Beqi: Revitalize the Senegalese Wolof Language with a Robust Spelling Corrector

Derguene Mbaye[1,2(✉)] and Moussa Diallo[2]

[1] Baamtu, Dakar, Senegal
[2] Université Cheikh Anta Diop, Dakar, Senegal
{derguenembaye,moussa.diallo}@esp.sn
https://www.baamtu.com/

Abstract. The progress of Natural Language Processing (NLP), although fast in recent years, is not at the same pace for all languages. African languages in particular are still behind and lack automatic processing tools. Some of these tools are very important for the development of these languages but also have an important role in many NLP applications. This is particularly the case for automatic spell checkers. Several approaches have been studied to address this task and the one modeling spelling correction as a translation task from misspelled (noisy) text to well-spelled (correct) text shows promising results. However, this approach requires a parallel corpus of noisy data on the one hand and correct data on the other hand, whereas Wolof is a low-resource language and does not have such a corpus. In this paper, we present a way to address the constraint related to the lack of data by generating synthetic data and we present sequence-to-sequence models using Deep Learning for spelling correction in Wolof. We evaluated these models in three different scenarios depending on the subwording method applied to the data and showed that the latter had a significant impact on the performance of the models, which opens the way for future research in Wolof spelling correction.

Keywords: Spelling correction · Spell checking · Deep Learning · LSTM · Transformer · Low-resource languages · African languages · Wolof

1 Introduction

Spelling mistakes are common in language usage and can be due to a lack of language skills or carelessness. They can become an important element to take into account when writing emails, speeches or when searching on the internet. This

Supported by the Google PhD Fellowship program.

K. Cheikh M. F. Kebe et al. (Eds.): InterSol 2024, LNICST 610, pp. 311–325, 2025.
https://doi.org/10.1007/978-3-031-86493-3_25

is the reason why automatic correctors can be found in various NLP applications such as Summarization [1], Machine Translation [2] and Search Engines [3]. Regarding Wolof specifically, it is a language that is more spoken than written, like most African languages. The Wolof alphabet has been defined by presidential decree since 1971[1] as well as spelling and word separation in 2005[2] but its adoption remains weak. Although it is the predominant language spoken in Senegal (statistically), Wolof is not taught in school as it has been supplanted by French, the official language since colonization. All these aspects contribute to the fact that the majority of the population has a weak grasp of the writing of this language and it is common to note spelling mistakes on social networks, advertising posters and even in television programs. Nevertheless, in recent years there has been a significant resurgence of interest in the language and several initiatives to revitalize it have been launched. A group of linguists called WAX ("Wolof Ak Xamle" meaning Wolof and knowledge sharing) has been created and is working on the popularization of Wolof[3] by content creation and the launch of an e-learning platform[4], among other things. All these initiatives contributed greatly to the acceleration of the adoption of the written form of this language.

However, incorrect writing has become so widespread that it can be considered as an orthographic system to which we will refer in this article by the term "conventional form". The one based on the official spelling will be called "Official Form". The existence of these two forms of writing creates a gap that can greatly hinder the performance of NLP applications designed for Wolof. In fact, the datasets collected to date in Wolof [4–7] are based on the official alphabet and the spelling used is different from the conventional form that is commonly used by the population. NLP applications designed from these datasets will therefore have a lot of trouble working once in production due to this gap. Figure 1 and Fig. 2 illustrate this problem with the translation system designed in [8] by Meta researchers[5].

It is thus crucial to have a spell checker in Wolof in order to bridge the gap between the conventional form and the official one. Wolof being a low-resource language, it makes this task even more challenging.

It is in this context that we introduce Beqi[6]: the first Deep Learning-based Wolof spelling corrector for end-to-end learning. We structured the paper as follows:

- We begin by presenting the work done in automatic spell correction in Wolof and other low-resource languages in Sect. 2.
- Data collection and synthetic data generation are discussed in Sect. 3.

[1] *Decree No. 71-566 of May 21st, 1971 concerning the transcription of national languages. Republic of Senegal, 1971.*

[2] *Decree no. 2005-992 of October 21st, 2005 concerning the spelling and separation of words in Wolof (currently effective).*

[3] Senegal: The Titan work of Wolof language academics, by *le360 Afrique (French)*.

[4] https://jangwolof.com/.

[5] The translations were performed with the NLLB model distilled to 600M parameters.

[6] A Wolof word meaning the action of correcting.

Fig. 1. A correctly done translation when the sentence in Wolof is written with the official form. The correct translation of `pàkk` would be `plot of land` but the overall meaning is maintained.

