NLAPOST2021 1st Shared Task on Part-of-Speech Tagging for Nguni Languages

*Pannach, Franziska**
Göttingen Centre for Digital Humanities
franziska.pannach@uni-goettingen.de

Meyer, Francois
University of Cape Town
francoisrmeyer@gmail.com

Jembere, Edgar
University of KwaZulu-Natal
Jemberee@ukzn.ac.za

Dlamini, Sibonelo Zamokuhle
University of KwaZulu-Natal
DlaminiS4@ukzn.ac.za

Abstract

Part-of-speech tagging (POS tagging) is a process of assigning labels to each word in text, to indicate its lexical category based on the context it appears in. The POS tagging problem is considered a mostly solved problem in languages with a lot of NLP resources such as English. However, this problem is still an open problem for languages with fewer NLP resources such as the Nguni languages. This is owing to unavailability of large amounts of labelled data to train POS tagging models. The rich morphological structure and the agglutinative nature of these languages make the POS tagging problem more challenging when compared to a language like English. With this in mind, we have organised a challenge for training POS tagging models on a limited amount of data for four Nguni languages: isiZulu, Siswati, isiNdebele, and isiXhosa.

Keywords: Shared Task, Competition, Part-of-Speech Tagging, Southern African Languages

1 Introduction

In this paper, we present the shared task and combined results of NLAPOST2021, Nguni Languages Part-of-Speech Tagging, hosted jointly by the Digital Humanities Association of Southern Africa Conference (DHASA) [1] and the Southern African Conference for Artificial Intelligence Research (SACAIR) [2].

The objective of the shared task was to invite researchers, students and other interested parties to provide systems that can reliably predict part-of-speech tags for isiNdebele, isiXhosa, isiZulu and Siswati. The motivation behind the organization of this shared task was three-fold: Firstly, we wanted to invite young researchers and students to participate in a Digital Humanities and Artificial Intelligence related conference, where they could showcase their expertise. Secondly, we wanted to utilize the newly published CTexT POS dataset [3]. Thirdly, we hoped our participants would achieve good results with diverse machine learning systems.

This paper is structured as follows: Section 2 contains a description of the data, Section 3 describes the shared task, and related work is presented in Section 4. The winning team's submission is introduced in Section 5, and the results are presented in Section 6. The paper is concluded by Section 7.

2 Data

The first shared task on Nguni Languages Part-of-Speech Tagging (NLAPOST) covered four different African languages: isiNdebele, isiXhosa, isiZulu and Siswati.

The data was provided to us by the Centre for Text Technology at the North-West University [4].

Each tab separated data file consists of text tokens, morphological analysis, lemma, treebank-specific part-of-speech (XPOS), and universal part-of-speech (UPOS). As of date, the publication describing the part-of-speech annotated data is yet to be published. Therefore, the authors have to refer to internal annotation protocols kindly provided to us by CTexT (CTexT 2020, Pienaar 2021).

Table 1 shows the number of tokens per language file. An example of the data format provided to the participants is shown in Table 2.

The shared task data consisted of columns for token, morphological analysis, and universal part-of-speech. The original dataset was split into training set (90 %) and test set (10 %) per language file.

3 Shared Task

The participants were asked to predict part-of-speech tags (UPOS) for all languages. They were provided with morphological segmentation, but not the full morphological analysis. The NLAPOST2021 shared task was published on codalab.com and announced on various mailing lists, social media channels, and grassroots research networks, such as Masakhane [5]. Participants were asked to register, after which they received the necessary information from the organizers. After an initial phase, in which the registered participants only had access to a small development set (two sentences per language), the training data was released. In total, participants had eight weeks to use the training data to develop their systems. Three weeks before the end of the competition, the test data was published.

Participants were free to make us of the morphological segmentation, but not required to do so. Furthermore, unified systems (one system for all languages) or individual systems (one system per language) were accepted. Participating teams were asked to submit individual files per language, containing only the token and the predicted UPOS tag.

