus as a reference.

The rules are designed to generate specific grammatical errors in grammatically correct sentences. That way, we can generate pairs of incorrect and correct sentences useful for compiling a training dataset.

The implemented rules (Table 2) generate errors of the types E1, E2, E3, E4 (Table 1) described in subsection 3.1. For each type of error a set of rules has been implemented so that most of the possible cases are covered. In some cases the application of the rule is bi-directional depending on whether the error generated in that way is also common.

The changes executed by the implemented rules consist of replacing specific words depending on the type of error (examples shown in Table 3). These replacements are made according to certain grammatical information and specific tokens that we obtain through the morphosyntactic analyzer for the Basque language Eustagger (Ezeiza et al., 1998):

- Rule R1.1 associated with the error type E1 is applied to sentences where the verb tense is future (suffixes *"ko"* and *"go"*) and the verb inflection is modified to transform it into present tense (suffixes *"ten"* and *"tzen"*).
- Rules R2.1, R2.2, R2.3, R2.4 associated with error type E2 modify the auxiliary verb to simulate the most common verbal paradigm confusions.
- Rule R3.1 associated with error type E3 modifies the subject's grammatical case (its declension) to transform ergative cases into absolutive ones.
- Rules R4.1, R4.2, R4.3 associated with error type E4 modify the auxiliary verb to simulate suffix errors in completive sentences.

| *Error* | *Rules* |
|---|---|
| E1 | R1.1: ko/go → ten/tzen |
| E2 | R2.1: nor-nork → nor-nori-nork<br>R2.2: nor-nori-nork ↔ nori-nor<br>R2.3: nor-nork ↔ nor<br>R2.4: nor-nork ↔ nori-nor |
| E3 | R3.1: subj_erg → sub_abs |
| E4 | R4.1: v_aux(-nik) → v_aux(-la)<br>R4.2: v_aux(-na) →v_aux(-la)<br>R4.3: v_aux(-laren) ↔v_aux(-lako) |

Table 2: Rules for errors associated with selected grammatical errors

| *Rule* | *Examples* |
|---|---|
| R1.1 | Arratsaldean **ikusiko** (→*ikusten*) gara. (See you in the afternoon) |
| R2.1 | Atzo hondartzan ikusi **zintudan** (→*nizun*). (I saw you on the beach yesterday) |
| R2.2 | Aholku kontrajarriak ematen ari **zaigu** (→digu). (He/She is giving us contradictory advice) |
| R2.3 | Azkenaldian asko argaldu **du** (→*da*). (He's lost a lot of weight lately) |
| R2.4 | Paisaia asko gustatzen **zait** (→*nau*). (I really like the landscape) |
| R3.1 | **Nik** (→*ni*) ez dut nahi. (*I don't want it).*<br>**Langileek** (→*langileak*) lan handia egin dute. (The workers have worked very hard) |
| R4.1 | Ez dut uste etorriko **denik** (→*dela*). (I don't think he/she's coming) |
| R4.2 | Badago beste arazo bat zehaztasuna **dena** (→*dela*). (There's another problem that is precision) |
| R4.3 | Laster zabaldu da denok gaixotuko **garelaren** (→*garelako*) albistea. (The news that we're all going to get sick has spread fast.) |

Table 3: Examples of errors generated by the implemented rules. In brackets, the English translation of the correct example

## 3.2 Strategies for building datasets

Each example in the training -and evaluation- datasets we create is composed of a sentence pair that includes a sentence containing grammatical errors and its corresponding corrected version. In order to generate those pairs we apply the rules described in the previous subsection over grammatically correct sentences. We also add pairs composed of the

original unmodified sentences so that trained models also take those cases into account.

Those grammatically correct sentences are extracted from a news corpus compiled from several Basque news websites: Berria.eus, Argia.eus and the various proximity media of the Tokikom.eus network. The collected corpus consists of 500,015 news items from which we extract 4,927,748 correct sentences $O_c=\{oc_i\}$ including 66 million words.