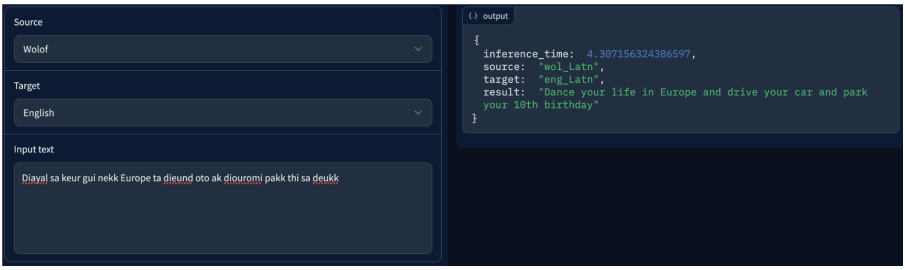

Fig. 2. A totally wrong translation when the same sentence is written with the conventional form.

– In Sect. 4, we present the model used and the experiments.
– Sect. 5 presents the results and perspectives.
– The conclusion is presented in Sect. 6.

2 Related Work

Several approaches have been studied to address the problem of automatic spelling correction in general. The study in [9] divides these approaches into three groups:

1. One that is based on expert rules;
2. One that adds a context model to rearrange candidate corrections;
3. One that learns error patterns from a set of training data.

A portable spellchecker for the Amharic language, spoken in Ethiopia, was developed in [10]. The system uses a corpus-driven approach that uses a noisy

channel to derive linguistic knowledge to correct spelling errors. Grammatical error correction in low-resource scenarios was studied in [11] with a focus on the Czech language. The researchers modeled the correction task as a machine translation task with a Transformer-based model [12]. Indian languages have also been studied in the spelling correction task in particular in [13] which uses a Deep Learning based approach and targets Hindi and Telugu languages. Their approach also leverages the machine translation framework and uses a sequential encoder-decoder model based on the Long short-term memory (LSTM) architecture [14].

Although significant work has been done in spelling correction in low-resource languages, little work has been done in this area for Wolof specifically. Several dictionaries were developed in the context of the Dictionnaires Langue Africaine-Français (DiLAF) project which covered five other African languages in addition to Wolof [15]. The implementation of a spellchecker for Wolof was studied in [16] with an approach based on a French-Wolof dictionary studied in [17] as a lexicon and a morphological analyzer of the Wolof language explored in [18]. But the work did not go as far as the actual implementation of a functional corrector and was limited to the state of the art of methods based on the first two previously mentioned approaches i.e. those based on expert rules and those using a context model based on n-gram language models. In addition, at the time of writing this article, all dictionaries developed in [15] are available online[7] except Wolof, which prevents us from exploring a dictionary-based approach. These are also difficult to maintain (the number of rules can quickly increase and their update is tedious), are limited by the size of the dictionary and do not take into account the context. The latter can be included thanks to a context model which is generally an n-gram language model [19] that defines the probability according to the history of the words. This language model thus only takes into account the previous words in addition to the current word, which limits the context considered. Although additional classifiers can be used to bridge this gap in context [9], the use of neural networks allows the inclusion of a broader context on both sides of a word.

Deep Learning is thus a promising approach that has been studied for the spelling correction task and for different languages. But to the best of our knowledge, this is the first attempt applied to the Wolof language.

3 Data Collection

We have collected an in-house dataset of 154,000 correctly written sentences in Wolof which is an extension of the dataset presented in [20]. These sentences were obtained by first collecting monolingual French data from various sources: Coran, Bible, books and news sites as illustrated in Fig. 3. Since Senegal is a French-speaking country, it is easier to find linguists who master both languages (Wolof and French) in order to make the best possible translations. We thus collaborated with a team of linguists from the Linguistic Department of the Cheikh Anta Diop

[7] Website of the DILAF project.

University of Dakar to manually translate the collected French corpus into Wolof. The Wolof corpus thus collected and written in the official form constitutes the "target language" that we wish to have as output. To obtain the data of the "source language" written in conventional Wolof, we scraped data on Twitter from accounts that generally publish in Wolof in order to detect recurrent spelling error patterns. Indeed, Twitter is a micro-blogging platform where people write casually about various topics. This makes it an ideal candidate for collecting data that may contain spelling errors and the platform is much in demand for data collection for NLP tasks such as Sentiment Analysis [21]. Author accounts of Wolof publications were identified using Twitter's advanced search functionality by searching for conventional Wolof keywords that appear in tweets and picking up the corresponding authors. From there, we scrape a sample of tweets and identify patterns of errors that we will subsequently reproduce on our corpus written in official form. The overall collection process is illustrated in Fig. 4.

