# Reducing Annotation Dependency in Low-Resource NLP Using Bootstrap Techniques: A Shona Case Study

Tadiwa Dean Shumbanhete
*Department of Computer Science, Rhodes University*
Grahamstown, South Africa
dshumbanhete@gmail.com

*Supervised by: Dr. Z Shibeshi,*
*Co-Supervisor: Prof. N Dlodo,*

***Abstract*—This research explores methods to overcome the dependency on manual annotation for Natural Language Processing (NLP) in Machine Learning (ML). The study focusses on Shona, an agglutinative language that is under-represented in ML. Unlike high-resource languages, African languages face data scarcity and agglutinative complexity, limiting technology deployment. The study employs bootstrapping, a statistical learning method that iteratively improves NLP models using minimal labelled training data and creates ensemble datasets by sampling with replacement. Chien and Ku (2015) claim that bootstrapping techniques can lend to task agnosticism, especially when coupled with the attention of a Bayesian LSTM. Using Bayesian Variational Inference in the weights calculations allows the model to effectively model the relationships between the posterior and priors as variables/nodes in a bidirectional cyclical graph. However, limited work has been done to explore the use of such a model for low-resource NLP, and preliminary results show promise in helping reduce annotation costs while enhancing scalability.**

**Shona's agglutinative traits mirror isiXhosa and isiZulu, allowing the results of this project to advance NLP in Southern Africa. By prioritising adaptive techniques, this study aims to bridge the gaps in global language technology access.**

***Index Terms*—Artificial Intelligence, Machine Learning, Natural Language Processing, Bayesian Variational Inference, RNN, Neural Networks, LSTM, Bootstrap, low resource languages.**

## I. Introduction

Africa, the world's second-largest continent, is home to remarkable linguistic diversity, with an estimated 1,000 to 2,000 indigenous languages—representing about a third of all languages spoken globally. Of these, at least 75 African languages have over a million speakers, while others are spoken by smaller communities ranging from a few hundred to several hundred thousand individuals [1]. These languages are scarcely represented in existing Natural Language Processing (NLP) datasets, research, and tools. According to Adelani *et al.* (2021), the reasons for these disparities stem from scarcity of annotated data, limited linguistic resources, and insufficient computational research attention. According to Nekoto *et al.* (2020), low-resource languages are characterised by the absence of large-scale labelled datasets, standardised morphological and syntactic tools, and robust pre-trained models, which are critical for modern data-driven NLP approaches. The reliance on supervised learning paradigms, which require extensive annotated corpora, further exacerbates the issue, as such resources are often unavailable for languages within Kachru (1997)'s outer circle which have limited digital presence. These outer circles are communities that use high-resource languages as lingua franca but maintain use of indigenous/low-resource languages contemporaneously.

The linguistic diversity observed in many low-resource languages introduces complexities in modelling morphological and syntactic structures, particularly when these features differ significantly from high-resource languages for which NLP tools are already developed Adelani *et al.* (2021). Transfer learning and cross-lingual adaptation techniques have been proposed as potential solutions; however, their efficacy is constrained by typological dissimilarities between source and target languages. Furthermore, the lack of standardised orthographies and the prevalence of code-switching in multilingual communities introduce additional noise, complicating text normalisation and tokenization processes. Consequently, addressing these challenges necessitates collaborative efforts involving linguists, native speakers, and computational researchers to develop sustainable data collection methodologies, unsupervised or semi-supervised learning frameworks, and linguistically informed architectures tailored to the unique characteristics of low-resource languages. However, this is not an easy task.

In order to overcome these hurdles in the context of Shona NLP, there is a need to employ techniques that optimise performance while maintaining accuracy. Yao *et al.* (2023) propose using bootstrap techniques to reduce the demands on large training resources. Chien and Ku (2015) show that the use of Bayesian Variational inference can increase the ability of a neural network to work in noisy conditions. According to Xue *et al.* (2021), Zaman *et al.* (2022) sophisticated uncertainty estimation of weight distributions, lend towards

[1]The African Language programme at Harvard: https://alp.fas.harvard.edu/introduction-african-languages.

a RNN model that is task agnostic. In order to properly assess this proposition, there must first be a firm understanding of the problem domain, and related concepts such as NLP and neural networks, as well as the Bayesian bootstrap.

### A. Research question

Can bootstrap methods reduce reliance on annotated resources for NLP tasks, such as Named Entity Recognition? If so, how would this system perform under low resource constraints (eg, -10% training data)?