To generate the training datasets (Tables 4 and 5) we have analyzed different strategies to apply the rules on a subset of $O_c$ of 4,921,748 sentences:

- Baseline ($D_{t0}$ dataset): We apply to each correct sentence $oc_i$ a variation of the substitution rule proposed by Zhao et al. (2019), since *seq2seq* models trained on data generated using the original method performed very poorly. The original method consists in replacing, with a 10% probability, each word with another randomly selected word from the corpus. To avoid highly artificial sentences, our variation adds the constraint that the bigram ($w_1$, $w_2$) exists in the corpus, where $w_2$ is the randomly selected word and $w_1$ is the word preceding the original replaced word. For each correct sentence we generate the pair ($z(oc_i)$,$oc_i$) composed of the incorrect sentence $z(oc_i)$ and the correct $oc_i$, in addition to the unmodified pair ($oc_i$,$oc_i$).
- Strategy 1 ($D_{t1}$ dataset): From the rules that can be applied to the correct sentence $oc_i$, a single $R_i$ rule is randomly selected and the pair ($R_i(oc_i)$,$oc_i$) is generated in addition to the unmodified pair ($oc_i$,$oc_i$). If no rule can be applied, only the pair ($oc_i$,$oc_i$) is generated.
- Strategy 2 ($D_{t2}$ dataset): For each $R_i$ rule that can be applied to the correct $oc_i$ sentence, the pair ($R_i(oc_i)$,$oc_i$) is generated, in addition to the unmodified pair ($oc_i$,$oc_i$). If no rule can be applied, only the pair ($oc_i$,$oc_i$) is generated.
- Strategy 3 ($D_{t3}$ dataset): We apply a set of defined rules to each correct sentence $oc_i$. From the rules that can be applied, a set $\{R_j\}$ of $n$ rules is selected at random, and applied sequentially to generate the pairs ($R_1(oc_i) \circ \ldots \circ R_n(oc_i)$,$oc_i$) and ($oc_i$,$oc_i$). If no rule can be applied, only the pair ($oc_i$,$oc_i$) is generated.

In the different training datasets created we differentiate three types of example pairs according to the number of rules applied for their generation: a) *None*: they are not the result of any rule, b) *Single*: they are the result of applying one rule, c) *Multi*: they are the result of applying several rules (Table 5).

| | *Pairs* | *R1* | *R2* | *R3* | *R4* |
|---|---|---|---|---|---|
| $D_{t0}$ | 8.49M | - | - | - | - |
| $D_{t1}$ | 9.33M | 0.37M | 3.43M | 0.54M | 0.07M |
| $D_{t2}$ | 17.38M | 1.04M | 9.66M | 1.60M | 0.19M |
| $D_{t3}$ | 9.33M | 0.59M | 3.84M | 0.89M | 0.10M |
| $D_{ea}$ | 6000 | 662 | 2924 | 871 | 1291 |
| $D_{em}$ | 672 | 49 | 307 | 92 | 143 |

Table 4: Number of total pairs (*None* included) and pairs generated by each rule included in the training ($D_{t0}$, $D_{t1}$, $D_{t2}$, $D_{t3}$) and evaluation ($D_{ea}$, $D_{em}$) datasets

| | *Pairs* | *None* | *Single* | *Multi* |
|---|---|---|---|---|
| $D_{t0}$ | 8.49M | - | - | - |
| $D_{t1}$ | 9.33M | 4.92M | 4.41M | 0 |
| $D_{t2}$ | 17.38M | 4.91M | 12.47M | 0 |
| $D_{t3}$ | 9.33M | 4.92M | 3.17M | 1.24M |
| $D_{ea}$ | 6000 | 2000 | 2000 | 2000 |
| $D_{em}$ | 672 | 250 | 221 | 201 |