Table 1: Size of Shared Task Dataset

	Tokens
isiNdebele	51,120
isiXhosa	49,104
isiZulu	50,166
Siswati	50,528

Table 2: Example Annotation (Siswati)

TOKEN	MORPH SEG	UPOS
Ngetulu	nga-tulu	ADV
kwaloko	kwa-loko	POSS
,	,	PUNC
kuba	ku-b-a	V
khona	khona	CONJ
kuniketela	ku-niket-el-a	V
kwekwakhiwa	kwe-ku-akh-iw-a	POSS
kwemaKomidi	kwe-ma-komidi	POSS
emaWadi	e-ma-wadi	N

Out of a number of registered participants, despite extension of the deadline for submitting systems, only one team submitted results (the submission presented in sections 5 and 6). However, the participating team delivered encouraging results across all four languages. Therefore, we invited the team to collaborate on this paper [6].

4 Related Work

A number of POS taggers have been developed over the years for poorly resourced agglutinative languages. The first reported work on POS tagging for the four Bantu languages we use for our shared task was done as part of a resource construction project for ten of South Africa's official languages (Eiselen & Puttkammer 2014). The open-source HunPOS tagger (Halácsy et al. 2007) was used on data for isiZulu, isiXhosa, isiNdebele, Siswati and achieved an accuracy of 83.83 %, 84.18 %, 82.57 %, and 82.08 % respectively.

Recently, Igbo, an agglutinative native language of Nigeria has been the subject of an effort to develop an effective POS tagger for the language (Onyenwe et al. 2019). A tagset of 70 tags was used to tag a combined corpus of 303 816 words. A wide range of POS tagging methods are used as a baseline in this study: unigram, ME, HMM, transformation-based learning and similarity-based learning. A rule-based algorithm is developed, which takes advantage of relatively accurate morphological analysis. Given the complexity of the morphosyntax of Igbo, the authors were able to produce specific rules which exploit this linguistic knowledge to produce results which are superior to the baseline models developed for a non-agglutinative language like English. This rule-based approach produces accuracies ranging from 82-100% on subsets of the aggregate corpus.

Bengali is the most spoken language in Bangladesh and the second most spoken language in India. Two models were developed for this agglutinative language, a hidden Markov model (HMM) and an Maximum Entropy (ME) tagger (Dandapat et al. 2007). A tag set of 40 tags was used to annotate 3625 sentences, which amounted to approximately 40 000 words. Both the HMM and ME models also integrated morphological information in their feature set, which significantly improved the accuracy of both models. In this study, it was the HMM supervised model which performed best, achieving an accuracy of 88.75%.

The HMM also performed well on another Indic language, Assamese (Saharia et al. 2009). This national language of India is spoken by approximately 30 million people. In this case a tagset of 172 tags was used to annotate a 10 000 word corpus for training and testing the corpus. Although no morphological information was used, the POS tagger was able to achieve an accuracy of 85.64%. This figure is difficult to put into context as the paper didn't report any results for other models on the same dataset.

As popular as the HMM and ME models are for POS tagging, other techniques have also been tried on the task. Conditional Random Fields (CRFs), Support Vector Machines (SVMs) and a rule-based approach were compared on Kokborok POS tagging (Patra et al. 2012). Kokborok is a language spoken by approximately 2.5 million native Indians based in the northern region of the country. A tagset of 26 tags was used to tag a corpus of 42 537 words. Morphological analysis was used to break the words down into morphemes so that features could be developed manually for the rule-based and machine learning approaches. Of the three approaches, the SVM model performed best, achieving an accuracy of 84.46%

Table 3: The different design choices we tried while developing our system. Our submission to the shared task is indicated in bold.

Component	Option	Label
Model	Bi-LSTM	lstm
	Bi-LSTM + CRF	**crf**
Features	Characters	char
	Character 2-grams	**2gram**
	Character 3-grams	3gram
Composition	**Sum**	**sum**
	Bi-LSTM	lstm

5 Methods

Our final submission to the shared task was a bidirectional LSTM (bi-LSTM) with a conditional random field (CRF) layer, using character 2-grams as input features. We chose this system as our final submission after experimenting with different design choices, as listed in table 3. In this section we present a baseline we initially developed to compare our subsequent systems to. We then go through the different components of our own system, and explain how we arrived at our final system.