*Objectives*

**Primary objectives:**

• To evaluate the impact of bootstrap techniques on reducing annotation dependency while maintaining or improving accuracy on downstream tasks (e.g. NER).

• To investigate the challenges of NLP in low-resource African languages by reviewing existing datasets and models.

• To assess the effectiveness of bootstrap techniques for self-supervised learning.

• To propose a means by which to increase the diversity of the dataset in self-supervised learning.

**Secondary objectives:**

• Explore whether bootstrapping can effectively aid the development of natural language processing in low-resource African languages.

• Evaluate the performance of bootstrapping techniques (e.g., self-training, paraphrasing, synthetic data generation, co-training, distant supervision) for NER.

• To propose a framework for scalable, low-resource NLP systems using minimal human annotation.

## II. LOW-RESOURCE NLP

Natural Language Processing (NLP) is a subfield within the field of Machine Learning (ML) and deals with the pre-processing and processing of plain text, the output of which can be useful to downstream applications and functions Bhagavatula *et al.* (2012). Some of the tasks in NLP include Named Entity Recognition, Part of Speech tagging, lemmatisation, text summarisation, and semantic analysis. These tasks aid complex NLP tasks, such as Machine Translation (MT) and Language Modelling (LM), which are vital steps in interfacing with users and automating workflows Jawad and Balázs (2024). Adelani *et al.* (2021) make a significant stride towards addressing under-representation of African languages NLP research. NLP research disproportionately prioritises high-resource languages like English, Mandarin, Spanish, and French. These languages have large, annotated datasets (corpora), well-developed tools, and significant research communities. Many languages are therefore termed low-resource and African languages like Shona are further constrained by complex morphological structures, such as agglutinative or polysynthetic morphologies.

The Adelani *et al.* (2021) research team coordinated an effort between numerous NLP stakeholders to establish the first open-source dataset, named The MasakhaNER 2.0 for Named Entity Recognition (NER), spanning 20 African languages. Through their first (MasakhaNER 1.0 with ten languages) and second implementation (in MasakhaNER 2.0 with another ten) Adelani *et al.* (2022) help researchers and practitioners better understand the challenges of developing NLP models for low-resource languages. Their findings show that transformer pre-trained Large Language Models (tPLMs) work the best for Shona. However, performance is still poor with mBERT's F1 score dropping by 25% on Shona NER tasks without fine-tuning. For Shona, the tPLMs perform 6 and 12 F1 worse on mBERT and AfroBERTa. Additionally, there are similar observations when these models are applied to a different domain in the same target language. Adelani *et al.* (2021) reveal that dominant NLP methods continue to popularise the use of tPLMs, however these methods introduce a high dependence on human annotation as well as scalability issues. These models also run into problems of overfitting to the training data when a small volume has been provided for pre-training. The MasakhaNER 2.0 dataset with around 60000 entities is considered a small-medium sized dataset. Yao *et al.* (2023) propose means by which to overcome this dependency on training resources by using statistical techniques that can either help us to reconstruct the distribution from a given sample, create synthetic data, or increase the ability to estimate in uncertain or noisy conditions typical of low-resource NLP.

NLP tasks often involve sequential data, where the meaning of a word depends on its context within a sentence. Typically neural networks dominate NLP research due to their ability to model sequential dependencies by maintaining a hidden state that captures temporal information. The probabilistic problem domain in NLP typically involves estimating the conditional probability distribution of the next sequence (word or character) given previous sequences of tokens $x = (x_1, x_2, \ldots, x_{\mathrm{T}})$ the objective is to model the joint probability distribution:

$$P(\mathbf{x}) = \prod_{t=1}^{T} P(x_t | x_{<t})$$

where $x_{<t} = (x_1, \ldots, x_{t-1})$ represents the history up to time $t$.

*Neural Network for NLP*

Neural networks (NN) have a feed-forward mechanism which processes fixed-length inputs, sequentially as independent. The simplest kind of feedforward neural network is a linear network, which consists of a single layer of output nodes with linear activation functions (eg. softmax, ReLU or tanh). The output of one time step is the input to the next time step via a series of weights. The sum of the products of the weights and the inputs is calculated at each node. The mean squared errors between these calculated outputs and the given target values are minimized by creating an adjustment to the weights. However, the longer the range of the dependencies, the gradients between the weights becomes too small. A standard NN for NLP assumes:

$$P(x_t|x_{<t}) \approx P(x_t) = \text{[Activation function]}(Wx_t + b)$$

This approach fails to capture long-range dependencies in NLP because of its fixed input dimensionality. Various solutions have been developed to solve this problem including Convolutional Neural Networks, Recurring Neural Networks (RNNs), and Long-Short-Term-Memory RNNs.

