The Development of Sepedi Text Generation Model Using Transformers

Simon Ramalepe
University of Limpopo
4 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Marelie Davel
North-West University
137 PUBLICATIONS **1,499** CITATIONS
SEE PROFILE

The Development of a Sepedi Text Generation Model Using Transformers

Simon P Ramalepe*, Thipe I Modipa*‡ and Marelie H Davel†‡
*Computer Science Department, University of Limpopo, Sovenga, 0727, South Africa.
†Faculty of Engineering, North-West University, Potchefstroom, South Africa.
‡Centre for Artificial Intelligence Research (CAIR), South Africa.
simon.ramalepe@ul.ac.za
thipe.modipa@ul.ac.za

***Abstract*—Text generation is one of the important sub-tasks of natural language generation (NLG), and aims to produce humanly readable text given some input text. Deep learning approaches based on neural networks have been proposed to solve text generation tasks. Although these models can generate text, they do not necessarily capture long-term dependencies accurately, making it difficult to coherently generate longer sentences. Transformer-based models have shown significant improvement in text generation. However, these models are computationally expensive and data hungry. In this study, we develop a Sepedi text generation model using a Transformer-based approach and explore its performance. The developed model has one Transformer block with causal masking on the attention layers and two separate embedding layers. To train the model, we use the National Centre for Human Language Technology (NCHLT) Sepedi text corpus. Our experimental setup varied the model embedding size, batch size and the sequence length. The final model was able to reconstruct unseen test data with 75% accuracy: the highest accuracy achieved to date, using a Sepedi corpus.**

***Index Terms*—Transformers, Generative pre-trained Transformer, Natural Language Generation, Text generation**

I. Introduction

Sepedi is one of the official languages of South Africa and the first language of 9.1% of the South African population [1]. It is largely spoken in the Limpopo province and some parts of the North Eastern region of South Africa. Sepedi is classified as an under-resourced language and currently there is no Transformer-based text generation model for Sepedi. Text generation models have been developed for well-resourced languages like English using the Transformer-based approach [2], [3].

Natural language generation (NLG) is a domain of artificial intelligence that is aimed at producing understandable natural language text [4]. The goal of NLG is to construct software systems that can coherently generate readable text in different applications, such as question and answering systems (including chatbots), automating the generation of storytelling, document summarisation, translating text, and improving the efficiency of paper writing in research and other professional writing [5]. Text generation, as a sub-task of NLG, is the process of automatically generating text given some input text. It is accomplished through the training of a language model on a large corpus using machine learning techniques to find the probability of the next word in a sequence of words [6]. That is, given

$$P(Y|X) = P(y_1, ..., y_j, ...y_n|X)$$

the text generation model generates y_j as a sequence of discrete tokens generated from the vocabulary conditioned on the input X.

Until recently, recurrent neural networks (RNNs) [7] and long short-term memory (LSTM) [8] networks have been the dominant approaches in NLG tasks. These models have been used to successfully implement text generation models. However, a significant drawback of RNNs is that they struggle to accurately capture long-term dependencies i.e, the probability of remembering words and maintaining context as the sentence gets longer diminishes due to either vanishing or exploding gradients [9], [10]. LSTM on the other hand, has the capability of learning and remembering words more than one thousand timestamps backwards hence it was deemed a solution to the problem of vanishing or exploding gradients [10].

Recently, the Transformer [11] has emerged as the *de facto* architecture for natural language processing tasks [12]. Models based on the Transformer architecture have shown significant improvement in NLG tasks like text generation and text translation [13]. Hence, in this study, we develop a Sepedi text generation model using the Transformer-based approach and explore the model's performance on a Sepedi text corpus.

The paper is structured as follows: Section II discuss background to the study. Section III discusses the dataset used for training the text generation model. In Section IV we discuss the experiments conducted, while in Section V we focus on the results obtained and the evaluation thereof. Concluding remarks are made in Section VI.

II. Background

Machine learning focuses on the development of algorithms that are able to learn, adapt and improve automatically through training [10]. That is, the more the algorithm is trained, the better it becomes. Previous algorithms for modelling natural language generation tasks were mainly based on RNNs. These algorithms were good at modelling sequential data [14]. Unlike the recurrence technology used in RNNs, which requires data or sequences to be processed in sequence, the

Transformer-based architecture uses self attention and a multi-head mechanism to circumvent the recurrence approach and to capture long-term dependencies. This mechanism allows the Transformer to be able to attend to information from different representations. The process is achieved by calculating the output score as a weighted sum of the values for each input sequence by multiplying the key (K), query (Q), and value (V) vectors for each embedding word using the attention function [11].