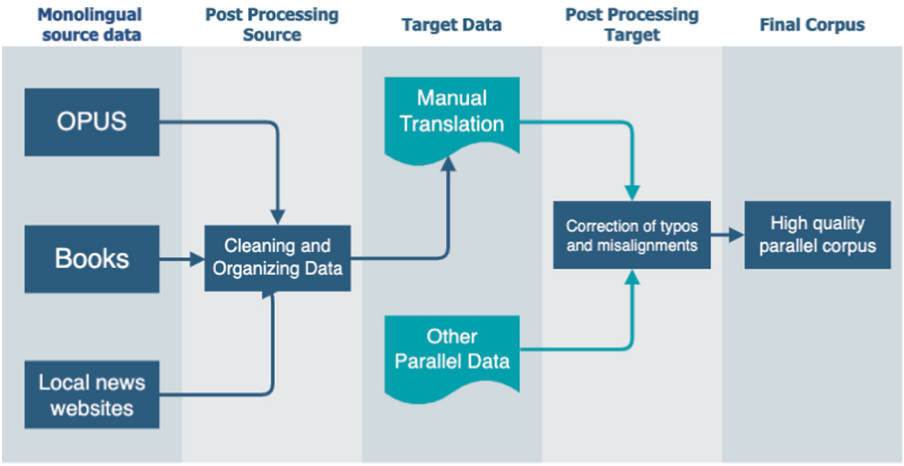

Fig. 3. Data collection pipeline

However, since people are generally bilingual, many publications are also written in French which includes artifacts in the collected data. To filter them, we first used the language identification model [22] included in the polyglot library[8] in its *16.7.4* version to detect the languages of the tweets in order to remove those in French. However, we encountered the problem illustrated in Fig. 2 where the model struggles to detect the language when the text is written in conventional form as illustrated in Fig. 5. We thus had to do the filtering manually.

[8] http://www.polyglot-nlp.com/.

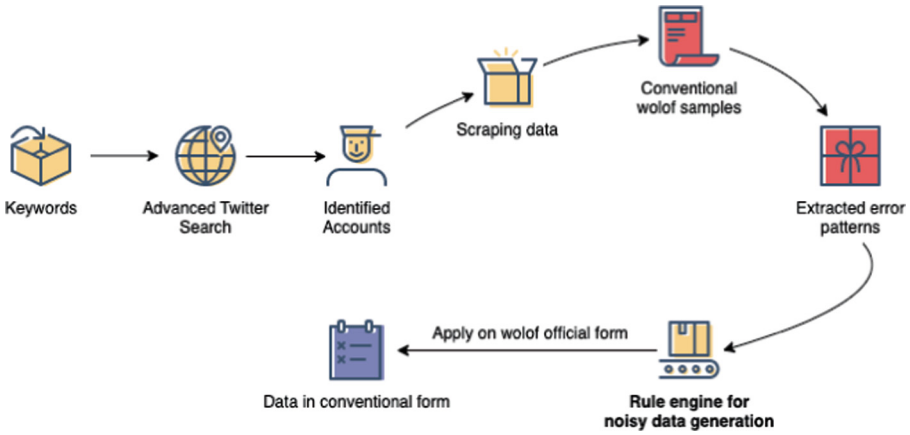

Fig. 4. The collection steps that led to the generation of the parallel corpus with noisy data as source and correctly written data as target.

```
WARNING:polyglot.detect.base:Detector is not able to detect the language reliably.
- Sentence: balma leral ma yane diafei diafei nga am fi mou tolou?
- Classification: name: Tongan      code: to       confidence:  98.0 read bytes:   208
------------------------------------------------------------------------
- Sentence: baalma leeralma yan jafe-jafe nga am fimu tollu?
- Classification: name: Wolof       code: wo       confidence:  97.0 read bytes:   490
```

Fig. 5. The model failing to recognize the text language when written in the conventional form (above the dash line) and succeeding when written in the official form.