5.1 Baseline

Our baseline system is a hidden Markov model (HMM) (Baum & Petrie 1966) using words as input features. A HMM is a statistical model which assumes that each observation in a sequence is produced by an unobserved *hidden* state. The hidden states follow a Markov process (the probability distribution of each hidden state depends only on the previous hidden state), and they in turn produce observations according *emission probabilities* that only depend on the current hidden state.

For our POS tagging baseline, we model the words in a sentence as the observed sequence $x_1, x_2, ..., x_n$ and their parts of speech as the hidden states $z_1, z_2, ..., z_n$. Training a HMM requires learning transition probabilities $p(z_t|z_{t-1})$ and emission probabilities $p(x_t|z_t)$. In our case we have a labelled training set available, so we train our model by simply counting transitions and emissions, and normalising them to obtain probabilities. To predict the POS tags of an unlabelled test sentence, we run the Viterbi algorithm, which computes the most likely hidden sequence given an observed sequence. It is a dynamic programming algorithm that computes hidden state sequence probabilities in a forward pass, and traces the most likely hidden state sequence in a backward pass.

5.2 System components

In developing our system we were faced with a number of decisions, regarding which methods to use in the components of our POS tagging system. Here we discuss the options we tried for the neural model, the input features, and word composition.

Neural models

Long short-term memory networks (LSTMs) (Hochreiter & Schmidhuber 1997) are recurrent neural networks for sequence modelling. At each step in a sequence, they update an internal hidden state vector through a number of learned *gates* that act as filters on the hidden state. The gates are computed from the current input vector $\mathbf{x_t}$ and the previous hidden state $\mathbf{h_{t-1}}$ as

$$\begin{aligned}\mathbf{f_t} &= \sigma(W_f\mathbf{x_t} + U_f\mathbf{h_{t-1}} + \mathbf{b_f})\\ \mathbf{i_t} &= \sigma(W_i\mathbf{x_t} + U_i\mathbf{h_{t-1}} + \mathbf{b_i})\\ \mathbf{o_t} &= \sigma(W_o\mathbf{x_t} + U_o\mathbf{h_{t-1}} + \mathbf{b_o})\end{aligned}$$

where $\mathbf{f_t}, \mathbf{i_t}, \mathbf{o_t}$ are referred to as the forget, input, and output gates respectively, and $W, U, \mathbf{b}$ are learned parameters. Using these gates, the hidden state $\mathbf{h_t}$ is computed as

$$\begin{aligned}\mathbf{\tilde{c}_t} &= \tanh(W_c\mathbf{x_t} + U_c\mathbf{h_{t-1}} + \mathbf{b_c})\\ \mathbf{c_t} &= \mathbf{f_t} \odot \mathbf{c_{t-1}} + \mathbf{i_t} \odot \mathbf{\tilde{c}_t}\\ \mathbf{h_t} &= \mathbf{o_t} \odot \tanh(\mathbf{c_t}),\end{aligned}$$

where $\mathbf{c_t}$ is referred to as the current cell state. A LSTM processes data sequentially in a single direction. A bi-LSTM is essentially two LSTMs combined - one processing the sequence in the forward direction and another processing it in a backward direction. The two are combined by concatenating the hidden states of the two LSTMs at each time step, so the combined network encodes information from both directions of the sequence. The hidden states of a LSTM are usually passed to further neural layers, that produce task specific output (POS tag probabilities in our case).