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

where d_k is the input dimension vectors of the keys.

To ensure that the model attends to information from different representations, the Transformer model uses multi-head attention mechanism where the key, query, and value vectors are projected linearly and the attention function is applied in parallel to obtain the value dimensions which are then concatenated to obtain the final value output. Multi-head attention is regarded as the main component of the Transformer architecture which calculates the contextual representations of each word by considering the entire input sequence at different sub-spaces [15]. It is defined as

$$Multi(Q, K, V) = concat(Head_1, Head_2, ..Head_n)W^O$$

$$\text{where } Head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

with QW_i^Q, KW_i^K, VW_i^V as the learned projection matrices and $W^0 \in \mathbb{R}^{d \times d}$ as the projection matrix for the output. Transformer-based models like generative pre-trained Transformer-3 (GPT-3) [16] and its earlier versions use this technique. The Transformer is basically meant to transform one sequence to another sequence using either the encoder or decoder part the architecture [3].

GPT is an autoregressive language model which uses the decoder part of the Transformer architecture by stacking multiple Transformer decoder layers with masked self attention heads. This language model has been trained to predict the next word x_i given the previous words $x_1, x_2..., x_{i-1}$ [17]. Its training objective is to maximise the log-likelihood

$$\sum_i log(P(x_i|x_1, x_2, ..., x_{i-1})); \theta T)$$

where θT are the model parameters. Fig. 1 shows the decoder part of the Transformer with just a single layer of Transformer block. GPT-2 small has 12 Transformer layers.

Pawade et al. [18] developed a story scrambler using a RNN-LSTM approach to generate plausible text. In their approach, several parameters like model size, number of layers, batch size and sequence length were varied during training in order to optimise the training loss. Their optimal value for the training loss was 0.01, obtained with a model size of 512, number of layers equal to 3, a batch size of 100 and a sequence length of 50. The evaluation of the performance of their model included human evaluation. Factors like grammar correctness, linkages of events, level of interest and uniqueness in the generated stories were rated by human evaluators and stories with an average score of 60% and higher in all aspects were accepted. Their model recorded an accuarcy of 63%. Moila and Modipa [19] also used a RNN-LSTM to develop a Sepedi text generation model. Their model produced an accuracy of 50.3% with limited data.

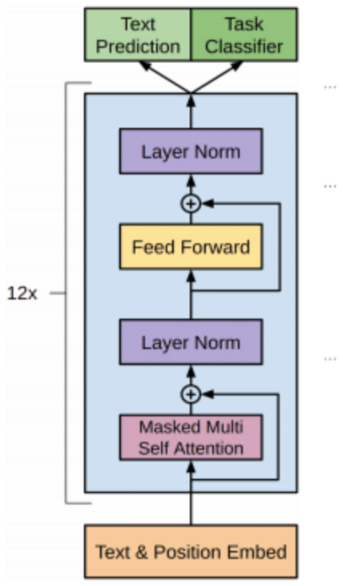

Fig. 1:
Decoder-only architecture used by GPT-2. Source: [20]

Du et al. [3] developed an automatic text generation model using deep learning approaches for providing large-scale support for online learning communities. In their approach, RNN and GPT-2 language models were trained with over two million comments from online communities. To assess the performance of their model, the generated text was measured through a readability test and human evaluation. Their study found that GPT-2 could generate more human-like written text than an RNN. In another study, Keh and Cheng [2] used Bidirectional Encoder Representations from Transformers (BERT), a pre-trained language model, to develop a model for personality classification and specific language generation. Their model recorded the lowest loss of 0.02 with an accuracy of 47% with just 10 epochs. Their experimental setup used a batch size of 16, a learning rate of 3.10^{-5}, max sequence length of 128, and a warm-up proportion of 0.1.

Transformer-based models have performed better in well-resourced languages like English, however, it still remains to be seen how they will perform in low-resourced languages, hence the choice of the Transformer approach in this study. Unlike [3] and [2] who used pre-trained Transformer models (GPT and BERT) in their studies on English corpus, this study trains the Transformer model from scratch using a Sepedi corpus.