*Convolutional Neural Network (CNN) for NLP*

CNNs apply filters over local windows of text, capturing n-gram patterns but lacking explicit memory of long-range dependencies. For a sequence $\mathbf{x}$, a 1D convolution computes:

$$h_i = \sigma(\mathbf{W} \cdot \mathbf{x}_{i:i+k-1} + b)$$

where $k$ is the kernel size. As CNNs convolutional layer slides a given filter across the entire input the model is faster at training than a typical neural network. This is attributable to parameter sharing between each convolutional operation. Parameter sharing of the convolutional operation does however increase linearly and therefore increases computational requirements quadratically. CNNs also excel at modelling spatial dependencies due to locality of the feature extraction as the output of the convolutional layer. CNNs however lack understanding of tasks that have temporal dependencies such as speech recognition, time-series forecasting, and language modelling.

*Recurrent Neural Network (RNN) for NLP*

An RNN processes sequential data through use of feedack loops that allow the model to retain information from previous inputs. At each time step, a hidden state is updated as a function of the current input and the previous hidden state $t$:

$$h_t = \sigma(\mathbf{W}_h h_{t-1} + \mathbf{W}_x x_t + b)$$

This formulation allows the network to condition predictions on prior context, making RNNs suitable for tasks like language modelling. However, standard RNNs suffer from vanishing/exploding gradients, limiting their ability to learn long-term dependencies. These challenges can be mitigated by use o advanced RNNs such as Gated Recurrent Units or LSTMs.

*Long Short-Term Memory (LSTM)*

LSTMs mitigate the vanishing gradient problem evident in RNNs using gating mechanisms:

$$\begin{aligned}
f_t &= \sigma(\mathbf{W}_f[h_{t-1}, x_t] + b_f) \quad \text{(Forget Gate)} \\
i_t &= \sigma(\mathbf{W}_i[h_{t-1}, x_t] + b_i) \quad \text{(Input Gate)} \\
\tilde{C}_t &= \text{[(f)]}(\mathbf{W}_C[h_{t-1}, x_t] + b_C) \quad \text{(Candidate Memory)} \\
C_t &= f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \quad \text{(Memory Update)} \\
o_t &= \sigma(\mathbf{W}_o[h_{t-1}, x_t] + b_o) \quad \text{(Output Gate)} \\
h_t &= o_t \odot \text{[Activation Function (f)]}(C_t) \quad \text{(Hidden State)}
\end{aligned}$$

LSTMs improve long-term dependency modelling but remain deterministic, lacking uncertainty quantification.

*Bayesian LSTM for NLP*

A Bayesian LSTM introduces probabilistic weight distributions:

$$\mathbf{W} \sim \mathcal{N}(\mu, \Sigma)$$

Let $\theta = \{\mathbf{W}, \mathbf{b}\}$ denote network parameters, which are treated as random variables with a prior $P(\theta)$. The posterior is approximated via variational inference:

$$P(\theta|\mathcal{D}) \approx q_\phi(\theta)$$

where $\phi$ are variational parameters. The hidden state update becomes stochastic:

$$h_t = \sigma(\mathbf{W}_h h_{t-1} + \mathbf{W}_x x_t + b), \quad \mathbf{W}_h, \mathbf{W}_x \sim q_\phi(\theta)$$

where $q_\phi(\theta)$ is a variational posterior. The predictive distribution becomes:

$$P(x_t|x_{<t}) = \mathbb{E}_{q_\phi(\theta)}\left[P(x_t|x_{<t}, \theta)\right]$$

Bayesian LSTMs therefore allow for more sophisticated weight calculations by turning a point estimate into a distribution. By using appropriate activation functions, such as the soft-max activation function, the model could effectively introduce nonlinearity while maintaining the ability to estimate uncertainty.

*Transformers*

Transformers are a from of neural network based architecture with an attention mechanism. These multi-head self-attention mechanisms allow transformers to learn long-range contexts. According to Vaswani *et al.* (2017) Transformers are superior in NLP due to their ability to have sliding context windows. These windows allow the model to work on inputs of various lengths but are limited in that their computational requirements increase quadratically with the size of the context window. This is partly due to transformers ability to parallelise operations on a sequence of data. In Adelani *et al.* (2021) their choice to use a tPLM could be attributable to the superior ability of transformer architectures to work much faster. However, without the access to the NVIDIA GeForce RTX 2080 Ti or similar cloud computing capabilities the model is unable to be run. Transformers were introduced as a response to the problems encountered with neural networks such as the vanishing or exploding gradient.