Table 5: Total number of pairs (*Pairs*) and pairs according to typology (*None*, *Single*, *Multi*) included in the training ($D_{t0}$, $D_{t1}$, $D_{t2}$, $D_{t3}$) and evaluation ($D_{ea}$, $D_{em}$) datasets

To create the evaluation datasets we use the same strategies as those used to create the training datasets, but on a different subset (6k correct sentences) of $O_c$. We guarantee a balance between the pair types *None*, *Single* and *Multi*. In this way, we generate a first evaluation dataset $D_{ea}$ fully automatically. Taking into account that, in some cases, the application of the rules can generate grammatically correct sentences, we also built another $D_{em}$ evaluation dataset consisting of a subset of 750 $D_e$ pairs but reviewed manually. Pairs generated by the rules that do not really include grammatical errors are eliminated. 78 pairs were discarded in the manual review

process, leaving a subset of 672 pairs (see tables 4 and 5).

## 4 Seq2seq architecture for GEC

The *seq2seq* neural architectures are being used successfully to address the GEC task. Unlike statistical translation models, *seq2seq* neural architectures can model dependencies between words (or similar word sets) that are critical in correcting grammatical errors (Sakaguchi et al., 2016).

In the literature, we can distinguish three main sequence-to-sequence architectures proposed for the correction of grammatical errors: architectures based on recurrent neural networks (Ge et al., 2018), architectures based on convolutional networks (Chollampatt and Ng, 2018), and architectures based on self-attention (Junczys-Dowmunt et al., 2018). These last ones are the ones we are going to use in this work, since they are the ones that provide better results in this task according to Zhao et al. (2019).

For training the grammatical correction models we have chosen the Transformer architecture (Vaswani et al., 2017). Specifically, we have used the implementation of the OpenNMT-py library. The Transfomer architecture is based on an encoder-decoder system with an attention mechanism. Both the encoder and the decoder are composed of 6 layers composed in turn by a recurrent neural network and a mechanism of attention. We have used the default values of the architecture without any optimization of the parameters. The size of the recurrent neural network of each layer is 512. Thus, 512 size embeddings have been used for both incorrect and corrected sentences. The Adam optimizer has been used during the training, and a learning-rate of 2 with a warm-up phase of 8000 steps. The dropout ratio is 0.1, the batch size is 4096 sentences, and the models have been trained until the results on the development set have not shown any improvement. For the development set 5000 sentence pairs have been selected randomly from the training data.

To avoid the open vocabulary issue and for a better translation of unknown words, BPE tokenization (Sennrich et al., 2016) has been applied to source and target sequences. Rare or unseen words are represented as a sequence of subword units. In the case of Basque, this encoding is particularly useful as declensions generate a larger vocabulary.

## 5 Results

We present results for four GEC systems. All of them are based on the Transformer model introduced in the previous section and trained on the synthetic datasets presented in section 3. Those systems were evaluated according to the standard metrics used in GEC: precision, recall and $F_{0.5}$ with respect to the set of edits needed to correct the incorrect sentences. The upperbound would be an oracle system that makes only the necessary edits to correct the errors included in the incorrect sentences. The following systems were built and evaluated:

- $D_{t0}$+*tr system*: Training of the Transformer model from the training dataset $D_{t0}$ (synthetic examples by random word replacement).
- $D_{t1}$+*tr system*: Training of the Transformer model from the training dataset $D_{t1}$ (Synthetic examples by application of a rule by sentence).
- $D_{t2}$+*tr*: Training of the Transformer model from the training dataset $D_{t2}$ (synthetic examples by application of n rules per sentence)
- $D_{t3}$+*tr*: Training of the Transformer model from the training dataset $D_{t3}$ (synthetic examples by simultaneous application of *n* rules per sentence)

Tables 6, 7, 8 and 9 show the results obtained for each of the systems with respect to the evaluation datasets, both the automatic $D_{ea}$ and the manually reviewed $D_{em}$.