III. Data collection

The data for this study was obtained from the National Centre for Human Language Technology (NCHLT) project. The dataset is composed of a collection of several South African government entities crawled from gov.za websites and collected from various language units from 2007 to 2011 [21]. The Sepedi text corpus has 69 254 rows of text with about 2.1 million tokens. Table I shows the size of the selected dataset with a split of 80% for training, 20% validation.

TABLE I: The number of sentences, words and unique tokens in the dataset.

Dataset	**Training**	**Validation**	**Total data**
Sentences	55 404	13 850	69 254
All words	1.5m	550k	2.1 mil
Unique tokens	50k	10k	50K

Fig. 2 shows a bar graph of the top 30 word frequencies in the train dataset. The y-axis depicts the frequency of each word in the dataset while the x-axis shows the words in the vocabulary sorted from most to least frequent. It is observed that bi-gram words have a high frequency in the Sepedi language.

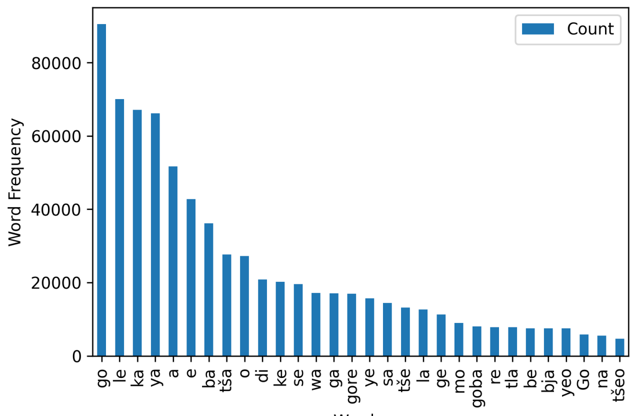

Fig. 2:
Word frequency in the train dataset

In Table II we notice that the word *go* has a high frequency of occurring before the word *ya* and the word *le* has high frequency of occurring before the word *go*. Importantly we note that only 5% of the words in Table II are uni-grams, 42.5% are bi-grams and 52.5% are tri-grams and higher. This shows that most bi-grams are largely followed by words with higher n-grams in the Sepedi language.

TABLE II: For each of the top 2 words, the 20 most frequent words that occur after them.

	Word	**Occurrence**		**Word**	**Occurrence**
go	ya	3 463	le	go	7 726
	ba	2 703		gore	1 635
	tšwa	2 030		ge	1 573
	se	1 684		ka	1 572
	na	1582		le	1 191
	dira	1 390		gona	1 003
	swana	1 271		ye	856
	feta	1 212		tša	645
	ka	1 019		ya	599
	tloga	880		lengwe	524
	le	823		a	493
	hwetša	772		ba	481
	šoma	754		bjalo	434
	tla	655		yena	421
	fihla	654		batho	362
	tšweletša	623		se	352
	fa	587		yo	347
	be	586		tše	345
	netefatša	551		bona	342
	e	518		yona	331

IV. Experimental Setup

Our experimental setup followed the approach of Pawade et al. [18] and Keh and Cheng [2]. To develop the model itself, we adopted the approach by Nandan [22] for developing a GPT model using a deep generative approach in Keras[1]. The developed GPT model has one Transformer block with causal masking on the attention layers, two separate embedding layers for tokens and a token index with one dense layer with 2 heads. The vocabulary size was set to 50K, the maximum size of unique tokens in the dataset. The model embedding size was varied at 32 and 64 to investigate the effect of these complexity hyperparameters on the developed model. Eight experiments were conducted, four for each model size varied by batch size and sequence length (see Table III). We used Adamax as the model optimiser and the rectified linear unit (ReLU) activation function. The accuracy of the model was measured by computing the total number of correct predictions as a percentage of the total number of predictions during training using the *SparseCategoricalAccuracy* function metric which is used mostly when making text prediction for sparse target (a condition where the yTrue is a huge metric that is almost all zeros) in deep learning. *SparseCategoricalAccuracy* metric checks if the maximum true value is equal to the index of the maximum predicted value. The model learning rate was kept constant at 10^{-3} throughout the experiments.