Conditional random fields (CRFs) (Lafferty et al. 2001) are undirected probabilistic graphical models (PGMs) for sequence labelling. The main advantage they offer over LSTMs is that they explicitly model dependencies between predicted outputs (LSTM predictions are conditionally indepdendent). Given an input sequence $\mathbf{x}$ and a label sequence $\mathbf{y}$, a CRF models the conditional distribution $p(\mathbf{y}|\mathbf{x})$. It does so by computing scores $f(\mathbf{x}, y_i, y_{i+1})$ for each position in a sequence, where f is a called the feature function. These scores are then normalised to obtain the probability $p(\mathbf{y}|\mathbf{x})$, using a dynamic programming algorithm for efficient computation. f is a often a parameterised combination of handcrafted functions that encode rules for the sequence labels (e.g. syntactic rules in the case of POS tags), but it is also possible to model f as a fully trainable function (e.g. a statistical model, or neural network). As in the case of HMMs, the most likely label sequence given an observed input sequence is computed with the Viterbi algorithm.

We experimented with two neural models for our POS tagging system - a bi-LSTM and a CRF with a bi-LSTM as feature function. Our bi-LSTM model produces concatenated hidden states for all the words in a sentence. These hidden states are then passed to a fully connected neural layer, which produces probabilities for all possible POS tags. For our CRF model, a bi-LSTM produces scores for each position in a sentence (i.e. we parameterise the feature function with a bi-LSTM). The scores are taken as input by a CRF, which computes probabilities for the tagged sentence. Both our models are trained by maximising the probabilities of tagged sentences in a training sets. For optimisation we use the Adam optimiser (Kingma & Ba 2015), a popular variant of stochastic gradient descent that adapts learning rates on a per-parameter basis.

Input features

One of our earliest findings while developing our system was that subword-based systems comfortably outperformed word-based models. This is expected, because all the task languages are agglutinative (words are formed by stringing together morphemes), so subword information is crucial. Furthermore, because the task datasets are relatively small (compared to those of high-resource languages), any held-out dataset will contain many previously unseen words. Incorporating subword features enables the system to handle new words, since it can use subword information to infer word-level properties.

We initially considered training a morphological segmenter on the morphologically analysed data, but decided against it. The task data contains canonically segmented words (words are divided into their standardised morphemes, as opposed to their surface forms) and canonical segmentation is a challenging task. Moeng et al. (2021) applied various models to the task of supervised canonical segmentation for the Nguni languages, and failed to exceed 0.75 F1 for any of the languages. Therefore we reasoned that the effort of developing a supervised canonical segmenter might not be worth the potential benefit to the model.

Instead, we incorporated subword information through two conceptually simple methods that were easy to implement and experiment with. Our first method simply segments words into their characters - the word "kuba" is represented as the character sequence "k-u-b-a". Our second method represents words as sequences of character n-grams. In our experiments we found that 2-grams worked well, and found no improvement in using higher order n-grams. Here the word "kuba" is represented as the 2-gram sequence "<k-ku-ub-ba-a>", where < and > are special symbols indicating the start and end of words.

Word composition

The final component of our system concerns how subword representations are composed to form word representations. Since POS tagging is a word-level task, we need some way to build word representations that can be processed as input by our neural networks. The method we settled on consists of simply summing the subword vector representations for a word. This was shown by Zhu et al. (2019) to be robust across different languages, compared to other composition functions. However, it discards sequential and positional information, modelling each word as a "bag-of-subwords". We also experimented with a bi-LSTM that processes a word as a sequence of subword units, and produces a vector representation for the word. Ling et al. (2015) showed that this improved performance on POS tagging, especially for morphologically rich languages. However, this significantly increased the training times of our models, and we observed no performance improvement over sum-based composition. Therefore we converged on sum-based composition early in our experiments.

5.3 Experimental setup

In addition to the components discussed above, we also employed various strategies that aid the training of deep learning models. We used a schedule for the learning rate, which determines the gradient descent step size in optimisation. We repeatedly decreased the learning rate by some factor according to a specified schedule. This ensures a high learning rate at the start of training and lower learning rates as training progresses (since smaller optimisation steps are required in the vicinity of maxima). We trained the model for a predefined number of iterations (epochs) of the training set and processed the data in batches made up of a predefined number of sentences.