## III. Methodology

Problem Statement

African languages lack a robust corpus for training a machine learning algorithm based on supervised, and semi-supervised methods. Therefore, by creating an model predicated on unsupervised learning, annotation dependency can become less of a barrier of African NLP development.

### A. Assessment: candidate low-resource architecture

In LSTM RNNs long-distance history words are continuously aggregated through feed forward mechanisms and back-propagation. Back-propagation in a Bayesian Long Short-Term Memory (LSTM) network involves optimisation under a probabilistic framework, where weights are treated as random variables rather than deterministic values. Unlike traditional back-propagation, which computes point estimates of gradients, Bayesian back-propagation incorporates uncertainty in parameters through variational inference or sampling-based methods. Instead of estimating multinomials in an n-gram, the LSTM estimates the stochastic (probabilistic) weights via a reparametrisation trick. At each time step, the weights are updated through linear transformations. These linear transformations are applied through all the layers, including input-hidden, recurrent-hidden, and hidden-output layers Chien and Ku (2015).

LSTMs are an ideal candidate architecture because they deal with the data sparseness problem through the continuous-space LM and simultaneously accommodate long-distance information through dynamic feedback. The network complexity is controlled by the size of dictionary and the number of hidden units in an LSTM.

We proceed by exploring the pre-training tasks for an LSTM-LM.

## IV. Pre-training: Masked Language modelling and bootstrap approaches to data augmentation

The study is primarily focused on a machine learning problem; that of annotation dependency and generalisation. The problem domain is a stochastic one, where weights between variables in the graph are represented as distributions. The challenge will involve creating a robust framework to work in noisy conditions, work with limited data for risky decision making, while preventing the effects of overfitting from increasing.

### A. Assessment: candidate pre-training task

Given a neural network $f_t W\odot$ where $W$ represents possibly fitted parameters from pre-training, as well as an input sample $x$, our goal is to evaluate the uncertainty of the prediction of the model, $y = fW(x)$. Specifically, we would like to quantify the prediction standard error, $\eta$, so that an approximate $\alpha$-level prediction interval can be constructed by $[yz\alpha/2\eta, y+z\alpha/2\eta]$ where $z\alpha/2$ is the upper $\alpha/2$ quantile of a standard Normal. With many options for pre-training, Adelani *et al.* (2022) identify back translation as an appropriate candidate to better model generic patterns in language families. Back translation would also be an ideal pre-training task for transfer learning to new tasks, and new domains. Such an approach would, however, necessitate creation of corresponding training and testing datasets. This does not address our primary objective of reducing annotation dependency. As such we reject this task in favour of the one used in the original study. In MasakhaNER 2.0 the authors implement a transformer fine-tuning architecture using a masked language model task. Feng *et al.* (2021) survey of data augmentation methods shows that this pre-training task is also appropriate when a model has not been pre-trained, and is task-agnostic. It is also compatible with the bootstrap principles outlined by Efron (1992) in his seminal paper where bootstrapping is first introduced.