TABLE III: Loss and accuracy observed for different experimental setups.

Exp setup	**Model size**	**Batch size**	**Max length**	**Loss**	**Accuracy**
1	32	128	80	1.728	0.657
			100	1.659	0.678
		256	80	1.758	0.656
			100	1.516	0.747
2	64	128	80	1.101	0.735
			100	1.055	0.748
		256	80	1.322	0.708
			100	1.019	0.755

[1]https://keras.io/

The train loss and accuracy were recorded at every epoch. The following parameters were varied at different levels of training: embedding size, batch size and maximum sequence length. The embedding size defines the complexity of the model, the batch size controls the number of samples the model should work through before the internal parameters are updated, and the sequence length controls the length of the tokens generated before padding. The model was run for 200 epochs. For regularisation, we first used the Transformer default dropout of 0.1. The dropout was gradually increased to 0.4 while monitoring the model's performance. The model generalised better with a dropout of 0.4. The activation function and the optimiser were kept the same throughout the experiments.

V. Results and Evaluation

Fig. 3 shows the loss curve (loss value at each epoch of the training process) for experiment setup one with model size of 32, while Fig. 4 shows the loss curve for experiment setup two with model size of 64. In both cases, the single best model is selected. Both the training and validation loss curves seem to converge in Fig. 3, however, this is not the case in Fig. 4 where the validation error at 200 epochs is higher than the training error, giving a possible indication of overfitting. The optimal number of iterations in Fig. 4 was between 100 to 125 epochs.

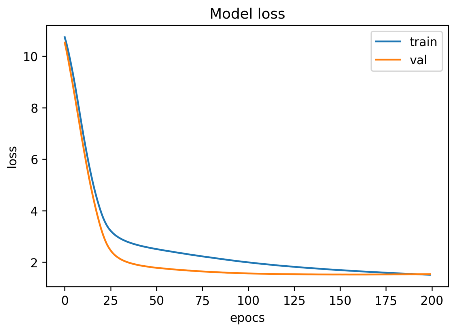

Fig. 3: Loss curve: model embedding size 32.

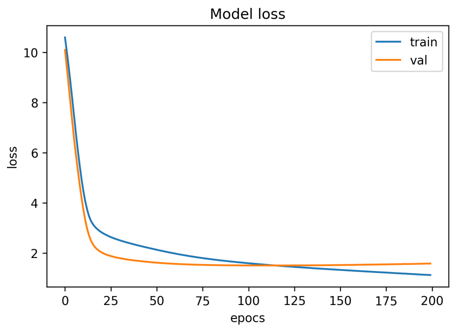

Fig. 4: Loss curve: model embedding size 64.

The developed Transformer model achieved an accuracy of 75%, which is a significant improvement over the RNN-LSTM accuracy of 50.3% achieved in [19] on a similar task.

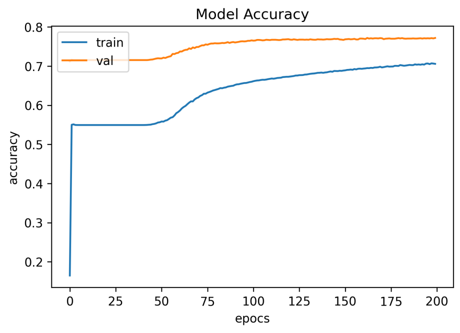

Fig. 5: Accuracy curve: model embedding size 32.

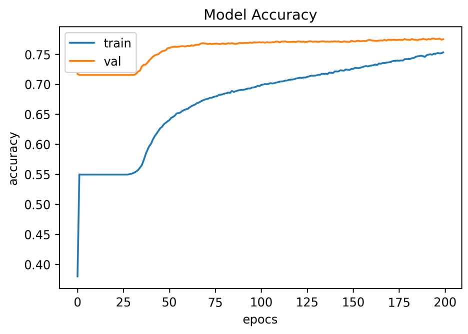

Fig. 6: Accuracy curve: model embedding size 64.

(The 63% achieved in [18] and 47% achieved with a pre-trained Transformer model in [2] are not directly comparable.) The optimal accuracy in our model was achieved with model embedding size 64: the change in accuracy during training for this model is shown in Fig. 6. Although the graph in Fig. 3 for model size 32 converges at 200 epochs, the accuracy (Fig. 5) still seems to be improving, although very slowly. An increased number of epochs could have helped to improve the accuracy, however from Fig. 5, it can be observed that the improvement is expected to be small.