Once samples of texts in conventional form were collected, we designed a rule engine based on regular expressions[9] where each identified error pattern is transcribed into a defined rule to be applied on the corpus. For example, in the conventional sentence "`Diappal bal bi`" meaning "`Catch the ball`", we derive the rule that the "J" followed by a vowel (except "i") is often wrongly replaced by the string "Di". Thus the correct writing of the previous sentence is "`Jàppal bal bi`". We reproduce this error in our corpus by replacing all the times where the letter "J" is followed by a vowel that is not "i", by the string "Di". The Table 1 presents the rules used in the engine when pre-processing the data. Two other rules were applied in a post-processing phase: one to remove spaces between a word and a vowel (used in formal Wolof to express a plurality for example) and another one to replace occurences of 'g' followed by vowels by 'gu'. All this process allowed us to collect as much synthetic data as formal data i.e. 154,000 parallel sentences of noisy text on one side and well written text on the other, that will be used to train the final spelling correction model. Some examples of the resulting output of this transformation are shown in Table 2.

[9] Patterns used to match character combinations in strings.

Table 1. Patterns used in regular expressions to map the correct writing to manually identified errors on the collected data ("f/b" means "followed by").

Patterns	Replacement	Description
ñ+	gn	Replace occurences of 'ñ' by 'gn'
η+	ng	Replace occurences of 'η' by 'ng'
ë+	eu	Replace occurences of 'ë' by 'eu'
u+	ou	Replace occurences of 'u' by 'ou'
u([blt]+)	ou\1	Replace occurences of 'ub/l/t' by 'oub/l/t'
q	kh	Replace every 'q' character by 'kh'
x	kh	Replace every 'x' character by 'kh'
u\b	ou	Replace words ended with 'u' by 'ou'
c([aeiouy]{1,})	th\1	Replace occurences of 'c' f/b vowels by 'th'
c{2}\b	thie	Replace 'cc' at the end of a word by 'thie'
[Jj]([eao]1,2)	di\1	Replace 'j' f/b a vowel (except i and u) by 'di'
[Jj]([i]+)	dj\1	Replace 'j' f/b occurences of 'i' by 'dj'
[Jj]([u]+)	dio\1	Replace 'j' f/b occurences of 'u' by 'dio'
th([aeouy]+)	thi\1	Replace occurences of 'th' f/b vowels (except i) by 'th'

Table 2. Examples of sentences in the official form converted to conventional form by the rule engine.

Official form	Conventional form
Nàngul kula raw, kula ëppalé	Nangoul koula raw, koula euppale
Kula gën a taaru ak kula mag	Koula gueuna taarou ak koula mag
Yii yëpp dula wàññi dara	Yii yeupp doula wagni dara
Wànté bul nangu mukk kula gën	Wante boul nangou moukk koula gueun
Lilakoy may, mooy nga sàmm sa ngor	Lilakoy may, mooy ngua samm sa ngor

4 Experiments

When designing the spelling correction system we considered two architectures commonly used in sequence-to-sequence mapping tasks: the LSTM [14] and the Transformer [12]. LSTMs are a particular type of Recurrent Neural Networks (RNNs) [23], consisting of several gates that allow them to manipulate the information flow. This manipulation is performed by forgetting or selectively memorizing the information of the previous temporal sequence in a dynamic memory as shown in Fig. 6[10].

[10] LSTMs Explained: A Complete, Technically Accurate, Conceptual Guide with Keras.

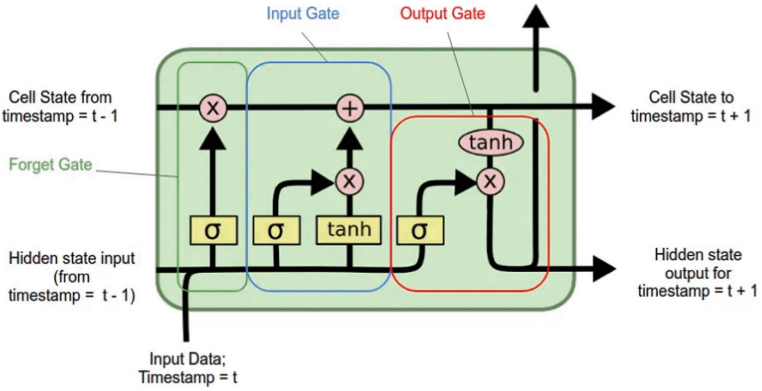

Fig. 6. Illustration of an LSTM cell

The LSTM is a sequential model in which one element of the sequence is processed at a time, which is not the case for the Transformer, illustrated in Fig. 7. The Transformer is a Deep Learning model (i.e. a neural network) of the seq2seq type (takes a sequence as input and returns a sequence as output) which has the particularity of only using the attention mechanism and no recurrent