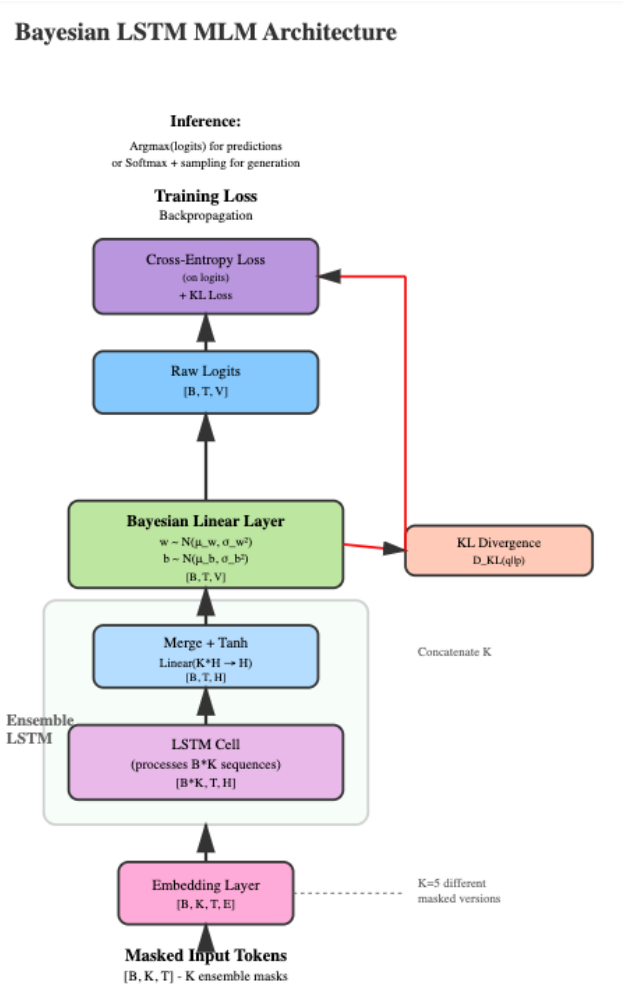


Fig. 1. Overview of the Bayesian NNLSTM architecture.

## V. Fine-tuning

The fine-tuning phase was designed to balance model stability, uncertainty calibration, and generalisation across low-resource contexts. A Bayesian LSTM architecture was employed, combining Cross-Entropy (CE) and Kullback–Leibler (KL) losses to ensure both predictive accuracy and uncertainty-aware regularisation. The model was fine-tuned for 100 epochs using a modest learning rate to prevent gradient instability, with early stopping monitored through validation CE loss trends. Batch size and sequence length were constrained to preserve coherence in word–cluster associations, while dropout and KL annealing were applied to mitigate over-confidence during convergence. Fine-tuning was performed

on the unbootstrapped corpus first to establish a baseline distribution of entity recognition, allowing subsequent comparison with bootstrapped and augmented variants. Overall, these design choices ensured stable convergence (KL ↓ 4100→134) and effective posterior regularisation, even though performance on novel data later revealed sensitivity to distributional drift—underscoring the importance of semi-supervised adaptation in low-resource named entity recognition. To keep with comparability, we implement the fine-tuning and pre-training split of training, development, and test consisting of 70%, 10%, and 20% of the data respectively. A 10% reduction was implemented in order to simulate low-resource constraints.

## VI. Data augmentation

As we have identified an architecture that implements (Bayesian) bootstrapping, we also attempt to incorporate bootstrapping techniques in reducing annotation dependencies for pre-training. We therefore apply data augmentation techniques to the masked language modelling pre-training task. Feng *et al.* (2021) posit that data augmentation is a means by which a model uses available data to create synthetic data on which to train a ML model. Data augmentation has been extensively used with RNNs and LSTMs in speech recognition and image processing tasks as it introduces diversity into the data set. Data augmentation techniques also maximise the ability to extract temporal dependencies from training data. In image processing, the use of methods such as cropping and rotating has been seen to improve training efficacy. In order to explore ways in which data augmentation techniques can be applied to NLP tasks we turn to Fabbri *et al.* (2020) survey of 30 data augmentation techniques. Fabbri *et al.* (2020) identify dependency tree morphing as a viable candidate for our proposed architecture. Dependency tree morphing does not require external knowledge (e.g. WordNet) or a pre-trained model Şahin and Steedman (2019). It is also task agnostic and works well at parsing dependencies. Data Augmentation methods can ease few-shot learning by adding more examples for novel classes introduced in the few-shot phase. Implementation of dependency tree morphing involves use of a clustering algorithm, where at each ensemble creation, variables with high prediction weights are swapped with other words from the cluster.

## VII. Assessment and refinement

The pre-training performance is assessed using cross-entropy loss. The Bayesian LSTM is compared to the XLM-RoBERTa baseline. The TPLM uss masked language model pre-training which is used to augment annotated data using dependency tree morphing. We choose to implement this task as it is coherent with the scope of this study, and adequately addresses annotation dependency. The efficacy of the model will be assessed with cross-entropy loss, and Kullback-Lieblier divergence for regularisation. Cross-entropy loss is the most appropriate loss function because it was used in the foundational MasakhaNER study, and it is also appropriate for assessing the models ability to reduce the distance between predicted and true labels for next word prediction with masking Fabbri *et al.* (2020). The fine-tuning stage will be assessed with mean F1, accuracy and recall scores. Recommendations will centre around the findings derived from these metrics.