Furthermore, we employed two regularisation strategies to combat overfitting - dropout and weight decay. We used dropout in our neural network layers, which randomly drops (zeroes

Table 4: The hyperparameter values we used for training our models.

Input embedding size	512
Hidden state size	512
LSTM layers	1
Initial learning rate	0.01
Adjustment schedule	every 3 epochs
Shrinkage factor	0.5
Epochs	15
Batch size	64
Dropout rate	0.2
Weight decay	1e-5
Gradient clipping	1.0

out) some proportion of units in the computed vector representations during training. Weight decay regularises the model by penalising large parameter values. We also applied gradient clipping during training, which scales gradient values if they exceed some specified threshold. This prevents excessively large gradients, which can be a problem for LSTMs.

During development, we used cross-validation to assess our systems and hyperparameters. This involved training our systems on 90% of the training set, and evaluating it on the remaining 10% (we did not have access to the test set at all during development). To ensure that our assessments were not overly dependent on a particular train/validation split, we performed multiple cross-validation experiments per system, each time assessing performance on a different 90%-10% split. To assess a system, we computed the macro-averaged F1 scores across all POS tags on the validation set (and averaged this over different validation splits). We then compared different systems according to this metric, since this is the evaluation metric used to evaluate submissions on the shared task. We repeated this tuning procedure across all the languages, but generally the same optimal hyperparameter values emerged. The hyperparameter values that we settled on through this process are listed in table 4.

6 Results

Here we present the results obtained by our systems on the shared task test datasets, and discuss our main findings. The results of the systems we experimented with are summarised in table 5, and we include a full breakdown of the performance of our final submission (crf, char + sum) in the appendix. Overall our systems performed well, consistently achieving accuracies and F1 scores above 0.85, often surpassing 0.9, and even reaching 0.95 in the case of isiXhosa.

All our systems comfortably outperform the baseline. The combination of subword features and sequential neural networks proves much more effective than the word-based HMM. Deep learning models sometimes perform poorly on small datasets, but the results confirm that the shared task datasets are large enough for neural networks to train effectively and learn generalisable rules.

Among our own systems, there are two that emerge as the best performing systems across the board. These are the CRF with 2-gram features and the bi-LSTM with character features. It is not obvious why these particular combinations of features and neural models work well, but what is clear is that the introduction of subword information proves highly effective. Performance levels vary across the languages, with all models (including the baseline) achieving their highest scores on isiXhosa and lowest scores on Siswati. This points to the existence of language-specific characteristics that may contribute to the relative difficulty of the task.

7 Conclusion

In this paper, we introduced the first shared task on Nguni Languages Part-of-Speech Tagging (NLAPOST). The dataset, which includes POS tags for four African languages (isiNdebele, isiXhosa, isiZulu and Siswati), is publicly available at `https://repo.sadilar.org/handle/20.500.12185/546`.

We also presented the results of the submitting team, which introduced an CRF classifier and a bi-LSTM system. Both systems performed well on

Table 5: The results obtained on the shared task test sets by our systems. We report the F1 scores macro-averaged over POS tags, and the test set accuracies.

		isiNdebele		isiXhosa		isiZulu		Siswati	
Model	Feature	acc	F1	acc	F1	acc	F1	acc	F1
hmm	word	0.75	0.58	0.77	0.60	0.76	0.58	0.72	0.54
crf	char + sum	0.86	0.85	0.90	0.90	0.87	0.85	0.85	0.81
	ngram + sum	0.90	**0.88**	**0.95**	**0.94**	0.91	0.86	**0.90**	**0.84**
lstm	char + sum	**0.91**	0.87	**0.95**	**0.94**	**0.92**	**0.87**	**0.90**	**0.84**
	ngram + sum	0.86	0.84	0.90	0.90	0.88	0.86	0.85	0.81

the test set, with the best results for isiXhosa (0.94 F1 score) and the lowest scores for Siswati (0.84 F1 score). These results are an encouraging contribution to natural language processing for the languages in the dataset.