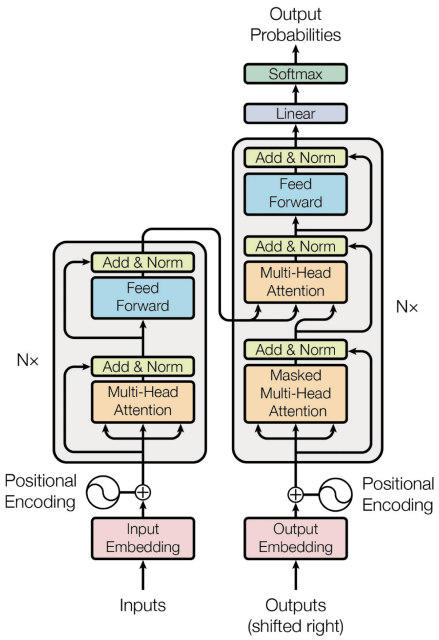

Fig. 7. Illustration of a Transformer architecture as presented in [12]

or convolutional network. The Transformer is more efficient in tracking remote dependencies but is however more data intensive.

To implement them, we used the OpenNMT library [24] which is an open source ecosystem for neural machine translation and neural sequence learning. Our implemented LSTM consists of two layers of encoders and two layers of decoders with 500 hidden units, an embedding of size 500 and a dropout layer associated with a rate of 0.3. As optimizer we defined the Stochastic Gradient Descent [25] with a learning rate of 1.0 as recommended in the OpenNMT documentation[11]. We added a global attention layer [26] which is a simplification of the "classic" attention mechanism proposed in [27] and which may achieve better results. Regarding the Transformer-based architecture, we have reproduced the same features as those defined in the base paper (Vanilla Transformer) [12]. To adapt it to our low-resource context, we reduced some parameter values such as the number of training steps, the early-stopping threshold, the number of validation steps and the warmup steps.

We then divided our dataset into train, validation and test sets with 140,000 sentences for the train set and 7,000 sentences for each of the remaining sets. We applied stratified sampling to ensure that each subset is representative of the overall dataset. This enables a fairer evaluation of the model's performance and limits the biases that can creep into the process. We then applied two different types of segmentation[12] on the data: SentencePiece [28] and Character-Level Subwording which has been shown in [29] to be very effective in translation tasks. All models are trained until convergence, which we consider as reached when no improvement on the validation set is observed after 04 epochs. In the case where the model does not converge, we set a limit of 30k epochs to stop the training. We then compared the performance of the models on the raw data (not subworded) and on the subworded data to evaluate the impact of this process on the models performance. The vocabularies used on the datasets are generated on all segments of training sets and models are evaluated with Accuracy[13] as a metric calculated on test sets at a sentence level. All experiments took place on a virtual machine with a Tesla V-100 GPU with 16 GB of RAM.

5 Results and Perspectives

We compared the two models on the same dataset in different subwording scenarios and Table 3 shows their performances in accuracy given in percentage. We notice that the LSTM model greatly outperforms the Transformer one when no subwording is applied with accuracies of 50.09% and 9.46% respectively which are the lowest performances. We observe a similar pattern with the SentencePiece subworded data where the LSTM still outperforms the Transformer with an accuracy of 69.14% versus 6.99%. The highest scores were achieved with character-level subworded data where the Transformer performed the best with

[11] https://opennmt.net/OpenNMT-py/options/train.html.
[12] Task of dividing a text into coherent and semantically meaningful segments.
[13] The percentage of correct predictions made by a model.

an accuracy of 81% compared to the LSTM which achieved an accuracy of 77.67%. In fact, when no tokenization is applied, it tends to reduce the size of the vocabulary to the total number of words in the corpus. This has the effect of limiting the occurrence of words and reduces the ability of the model to learn these words [30]. The models are thus very sensitive to rare or out-of-vocabulary words (OOV), which results in the generation of `<unk>` tags during predictions and greatly hinders models' capabilities. In addition, since the Transformer has significantly more parameters than the LSTM, it requires much more data to capture the most error patterns. This may explain why it performs poorly than LSTM under these conditions. This problem is therefore alleviated by tokenizing into subwords or at a character level and the latter has the great advantage to make the model usable on all languages. Furthermore, since most spelling errors occur at the character level (omission, addition and replacement), a model that processes text under these conditions will have a greater ability to capture these kinds of errors.

Table 3. Performance of LSTM and Transformer models evaluated with Accuracy on synthetic Wolof data depending on the type of subwording applied.