The best results are obtained by the $D_{t3}$+*tr* system, on both evaluation datasets and also on the *Single* (sentences with one error) and *Multi* (sentences with more than one error) subsets, as well as on the four types of errors (E1, E2, E3 and E4). The Transformer model seems able to better learn the task from examples that can combine more than one error, which is the configuration of the $D_{t3}$ training dataset. Error analysis revealed that $D_{t3}$+*tr* works well with sentences with more than one error, but tends to make incorrect fixes in sentences with no errors or containing a single error.

The $D_{t0}$+*tr* system trained from the baseline dataset has a very low performance, and points out that the generic replacement rules are not adequate to generate synthetic training datasets, at least in this case study. The model does not have enough information to perform the necessary fixes. It does not create new mistakes, but neither corrects them.

The results of the $D_{t1}$+*tr* and $D_{t2}$+*tr* systems differ slightly from each other, the former being better. But the results of both are notably lower than those of $D_{t3}$+*tr*, especially in terms of recall. This difference in performance with respect to $D_{t3}$+*tr* is especially accentuated (see table 7) when dealing with sentences with more than one error (*Multi*). These systems rarely solve more than one error in the same sentence.

With regard to the different types of errors, there are no major differences, and in general better results are obtained for types E1 and E2 (see tables 8 and 9).

| | $D_{ea}$ | | | $D_{em}$ | | |
|---|---|---|---|---|---|---|
| | *P* | *R* | $F_{0.5}$ | *P* | *R* | $F_{0.5}$ |
| $D_{t0}$+*tr* | 0.26 | 0.02 | 0.07 | 0.26 | 0.02 | 0.07 |
| $D_{t1}$+*tr* | 0.86 | 0.57 | 0.78 | 0.88 | 0.58 | 0.80 |
| $D_{t2}$+*tr* | 0.83 | 0.56 | 0.75 | 0.80 | 0.60 | 0.75 |
| $D_{t3}$+*tr* | **0.88** | **0.73** | **0.85** | **0.90** | **0.76** | **0.87** |

Table 6: Precision (P), recall (R) and $F_{0.5}$ of the systems with respect to the automatic ($D_{ea}$) and manually reviewed ($D_{em}$) datasets

| | $D_{ea}$ | | | $D_{em}$ | | |
|---|---|---|---|---|---|---|
| | *S* | *M* | *S*+*M* | *S* | *M* | *S*+*M* |
| $D_{t0}$+*tr* | 0.08 | 0.06 | 0.07 | 0.06 | 0.09 | 0.07 |
| $D_{t1}$+*tr* | 0.81 | 0.79 | 0.80 | 0.84 | 0.81 | 0.82 |
| $D_{t2}$+*tr* | 0.82 | 0.77 | 0.79 | 0.83 | 0.78 | 0.80 |
| $D_{t3}$+*tr* | **0.83** | **0.88** | **0.86** | **0.86** | **0.90** | **0.89** |

Table 7: $F_{0.5}$ results of the systems with respect to the *Single* (*S*) and *Multi* (*M*) subsets of the evaluation datasets

| | *E1* | *E2* | *E3* | *E4* |
|---|---|---|---|---|
| $D_{t0}$+*tr* | 0.07 | 0.07 | 0.04 | 0.06 |
| $D_{t1}$+*tr* | 0.81 | 0.81 | 0.74 | 0.77 |
| $D_{t2}$+*tr* | 0.79 | 0.79 | 0.72 | 0.78 |
| $D_{t3}$+*tr* | **0.88** | **0.87** | **0.86** | **0.85** |