Considering the amount of data we had in the dataset, the performance of the model was satisfactory. However, in both experiments, we observe that the validation accuracy is higher than training accuracy. This can occur for more than one reason: either the validation set is in some way easier to model than the training data, or the model is underfitting. The latter is a possibility since the system has only one dense layer with two heads. We expect that, with the addition of more dense layers and an increase in a number of heads, the model performance can improve.

In order to demonstrate performance of the model in practice, we show a random text sample generated by the model. We supplied the model with a seed of 10 words and the model generated the text in Table IV. The first row is the original text generated. We then corrected the grammar in the second row and manually translated the text in the third row. Given the amount of data we had and limited computational resources, the generated text was viewed as a preliminary success. The results shows that even though the word frequency in Fig. 2 shows that bi-grams occur the

TABLE IV: Generated text, corrected version and the English version

molaotheo wa naga o swanetše go fetolwa gore badudi ba naga ya mohlagase wo o bego e tlago ka kgwedi ya ikonomi yeo e swanago le seripa ya setšhaba , le go phethagatša leano la selegae , leo go hwetša temokrasi di hlokegago go kaonafatša karolo ya ka gare ga mengwaga ye , go tloga ka se , le seripa . mabapi le go aga lefase ya mešomo ye go batho ba tla phethagatša ka ngwaga wa go akaretšwa go hlola lenaneo la mebušo ya lefase . mabapi le go re tla ba go kaonafatša mokgwa wa go ba go ba tšea swana
molaotheo wa naga o swanetše go fetolwa gore badudi ba naga ya mohlagase wo o bego o etla ka kgwedi ya ikonomi yeo e nago le seripa **sa** setšhaba , le go phethagatša leano la selegae , **le** go hwetša temokrasi. **se se** hlokegago **ke** go kaonafatša karolo ya ka gare ga mengwaga ye , go tloga ka se , le seripa **se** mabapi le go aga lefase **le** mešomo ye batho ba tla phethagatšago ka ngwaga go akaretšwa go hlola lenaneo la mebušo ya lefase mabapi le go kaonafatša mokgwa wa go tšea swana
The constitution of the country should change so that citizens of the country that electricity which used to come in the economic month which has part within the community, should complete the local plan and to get democracy. what is is missing is to make better the parts within this years from this and this part in relation with building the world and employments which people should work in a year including implementation of the world governance and complete the behaviour of being the same.

most because of the disjunctive nature of Sepedi language, they are mostly followed by words with higher n-grams (Table II) which are then used to generate sentences that are approximately grammatically correct.

VI. Conclusion

The intent of this research was to explore the performance of a Transformer-based model on the disjunctive under-resourced language, Sepedi. Our aim was to develop and train the model rather than to use a pre-trained model. The developed model was trained by varying the embedding size, batch size and sequence length in order to observe the effect that model complexity has on performance. The model achieved an accuracy of 75% without human evaluation, which is the highest accuracy reported for Sepedi to date. The inclusion of human evaluators could have given us a better analysis of model performance: this remains future work.

From our experimental results, it was observed that the model size had an impact on the accuracy of the model. When the model size was lower the accuracy was lower. The dataset consisted of about 2.1M tokens and we could only run up to 200 epochs per training run. Increasing the model size and number of epochs with the same dataset is not expected to improve the quality of the generated text significantly but would result in the model being more complex for the data that we had. For future work, we aim to experiment with more data, larger models and longer training runs and to compare the performance of these models with models developed for other under-resourced languages.

References

[1] P. Lehohla, "Census 2011 Census in brief," Statistics South Africa., Pretoria, Tech. Rep., 2012.

[2] S. S. Keh and I.-T. Cheng, "Myers-Briggs Personality Classification and Personality-Specific Language Generation Using Pre-trained Language Models," in *arXiv preprint arXiv:1907.06333.*, 7 2019.

[3] H. Du, W. Xing, and B. Pei, "Automatic text generation using deep learning: providing large-scale support for online learning communities," *Interactive Learning Environments*, 2021.