Model Architecture	Accuracy (%)	
LSTM	**No Subword**	**50.09**
	SentencePiece	**69.14**
	Character-level	77.67
Transformer	No Subword	09.46
	SentencePiece	06.99
	Character-level	**81.00**

Table 4 and 5 show some predictions of the LSTM and Transformer models on character-level subworded data. We notice that both models are able to learn the errors while considering the context. This can be attributed to the attention module which integrates information from surrounding words into the embedding of the current word. The other advantage of these models over dictionary-based models is that they are very robust to out-of-vocabulary words. They are also scalable in the sense that their performance increases as they are used when live data is collected back, corrected and then re-injected as training data, which makes them very powerful. However, we note that the LSTM model fails in some cases such as the last two rows of Table 4 where it seems to have trouble correcting accents while the Transformer model did a perfect job on the considered extract. The latter nevertheless still seems to have concerns about handling accents as shown in Table 5. The first row of this table also illustrates an error on the reference side, which suggests the presence of artifacts in the training data that may explain this phenomenon. The last two rows illustrate an interesting problem related to the rule engine. Some rules appear to be too generic, so that

some errors that are generated by them would never occur in "real" text (e.g., some types of character replacements may only occur in certain contexts, such as the beginning, middle or end of words, etc.). We observe this phenomenon in the first row of Table 4, where the word "juin" (French word meaning June) is changed to "diouin", representing an error that would never occur in a real context. Such over-generation could lead the model to make corrections when it should not, as in the case of proper nouns (e.g. Kouchner, last line of Table 5) or with some common nouns such as Espagnol (second last line).

Table 4. Qualitative evaluation of the Character-level LSTM predictions on few conventional Wolof inputs along with corresponding correction (prediction) expected outputs (reference).

Input	Prediction	Reference
Ndieekhitaloum diouin bi	Njeexitalum juin bi	Njeexitalum juin bi
Daa nourou kou beg	Daa nuru ku bég	Daa nuru ku bég
Dougnou leen dakh	Duñu leen dàq	Duñu leen dàq
Gnoo and ak orob	ñooànd ak **orob**	Ñoo ànd ak **órób**
Bignouy oubbi bank bi	Biñuy ubbi **bank** bi	Biñuy ubbi **bànk** bi

Table 5. Qualitative evaluation of the Character-level Transformer predictions on few official Wolof sentences (model's outputs) along with references (expected outputs)

Input	Prediction	Reference
Nitou loot ya weddi woon nagnou	Nitu Lóot ya weddi woon nañu	Nitu **Loot** ya weddi woon nañu
Yeena nou moucthial noun gnepp	**Yeen** a nu muccal nun ñépp	Yéen a nu muccal nun ñépp
Yakkamti naa degg lignouy wakh	Yàkkamti naa dégg liñuy wax	**Yakkamti** naa dégg liñuy wax
Espagnol bi dooleel na kou gnoul ki	**Espagnol** bi dooleel na ku ñuul ki	**Español** bi dooleel na ku ñuul ki
Na dem te yobbaale Bhl ak Kouchner	Na dem te yóbbaale Bhl ak **Kuchner**	Na dem te yóbbaale Bhl ak Kouchner

This paper is an initial work opening the way to investigate Deep Learning based approaches to address the spelling correction problem in Wolof. The Transformer model already shows promising performances and can be further improved to better adapt to low-resource scenarios as studied in [31]. In perspective, we will further analyze the nature of the errors made by the model in order to study the appropriate solutions.

We will also improve our noisy data generator by collaborating with linguists to better identify common errors and create corresponding rules. The e-learning platform of the WAX group of linguists would be very useful to collect data from dictated exercises performed by students. A similar approach has been taken in [32] which has resulted in a high quality, real-world corpus. We will also explore unsupervised approaches to learn common errors from a noisy corpus instead

of a rule engine like the one used in this paper. We also plan to extend the polyglot language identification model on the collected synthetic data to improve its performance in detecting conventional Wolof. This will allow us to later scrape real data from social networks, have it corrected by linguists and then use the resulting parallel corpus to fine-tune our spelling correction model on it. This is particularly important in order to take into account sensitive phenomena such as code-switching, which refers to the passage from one language to another in the same conversation. This phenomenon is very characteristic of everyday Wolof which is strongly influenced by French. We will thus explore NLP approaches addressing this phenomenon of code-switching as studied in [33] in order to make the model more robust to real-world cases. We will also explore other tokenization mechanisms specific to Wolof that could be more efficient than the Character-Level one used here and extend the current system to a model that can make the correspondence in both directions between the two forms of writing.

