SenTekki: Online Platform and Restful Web Service for Translation Between Wolof and French

Alla Lo[1], Elhadji Mamadou Nguer[2(✉)], Sileye O. Ba[3], Cheikh M. Bamba Dione[4], and Moussa Lo[2]

[1] Université Gaston-Berger, Dakar, Senegal
[2] Université Virtuelle, Diamniadio, Senegal
{elhadjimamadou.nguer,moussa.lo}@uvs.edu.sn
[3] Dailymotion, Paris, France
sileye.ba@dailymotion.com
[4] University of Bergen, Bergen, Norway
dione.bamba@uib.no

Abstract. In this article, we propose a machine translation Web Service and Platform for Wolof, a low-resource Niger-Congolese language. We first developed a Transformer model, and different configurations adapted to these models to translate between Wolof and French. We trained the model on a very limited amount of French-Wolof parallel data of approximately 85,000 parallel sentences. The best model was able to achieve a good performance of around 28 BLUE score. We then developed a web platform that allows text to be translated between French and Wolof, but also a RestFull Web Service that will allow other applications to benefit from the translation service.

Keywords: Neural machine translation platform · Neural machine translation web service · Low resource language · Wolof

1 Problem Statement

These days, web applications are changing individuals' life everywhere. Applications such as Google-Map permit individuals to find themselves. Facebook, Snapchat, Instagram permit people to be associated with their companions. Amazon, Alibaba permit individuals to do easy internet based shopping. Netflix permits individuals to watch video substance. For smooth communication, these applications require an understanding of web predominant dialects like English, Spanish, Chinese, Arabic, French, etc. Others dialects, spoken in Africa, for example, Eastern Ethiopian Amharic, Southern African Swahili, Western African Fulani or Wolof can not be utilized with these applications. The fundamental reason being that contrasted with English, for example, large numbers of these dialects are spoken by little sized populations.

A. D. Mambo et al. (Eds.): InterSol 2022, LNICST 449, pp. 290–298, 2022.
https://doi.org/10.1007/978-3-031-23116-2_25

In machine learning, natural language processing models have been created to extricate pertinent information from electronic documents on the web. These models for the most part address documents written in well-resourced languages like English, Mandarin, Spanish, Arabic, French, and so on. Aside from rare cases, low-resource language, like Eastern Ethiopian Amharic, Southern African Swahili, Western African Fulani or Wolof, are seldom addressed. Having machine learning models such as machine translation frameworks that can change data from high to low resource languages would have incredible effect. This will permit population communicating in low-resource dialects to make full utilization of web substance and applications.

This is the reason why we conducted some researches in building neural system for translation between Wolof and French language. This research led to the construction of a translation model between Wolof and French based on the architecture of the Transformer. This is a model that performed well for a low-resourced language with more than 28 BLUE points.

However, for this model to benefit the public, it must be accessible and usable via the Internet. It is with this in mind that we have started the development of a translation platform accessible online. We have also developed a RESTful web service to allow other applications to consume the services of the model to have text translation between Wolof and French.

The remainder of this paper is organized as follows. Section 2 discusses related work . Section 3 presents the translation platform we developed. Section 4 gives details about the Neural machine translation model we have investigated so far. Section 5 concludes the discussion and outlines future work.

2 Related Work

Because of the importance of language in human interaction, natural language understanding has been widely investigated in computer science. Natural language understanding can be approached either from speech or text inputs. In this paper our focus is on text inputs. Without being exhaustive, natural language understanding can have as object parts-of-speech tagging, named entity recognition, text classification, automatic translation, etc. [1,2].

Many applications in these aforementioned areas are developed for Western languages. However, for poorly resourced languages such as local African languages, few investigations have been conducted so far. To our knowledge, [3–6] are among the very few studies that have so far explored that language using neural network-based methods. These investigations led to the creation of an automatic translation model between French and Wolof [5]. There are other translation platforms between French and Wolof. However, it is not machine translation strictly speaking. They just put a list of French sentences with their Wolof translations. This kind of system will not be able to translate a sentence which is not part of the pre-recorded sentences.

The purpose of this paper is to provide a platform and a service to address the aforementioned model [5]. So far, investigations about Wolof have mostly been about building dictionaries [7], and studying the grammatical structures [8–10] of the language. Other African languages such as Swahili (Southern Africa) and Amharic (Eastern Africa) have been subject of research about automatic statistical based machine translation [11–13]. However, to our knowledge, this work (our Neural Machine Translation model) is the first to address Wolof (Western Africa) using Neural Network based Machine Translation.