Although the participation in the shared task was relatively low (despite a much higher number of registered teams), we consider the NLAPOST21 shared task a success in the sense that firstly, the winning team presented encouraging results. Secondly, the organization proved a successful collaboration between SACAIR and DHASA. Lastly, even though only one team submitted results, the dataset was introduced and made accessible to many early-carreer researchers and interested scholars, who will hopefully engage with it further.

Notes

[1] `https://dh2021.digitalhumanities.org.za/`

[2] `https://2021.sacair.org.za/`

[3] `https://repo.sadilar.org/handle/20.500.12185/546`

[4] `http://humanities.nwu.ac.za/CTexT`

[5] https://www.masakhane.io/

[6] FP, EJ and SD organized the shared task, FM delivered the system as the only participating party.

Acknowledgements

We thank CTexT, North-West University, particularly Martin Puttkammer, for supplying the dataset and documentation to us and allowing us to use it in the NLAPOST21 shared task. The work of Francois Meyer was financially supported by Hasso Plattner Institute for Digital Engineering, through the HPI Research School at UCT.

References

Baum, L. E. & Petrie, T. (1966), 'Statistical inference for probabilistic functions of finite state markov chains', *The annals of mathematical statistics* **37**(6), 1554–1563.

CTexT (2020), Annotation protocol: Part-of-speech tagging (isiZulu). Unpublished Protocol.

Dandapat, S., Sarkar, S. & Basu, A. (2007), Automatic part-of-speech tagging for bengali: An approach for morphologically rich languages in a poor resource scenario, *in* 'Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions', pp. 221–224.

Eiselen, R. & Puttkammer, M. (2014), Developing text resources for ten South African languages, *in* 'Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)', European Language Resources Association (ELRA), Reykjavik, Iceland, pp. 3698–3703.

URL: *http://www.lrec-conf.org/proceedings/lrec2014/pdf/1151ₚaper.pdf*

Halácsy, P., Kornai, A. & Oravecz, C. (2007), Hunpos: An open source trigram tagger, *in* 'Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions', ACL '07, Association for Computational Linguistics, USA, p. 209–212.

Hochreiter, S. & Schmidhuber, J. (1997), 'Long short-term memory', *Neural computation* **9**(8), 1735–1780.

Kingma, D. P. & Ba, J. (2015), Adam: A method for stochastic optimization, *in* 'International Conference on Learning Representations (ICLR)'.

Lafferty, J. D., McCallum, A. & Pereira, F. C. N. (2001), Conditional random fields: Probabilistic models for segmenting and labeling sequence data, *in* 'Proceedings of the Eighteenth International Conference on Machine Learning', ICML '01, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, p. 282–289.

Ling, W., Dyer, C., Black, A. W., Trancoso, I., Fermandez, R., Amir, S., Marujo, L. & Luís, T. (2015), Finding function in form: Compositional character models for open vocabulary word representation, *in* 'Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing', Association for Computational Linguistics, Lisbon, Portugal, pp. 1520–1530.

Moeng, T., Reay, S., Daniels, A. & Buys, J. (2021), 'Canonical and surface morphological segmentation for nguni languages'.

Onyenwe, I. E., Hepple, M., Chinedu, U. & Ezeani, I. (2019), 'Toward an effective igbo part-of-speech tagger', *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)* **18**(4), 1–26.

Patra, B. G., Debbarma, K., Das, D. & Bandyopadhyay, S. (2012), Part of speech (pos) tagger for kokborok, *in* 'Proceedings of COLING 2012: Posters', pp. 923–932.

Pienaar, W. (2021), Annotation protocol: Morphological analysis (isiZulu). Unpublished Protocol.

Saharia, N., Das, D., Sharma, U. & Kalita, J. (2009), Part of speech tagger for assamese text, *in* 'Proceedings of the ACL-IJCNLP 2009 Conference Short Papers', pp. 33–36.

Zhu, Y., Vulić, I. & Korhonen, A. (2019), A systematic study of leveraging subword information for learning word representations, *in* 'Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)', Association for Computational Linguistics, Minneapolis, Minnesota, pp. 912–932.