6 Conclusion

We presented the first dataset for spelling correction in Wolof to date, as well as the first approach that addresses the issue from a Deep Learning and Machine Translation perspective. The corpus contains 154,000 sentences, making it the largest parallel corpus collected to date for wolof spelling correction. As the collection is still in progress, the datasets are not yet publicly available. In addition, we have performed experiments on the two most used NLP architectures, namely the LSTM and the Transformer, on the collected synthetic data. We implemented these architectures using the OpenNMT library and built baseline models. We evaluated these models based on the Accuracy metric computed at a sentence level and we compared their performance regarding the type of subwording applied to the data. We then showed that the Vanilla Transformer model used on character-level subworded data performed the best. We ended by proposing possible improvements that could broaden the scope of such systems and greatly boost their performance.

We have also shown that such a system is crucial for the proper working of NLP applications for Wolof that are being built and will be built in the future. It could also be a major asset in the adoption of the written form of Wolof through large-scale integration into the keyboards of smartphones and other devices.

References

1. El-Kassas, W.S., Salama, C.R., Rafea, A.A., Mohamed, H.K.: Automatic text summarization: a comprehensive survey. Expert Syst. Appl. **165**(113), 679 (2021). https://doi.org/10.1016/j.eswa.2020.113679. https://www.sciencedirect.com/science/article/pii/S0957417420305030
2. Yang, S., Wang, Y., Chu, X.: A survey of deep learning techniques for neural machine translation (2020). https://doi.org/10.48550/ARXIV.2002.07526
3. Sudeepthi, G., Anuradha, G., Babu, M.S.P.: A survey on semantic web search engine (2012)

4. Tiedemann, J.: Parallel data, tools and interfaces in opus. In: Chair, N.C.C., et al. (eds.) Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC 2012) European Language Resources Association (ELRA), Istanbul, Turkey (2012)
5. Strassel, S., Tracey, J.: LORELEI language packs: data, tools, and resources for technology development in low resource languages. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016), European Language Resources Association (ELRA), Portorož, Slovenia, pp. 3273–3280 (2016). https://aclanthology.org/L16-1521
6. Adelani, D.I., et al.: MasakhaNER: named entity recognition for African languages. Trans. Assoc. Comput. Linguist. **9**, 1116–1131 (2021). https://doi.org/10.1162/tacl_a_00416. https://aclanthology.org/2021.tacl-1.66
7. Goyal, N., et al.: The Flores-101 evaluation benchmark for low-resource and multilingual machine translation. Trans. Assoc. Comput. Linguist. **10**, 522–538 (2022). https://doi.org/10.1162/tacl_a_00474. https://aclanthology.org/2022.tacl-1.30
8. NLLB Team, et al.: No language left behind: scaling human-centered machine translation (2022). https://doi.org/10.48550/ARXIV.2207.04672
9. Hládek, D., Staš, J., Pleva, M.: Survey of automatic spelling correction. Electronics **9**(10), 167 (2020). https://doi.org/10.3390/electronics9101670
10. Gezmu, A.M., Nürnberger, A., Seyoum, B.E.: Portable spelling corrector for a less-resourced language: Amharic. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), European Language Resources Association (ELRA), Miyazaki, Japan (2018). https://aclanthology.org/L18-1651
11. Náplava, J., Straka, M.: Grammatical error correction in low-resource scenarios. In: Proceedings of the 5th Workshop on Noisy User-Generated Text (W-NUT 2019), Association for Computational Linguistics, Hong Kong, China, pp. 346–356 (2019). https://doi.org/10.18653/v1/D19-5545. https://aclanthology.org/D19-5545
12. Vaswani, A., et al.: Attention is all you need. In: Guyon, I., et al. (eds.) Advances in Neural Information Processing Systems, vol. 30. Curran Associates, Inc. (2017). https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf
13. Etoori, P., Chinnakotla, M., Mamidi, R.: Automatic spelling correction for resource-scarce languages using deep learning. In: Proceedings of ACL 2018, Student Research Workshop, Association for Computational Linguistics, Melbourne, Australia, pp. 146–152 (2018). https://doi.org/10.18653/v1/P18-3021. https://aclanthology.org/P18-3021
14. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**, 1735–1780 (1997). https://doi.org/10.1162/neco.1997.9.8.1735
15. Mbodj, C., Enguehard, C.: Production et mise en ligne d'un dictionnaire électronique du wolof 2 (2015). https://talaf.imag.fr/2016/Actes/MBODJ_ENGUEHARD%20-%20Production%20et%20mise%20en%20ligne%20d%E2%80%99un%20dictionnaire%20%C3%A9lectronique%20du%20wolof.pdf
16. Lo, A., et al.: Correction orthographique pour la langue wolof : état de l'art et perspectives. In: JEP-TALN-RECITAL 2016: Traitement Automatique des Langues Africaines TALAF 2016, Paris, France (2016). https://hal.archives-ouvertes.fr/hal-02054917
17. Khoule, M., Mangeot, M., Nguer, E.H.M., Cissé, M.T.: ibaatukaay : un projet de base lexicale multilingue contributive sur le web à structure pivot pour les langues africaines notamment sénégalaises (2016)