3 Machine Translation Platform and RestFull Web Service

In this section we will present the global architecture of our system. It is composed of three main modules. Namely a web and mobile client, a web server, a REST API, and a translation model. Figure 1 presents the overall architecture of the application.

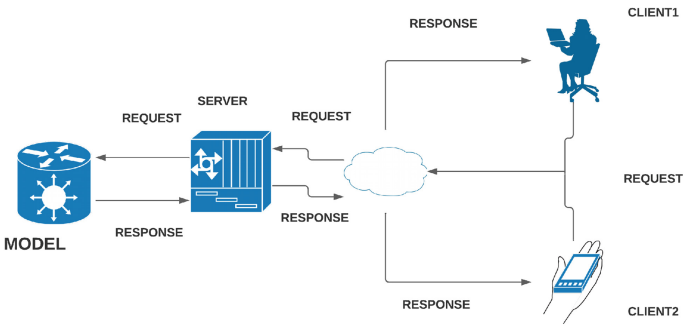

Fig. 1. Overall architecture

In general, the scenario is as follows: a web client sends text to be translated to the web server. The server retrieves this text, performs some processing before sending it to the translation model. The model translates the text and sends it back to the server. The server returns the result to the client. We will detail each module of the system to explain its real operation.

3.1 Client

The client designates the person who will request a translation of a text. It sends the text to be translated via the internet, this text ends up in the server which will do the necessary work. We have two types of client as described below.

3.2 Web and Mobile Client

The web client has a simple interface in which he can choose the languages and the direction of the translation (eg., French to Wolof). He has a text zone where he must type the text to be translated before clicking on the translate button. Once he clicks on the translate button, the result of the translation will be displayed to him in the text area located at the bottom of the said button. It also has a button to download the translation results in .txt format (see Fig. 2).

The same principle is used for the mobile client with a similar interface, but the mobile interface do note allow to download the results of the translation into a file (Fig. 3).

Fig. 2. Web client interface

3.3 Application Client

In the case where the client is an application, it sends the text to be translated into the provided endpoint. The text must be in JSON format where we have key **sent** and the value is the list of sentences to be translated.

Example:

```
texte = {
         "sent" : ["Je veux aller à Dakar demain",
                   "Je dois acheter une voiture"]
        }
```

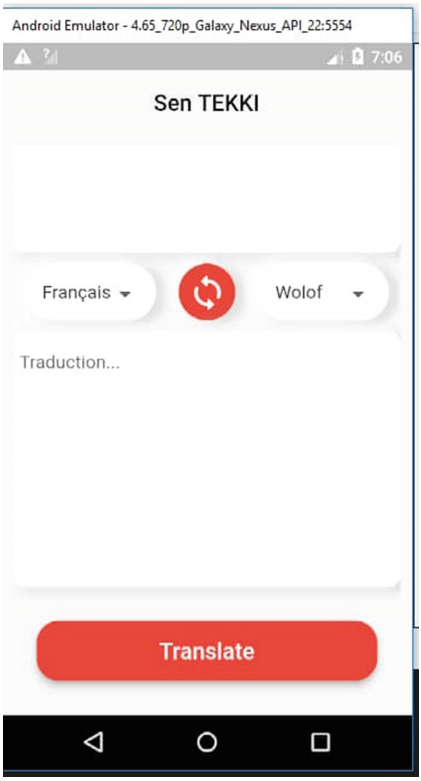

Fig. 3. Mobile client interface

3.4 Server

In our server we have two main elements. A Web App and a Rest API. We will explain both components in the following.

Web Server. The web server receives a request from the client, extracts the content of the request (the text to be translated) and gives it to the tokenizer. The latter cuts the text into a list of sentences and then send them to the translation model. The model does the translation and sends the results back to the server, which then will send it back to the client. The server will also at the same time keep the client's sentences and their translation in a database (see Fig. 4).

RESTfull Web Service. To access the API, the client application must send a JSON file to an URL given by the web service. The JSON data must have as **key** *sent* and as value the list of sentences to translate.