[4] R. Perera and P. Nand, "Recent Advances In Natural Language Generation: A Survey And Classification of the Empirical Literature," *Computing and Informatics*, vol. 36, pp. 1–32, 2017.

[5] A. Celikyilmaz, E. Clark, and J. Gao, "Evaluation of Text Generation: A Survey," in *arXiv preprint arXiv:2006.14799*, 6 2020.

[6] J. Li, T. Tang, W. X. Zhao, and J.-R. Wen, "Pretrained Language Models for Text Generation: A Survey," in *arXiv preprint arXiv:2105.10311.*, 5 2021.

[7] R. J. Williams and D. Zipser, "A Learning Algorithm for Continually Running Fully Recurrent Neural Networks," *Neural Computation*, vol. 1, no. 2, pp. 270–280, 1989.

[8] S. Hochreiter and J. Urgen Schmidhuber, "Long Short-Term Memory," in *Neural computation.*, vol. 9, no. 8, 1997, pp. 1735–1780.

[9] S. Lu, Y. Zhu, W. Zhang, J. Wang, and Y. Yu, "Neural Text Generation: Past, Present and Beyond," in *arXiv preprint arXiv:1803.07133.*, 3 2018.

[10] R. C. Staudemeyer and E. R. Morris, "Understanding LSTM – a tutorial into Long Short-Term Memory Recurrent Neural Networks," in *arXiv preprint arXiv:1909.09586*, 9 2019.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," *In Advances in neural information processing systems.*, pp. 5998–6008, 6 2017.

[12] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "HuggingFace's Transformers: State-of-the-art Natural Language Processing," *arXiv:1910.03771.*, 10 2019.

[13] H. Li, Y. Wang, Y. Liu, D. Tang, Z. Lei, and W. Li, "An Augmented Transformer Architecture for Natural Language Generation Tasks," in *2019 International Conference on Data Mining Workshops (ICDMW).IEEE*, 2019, pp. 1–7.

[14] T. Iqbal and S. Qureshi, "The survey: Text generation models in deep learning," *Journal of King Saud University - Computer and Information Sciences*, pp. 2515–2528, 2020.

[15] C. Zhu, W. Ping, C. Xiao, M. Shoeybi, T. Goldstein, A. Anandkumar, and B. Catanzaro, "Long-Short Transformer: Efficient Transformers for Language and Vision," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17 723–17 736, 2021.

[16] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. Mccandlish, A. Radford, I. Sutskever, and D. Amodei, "Language Models are Few-Shot Learners," in *Advances in neural information processing systems.1877-1901.*, 2020, pp. 1877–1901.

[17] B. Min, H. Ross, E. Sulem, A. P. B. Veyseh, T. H. Nguyen, O. Sainz, E. Agirre, I. Heinz, and D. Roth, "Recent Advances in Natural Language Processing via Large Pre-Trained Language Models: A Survey," in *arXiv preprint arXiv:2111.01243*, 11 2021.

[18] D. Pawade, A. Sakhapara, M. Jain, N. Jain, and K. Gada, "Story Scrambler - Automatic Text Generation Using Word Level RNN-LSTM," *International Journal of Information Technology and Computer Science*, vol. 10, no. 6, pp. 44–53, 6 2018.

[19] M. M. Moila and T. I. Modipa, "The development of a sepedi text generation model using long-short term memory," in *Proceedings of the 2nd International Conference on Intelligent and Innovative Computing Applications*. New York, NY, USA: ACM, 9 2020, pp. 1–5.

[20] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever, "Improving Language Understanding by Generative Pre-Training," 2018.

[21] R. Eiselen and M. J. Puttkammer, "Developing Text Resources for Ten South African Languages," in *In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, 2014, pp. 3698–3703.

[22] A. Nandan, "Text generation with a miniature GPT," 5 2020. [Online]. Available: https://keras.io/examples/generative/text_generation_with_miniature_gpt/

Simon Ramalepe obtained his MSc in Computer Science at the University of Limpopo. He is currently studying towards his PhD degree. His research interest is in natural language processing and software engineering.

Thipe Modipa received his PhD (Information Technology) from the North-West University. His research interests include: code-switching for under-resourced languages, text generation, automatic speech recognition.

Marelie Davel is a Professor of Engineering at North-West University and the director of the *MUST Deep Learning* research group. She studies the theory and application of deep learning models.