18. Dione, C.M.B.: A morphological analyzer for Wolof using finite-state techniques. In: Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2012), European Language Resources Association (ELRA), Istanbul, Turkey, pp. 894–901 (2012). http://www.lrec-conf.org/proceedings/lrec2012/pdf/572_Paper.pdf
19. Goodman, J.: The state of the art in language modeling. In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: Tutorials - Volume 5, Association for Computational Linguistics, USA, NAACL-Tutorials 2003, p 4 (2003). https://doi.org/10.3115/1075168.1075172
20. Mbaye, D., Diallo, M., Diop, T.I.: Low-resourced machine translation for Senegalese Wolof language (2023). arXiv2305.00606
21. Drus, Z., Khalid, H.: Sentiment analysis in social media and its application: Systematic literature review. Procedia Comput. Sci. **161**, 707–714 (2019). https://doi.org/10.1016/j.procs.2019.11.174. https://www.sciencedirect.com/science/article/pii/S187705091931885X
22. Hughes, B., Baldwin, T., Bird, S., Nicholson, J., MacKinlay, A.: Reconsidering language identification for written language resources. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC 2006), European Language Resources Association (ELRA), Genoa, Italy (2006). http://www.lrec-conf.org/proceedings/lrec2006/pdf/459_pdf.pdf
23. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning Internal Representations by Error Propagation, pp. 318–362. MIT Press, Cambridge (1986)
24. Klein, G., Kim, Y., Deng, Y., Senellart, J., Rush, A.: OpenNMT: open-source toolkit for neural machine translation. In: Proceedings of ACL 2017, System Demonstrations, Vancouver, Canada, pp. 67–72. Association for Computational Linguistics (2017). https://www.aclweb.org/anthology/P17-4012
25. Kiefer, J., Wolfowitz, J.: Stochastic estimation of the maximum of a regression function. Ann. Math. Stat. **23**(3), 462–466 (1952). https://doi.org/10.1214/aoms/1177729392
26. Luong, M.T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation (2015). arXiv:1508.04025
27. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate (2016). arXiv:1409.0473
28. Kudo, T., Richardson, J.: SentencePiece: a simple and language independent subword tokenizer and detokenizer for neural text processing. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Association for Computational Linguistics, Brussels, Belgium, pp. 66–71 (2018). https://doi.org/10.18653/v1/D18-2012. https://aclanthology.org/D18-2012
29. Lee, J., Cho, K., Hofmann, T.: Fully character-level neural machine translation without explicit segmentation. Trans. Assoc. Comput. Linguist. **5**, 365–378 (2017). https://doi.org/10.1162/tacl_a_00067
30. Domingo, M., Garcıa-Martınez, M., Helle, A., Casacuberta, F., Herranz, M.: How much does tokenization affect neural machine translation? arXiv e-prints arXiv:1812.08621 (2018)
31. Araabi, A., Monz, C.: Optimizing transformer for low-resource neural machine translation. In: Proceedings of the 28th International Conference on Computational Linguistics, International Committee on Computational Linguistics, Barcelona, Spain (Online), pp. 3429–3435 (2020). https://doi.org/10.18653/v1/2020.coling-main.304. https://aclanthology.org/2020.coling-main.304

32. Mizumoto, T., Komachi, M., Nagata, M., Matsumoto, Y.: Mining revision log of language learning SNS for automated Japanese error correction of second language learners. In: Proceedings of 5th International Joint Conference on Natural Language Processing, Asian Federation of Natural Language Processing, Chiang Mai, Thailand, pp. 147–155 (2011). https://aclanthology.org/I11-1017
33. Çetinoğlu, Ö., Schulz, S., Vu, N.T.: Challenges of computational processing of code-switching. In: Proceedings of the Second Workshop on Computational Approaches to Code Switching, Austin, Texas, p. 1. Association for Computational Linguistics (2016). https://doi.org/10.18653/v1/W16-5801. https://aclanthology.org/W16-5801