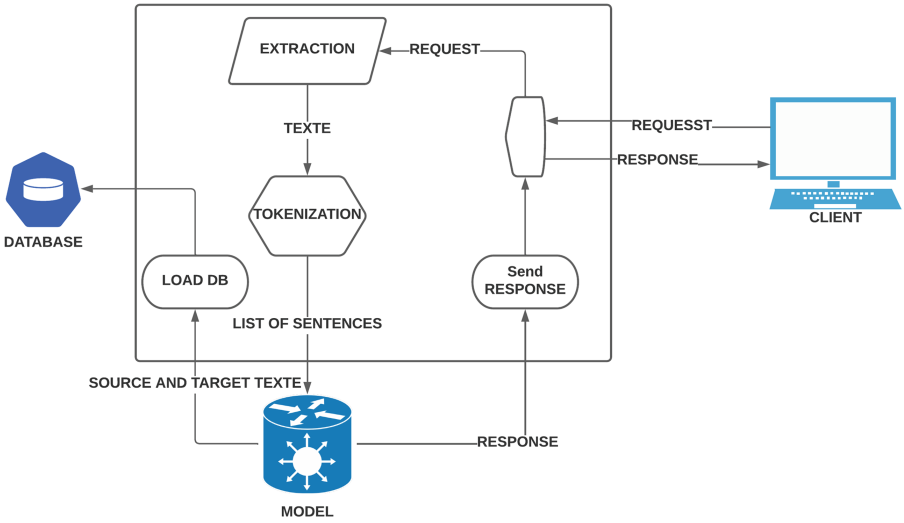

Fig. 4. Web application server side

```
url='http://localhost:5000/tekki'
sentences = str(['Donne-moi une minute.',
    'Dites-moi où il se trouve !',
    "D'accord, je pense que j'ai capté.",
    "Mais les enfants de Coré ne moururent point.",
    "Le garçon se mit à pleurer.",
    "J'imagine que Tom a raison.",
    "Je suis également à la retraite.",
    "Elle mourut de la tuberculose.",
    "Jésus leur répondit: Vous croyez maintenant?",])

res = requests.post(url, data= {"sent":sentences}).json()
```

The API receives the JSON data, extracts the sentences and sends it to the translation model. The latter translates the sentences and returns the result. They will be returned in JSON format before being sent back to the client application.

4 The NMT Model

The Transformer is a new architecture introduced by the article *Attention Is All You Need* [14]. This architecture transforms one sequence into another using a block of encoders and a block of decoders. Encoder block and decoder block are multiple identical encoders and decoders stacked on top of each other.

Encoder and decoder have the same number of units. Each encoder consists of two layers: self-attention and Feed Forward Neural Network. The first (and

only the first) encoder in the encoder block and the first decoder in the decoder block start with an embedding layer. Word embeddings from the input sequence are passed through the first encoder layer of the encoder block. It first passes through the layer of self-attention. This layer helps the encoder look at other words in the input sentence while it is encoding a specific word. The output of self-attention is then given to the FFNN layer. A normalization layer is used after the self-attention layer and after the Feed Forward Neural Network layer. The encoder output is then propagated to the next encoder layer. The output of the last encoder in the encoder stack is passed to all decoders in the decoder stack.

Self-attention is calculated using three vectors: *query (Q), key (K), value (V).* This vector is trained and updated during the training process. Self-attention is calculated for each word of the input sentence using these three vectors according to this formula:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}}) \tag{1}$$

.

Like the encoder, each decoder in the decoder block has a self-attention layer and an anticipation neural network layer. But between these layers is an attention layer that helps the decoder focus on the relevant part of the input sentence.

Our transformer follows the architecture proposed by [14]. It has a block of 3 encoders and a block of 3 decoders which process source sequences and target sequences respectively (see Fig. 5).

During encoding, each input word is transformed into a vector using an embedding layer. Next, positional encoding is used to inject positional information into the input embeddings. Each encoder uses two layers to convert the input into a continuous representation with attention information: multi-head attention and a position-dependent feedforward neural network (FFN). Multi-headed attention uses the self-attention mechanism, which allows models to associate each word in the input with other words. Self-attention is achieved by first creating the query (**Q**), key (**K**), and value (**V**) vectors from the input. Next, we calculate a score matrix by multiplying the query with the key vector and dividing by the square root of the dimension of the key vectors (denoted by d_k).

We apply softmax on the scaled score matrix to obtain the attention weights which are used to obtain an output vector. Adding this last vector to the original positional input integration creates a residual connection. The residuals pass through layer normalization and are projected through the point FFN, i.e. a few linear layers with ReLU activation in between. The output of this is then again added to the input of the point lookahead network and further normalized. The point anticipation layer is used to project attention outputs potentially giving it a richer representation.

$$Warning(Q, K, V) = softmax(\frac{Q \times K^T}{\sqrt{d_k}}) \tag{2}$$

The decoder has two multi-head attention blocks in a layer, one for the target sequences and one for the encoder output.