Table 8: $F_{0.5}$ results of the systems with respect to the automatic evaluation dataset $D_{ea}$ depending on the grammatical error to correct

| | *E1* | *E2* | *E3* | *E4* |
|---|---|---|---|---|
| $D_{t0}$+*tr* | 0.07 | 0.07 | 0.01 | 0.06 |
| $D_{t1}$+*tr* | 0.88 | 0.83 | 0.76 | 0.79 |
| $D_{t2}$+*tr* | 0.85 | 0.79 | 0.72 | 0.82 |
| $D_{t3}$+*tr* | **0.94** | **0.90** | **0.90** | **0.87** |

Table 9: $F_{0.5}$ results of the systems with respect to the manually revised dataset $D_{em}$ depending on the grammatical error to be corrected

## 6 Conclusions

In this work we have been able to prove that it is possible to implement a grammar checker based on *seq2seq* neural models for a less-resourced language, represented by Basque in this case of study. For this type of language where no training data is available for the GEC task, we have found that a strategy based on building synthetic training datasets from monolingual corpora is feasible. The proposed method, based on combining different linguistic rules to generate grammatical errors, allows the creation of large valid datasets to train high performance *seq2seq* neuronal models. In the future, we plan to extend the repertoire of linguistic rules for generating synthetic errors, and also study other methods of synthetic data generation, in order to include more types of grammatical errors in the *seq2seq* model.

## References

Brockett, C., W. B. Dolan and M. Gamon. 2006. Correcting ESL errors using phrasal SMT techniques. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics* (pp. 249-256).

Chollampatt, S. and H.T. Ng. 2018. A multilayer convolutional encoder-decoder neural network for grammatical error correction. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Ezeiza, N., I. Alegria, J.M. Arriola, R. Urizar and I. Aduriz. 1998. Combining stochastic and rule-based methods for disambiguation in agglutinative languages. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1* (pp. 380-384).

Gamon, M. 2010. Using mostly native data to correct errors in learners' writing: a meta-classifier approach. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 163-171).

Ge, T., F. Wei and M. Zhou. 2018. Fluency boost learning and inference for neural grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1)* (pp. 1055-1065).

Grundkiewicz, R. and M. Junczys-Dowmunt. 2014. 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).

Izumi, E., K. Uchimoto, T. Saiga, T. Supnithi and H. Isahara. 2003. Automatic error detection in the Japanese learners' English spoken data. In *The Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics* (pp. 145-148).

Junczys-Dowmunt, M., R. Grundkiewicz, S. Guha and K. Heafield. 2018. Approaching Neural Grammatical Error Correction as a Low-Resource Machine Translation Task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)* (pp. 595-606).

Lichtarge, J., C. Alberti, S. Kumar, N. Shazeer, N. Parmar and S. Tong. 2019. Corpora Generation for Grammatical Error Correction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* (pp. 3291-3301).

Oronoz, M. 2009. *Euskarazko errore sintaktikoak detektatzeko eta zuzentzeko baliabideen garapena: datak, postposizio-lokuzioak eta komunztadura* (Doctoral dissertation, Universidad del País Vasco-Euskal Herriko Unibertsitatea).

Rei, M. and H. Yannakoudakis. 2017. Auxiliary Objectives for Neural Error Detection Models. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications* (pp. 33-43).

Rozovskaya, A. and D. Roth. 2016. Grammatical error correction: Machine translation and classifiers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 2205-2215).

Sakaguchi, K., C. Napoles, M. Post and J. Tetreault. 2016. Reassessing the goals of grammatical error correction: Fluency instead of grammaticality. *Transactions of the Association for Computational Linguistics*, *4*, 169-182.

Sennrich, R., B. Haddow, and A. Birch. 2016. Edinburgh Neural Machine Translation Systems for WMT 16. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task* (pp. 371-376).

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

Yuan, Z. and M. Felice. 2013. Constrained grammatical error correction using statistical machine translation. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task* (pp. 52-61).

Zhao, W., L. Wang, K. Shen, R. Jia and J. Liu. 2019. Improving Grammatical Error Correction via Pre-Training a Copy-Augmented Architecture with Unlabeled Data. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1* (pp. 156-165).