A Full submission results

On the next page we present a detailed breakdown of the performance of our final system on the test data. The results are broken down by part-of-speech tag, and the right-most column ("support") indicates the number of test examples per tag type. Our final system was a CRF on top of a bi-LSTM, using 2-grams as input features.

isiNdebele

	precision	recall	F1 score	support
ABBR	0.86	0.92	0.89	13
ADJ	0.98	0.88	0.93	106
ADV	0.95	0.96	0.95	693
CDEM	0.87	0.80	0.84	76
CONJ	0.93	0.95	0.94	176
COP	0.76	0.53	0.62	36
FOR	1.00	0.75	0.86	4
IDEO	1.00	0.80	0.89	10
INT	0.79	0.73	0.76	15
N	0.90	0.89	0.89	1375
NUM	0.89	1.00	0.94	8
POSS	0.91	0.92	0.91	769
PRO	0.93	0.94	0.94	71
PUNC	1.00	1.00	1.00	583
REL	0.83	0.85	0.84	444
V	0.80	0.81	0.81	647
accuracy			0.90	5026
macro avg	0.90	0.86	0.88	5026
weighted avg	0.90	0.90	0.90	5026

isiZulu

	precision	recall	F1 score	support
ABBR	1.00	0.75	0.86	8
ADJ	0.91	0.85	0.88	82
ADV	0.92	0.95	0.94	643
CDEM	0.92	0.91	0.92	107
CONJ	0.95	0.93	0.94	228
COP	0.73	0.62	0.67	65
FOR	0.67	0.67	0.67	9
IDEO	0.50	0.75	0.60	4
INT	0.82	0.75	0.78	12
N	0.89	0.91	0.90	1075
NUM	1.00	1.00	1.00	10
POSS	0.92	0.94	0.93	712
PRO	1.00	0.98	0.99	55
PUNC	1.00	1.00	1.00	590
REL	0.87	0.89	0.88	511
V	0.90	0.84	0.87	844
accuracy			0.91	4955
macro avg	0.87	0.86	0.86	4955
weighted avg	0.91	0.91	0.91	4955

isiXhosa

	precision	recall	F1 score	support
ABBR	0.94	0.94	0.94	16
ADJ	0.87	0.93	0.90	82
ADV	0.94	0.98	0.96	613
CDEM	0.98	0.96	0.97	84
CONJ	0.99	0.98	0.99	195
COP	0.86	0.74	0.80	167
FOR	0.92	0.75	0.83	16
IDEO	1.00	0.89	0.94	9
INT	0.92	1.00	0.96	12
N	0.96	0.97	0.97	1097
NUM	1.00	1.00	1.00	11
POSS	0.96	0.95	0.95	756
PRO	0.94	0.98	0.96	48
PUNC	1.00	1.00	1.00	599
REL	0.93	0.91	0.92	430
V	0.95	0.95	0.95	775
accuracy			0.95	4910
macro avg	0.95	0.93	0.94	4910
weighted avg	0.95	0.95	0.95	4910

Siswati

	precision	recall	F1 score	support
ABBR	0.88	0.64	0.74	11
ADJ	0.87	0.92	0.89	64
ADV	0.88	0.91	0.89	591
CDEM	0.82	0.68	0.74	78
CONJ	0.87	0.84	0.85	245
COP	0.72	0.57	0.64	49
FOR	1.00	0.20	0.33	10
IDEO	1.00	1.00	1.00	1
INT	0.84	0.73	0.78	22
N	0.90	0.90	0.90	1013
NUM	1.00	1.00	1.00	17
POSS	0.89	0.88	0.89	751
PRO	1.00	0.92	0.96	37
PUNC	1.00	1.00	1.00	605
REL	0.88	0.88	0.88	413
V	0.88	0.91	0.90	789
accuracy			0.90	4696
macro avg	0.90	0.81	0.84	4696
weighted avg	0.90	0.90	0.90	4696