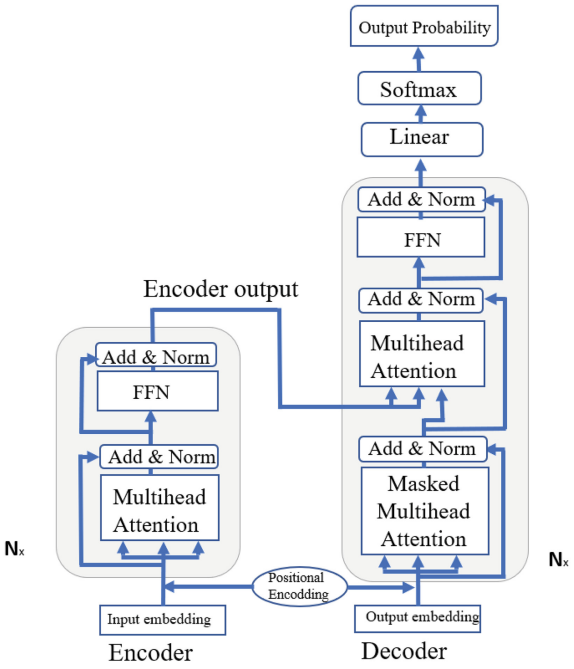

Fig. 5. Our basic transformer model

The old multi-headed attention is hidden to prevent the calculation of attention scores for future words. The decoder also has a point lookahead layer, residual connections, and layer normalization after each sublayer. As with encoding, during decoding the input goes through an integration layer and a positional encoding layer to achieve positional integrations. Then, these integrations are fed into the first multi-head attention layer which computes the attention scores for the decoder input. The second multidirectional attention layer uses the encoder outputs as requests and keys, and the first multidirectional attention layer outputs are the values. This process matches the encoder input to the decoder input, allowing the decoder to decide which encoder input to focus on. The output of the second multidirectional attention passes through a point FFN layer for further processing. The output of the final point lookahead layer goes through a final linear layer, which acts as a classifier.

5 Conclusion

In this article, we presented the investigations we carried out to build a translation platform between Wolof and the French language. The system also provides a RestFull web service that allows other applications to use the translation service. The overall architecture can be described as a client-server application. The application provides a user-friendly interface that allows a user to type text in

the source language and simply click a button to get the translation. The interface gives the possibility to the user to download the results of the translation in text format. So this work resulted in the first Wolof translation platform, with a model that achieved encouraging results for a language so poorly endowed with resources. The platform also allows at the same time to increase our corpus because keeping all the translations that it will have to operate in a database.

In the rest of our work, we plan to add a feature that allows users to correct the translation of the machine in order to allow it to make fewer mistakes the next time. This corrected data will be added to our corpus to re-train the translation model.

Acknowledgments. Authors thank the CEA MITIC of Universite Gaston Berger in Senegal for partly funding this work.

References

1. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of the ICML (2008)
2. Young, T., Hazarika, D., Poria, S., Cambria, E.: Recent trends in deep learning based natural language processing. arXiv:1708.02709 (2018)
3. Alla, L., Sileye, B., El Hadji Mamadou, N., Moussa, L.: 2019 Neural words embedding: Wolof Language Case. In: IREHI 2019 (2019)
4. Alla, L., Bamba, D.C., El Hadji Mamadou, N., Sileye, B., Moussa, L.: 2020 building word representations for Wolof using neural networks. In: CNRIA 2020 (2020)
5. Alla, L., Sileye, B., El Hadji Mamadou, N., Moussa, L.: 2019 Neural words embedding: Wolof Language Case. In: ICLR workshop 2020 (2020)
6. Bamba, D.C.: LSTM based Language models for wolof. human language technologies as a challenge for computer science and linguistic. 190–194 (2019). 978–83-65988-30-0
7. Khoule, M., Thiam, M.N., Nguer, E.M.: Towards the establishment of a LMF-based Wolof language lexicon. Traitement Automatique des Langues Africaines (TALAf) (2014)
8. Dione, C.B.: LFG parse disambiguation for Wolof. J. Lang. Model. **2**, 105 (2014)
9. Dione, C.B.: Valency change and complex predicates in Wolof: an LFG account. In: LFG Conference (2013)
10. Dione, C.B.: An LFG approach to Wolof cleft constructions. In: LFG Conference (2012)
11. Pauw, G.D., Wagacha, P.W., Schryver de, G.-M.: Towards English - Swahili Machine Translation. In: Research Workshop of the Israel Science Foundation (2011)
12. Ombui, E.O., Wagacha, P.W., Ng'ang'a, W.: InterlinguaPlus machine translation approach for under-resourced languages: Ekegusii and Swahili. In: Workshop on the Use of Computational Methods in the Study of Endangered Languages (2014)
13. Gebreegziabher, M., Besacier, L.: English-amharic statistical machine translation. In: Workshop on Spoken Language Technologies for Under-Resourced Languages (2012)
14. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, A.: Attention is all you need. arXiv:1706.03762v5 [cs.CL] (2017)