## VIII. Results, Analysis and Dicussion

## IX. Bayesian LSTM pre-training results

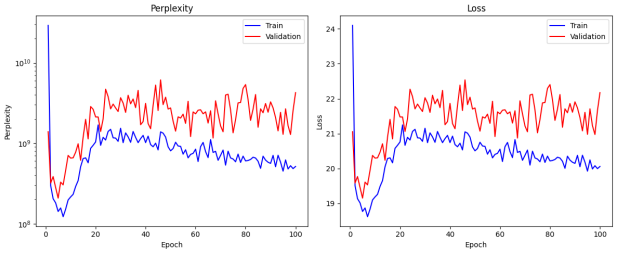


Fig. 2. Bayesian LSTM MLM.

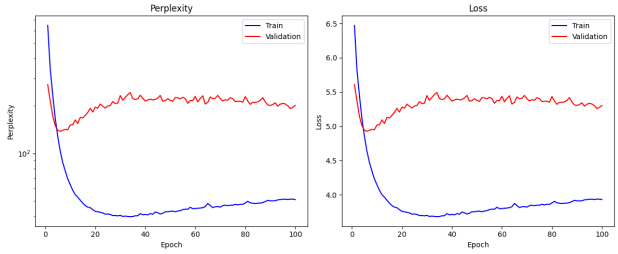


Fig. 3. HuggingFace Transformer MLM

TABLE I
Summary of Experimental Results for the Shona NER Model

| Experiment | Dataset Type | Accuracy | Macro Avg (F1) | Weighted Avg (F1) | Notes / Observations |
|---|---|---|---|---|---|
| **Exp. 1: Pre-training (Bayesian LSTM)** | Validation | 71.7% | – | – | Training accuracy improved from 66.2% to 71.5%; unstable perplexity (1.77M best). |
| **Exp. 1: Transformer Baseline** | Validation | 94.2% | – | – | Significantly lower validation perplexity (200.95); smoother convergence. |
| **Exp. 2a: Unbootstrapped NER** | Validation | 93% | 0.76 | 0.93 | Excellent on I-PER (F1=0.97); weak on I-LOC (F1=0.64), showing poor generalisation to less supported classes. |
| **Exp. 2a: Unbootstrapped NER** | Test | 76% | 0.33 | 0.76 | Sharp performance drop on novel data; I-PER F1 down from 0.97→0.29. |
| **Exp. 2b: Bootstrapped NER** | Validation | 93% | 0.77 | 0.94 | Balanced improvement across minority classes; high precision/recall on I-PER (F1=0.97). |
| **Exp. 2b: Bootstrapped NER** | Test | 70% | 0.30 | 0.73 | F1 drop (−23%); shows correlated accuracy shifts across classes. |
| **Exp. 3a: Data Augmentation** | Validation | 74% | 0.11 | 0.71 | Severe degradation across entity classes; I-LOC remained strongest (F1=0.86). |
| **Exp. 3a: Data Augmentation** | Test (Generated Data) | 74% | 0.11 | 0.72 | Noise from paraphrasing reduced meaningful associations. |
| **Exp. 3b: Data Augmentation** | Validation | 74% | 0.11 | 0.71 | Severe degradation across entity classes; I-LOC remained strongest (F1=0.86). |
| **Exp. 3b: Data Augmentation** | Test (Paraphrased Data) | 74% | 0.11 | 0.72 | Noise from paraphrasing reduced meaningful associations. |

### A. Experiment 1: Pre-training (Bayesian LSTM

An LSTM model, when run for 100 epochs, produces the best valid perplexity, 1772845.86, which was observed at epoch 7. The final train/valid accuracy gap is 0.20. There is no sign that training or validation loss has plateaued, and there is still +153.14% perplexity drift between earlier and later epochs.

### B. Experiment 1: Transformer Pre-training (Baseline)

For the Transformer pre-training over 100 epochs, the final training perplexity was approximately 50.94, and the final training loss was 3.9307. On the validation set, the final perplexity was 200.95, and the final loss was 5.3031.

The plots for the Transformer pre-training show that the training perplexity decreases steadily, while the training loss increases, indicating ineffective learning on the training data. The validation perplexity also decreases initially but then stabilises, and the validation loss shows a similar trend, suggesting that the model is learning the associations.

*Discussion*

Comparing the Transformer pre-training results to the Bayesian MLM pre-training results (where the final training perplexity was approximately 514.33, and validation perplexity was around 4259857092.49), we observe notable differences.

While both models exhibit a gap between training and validation performance, the Transformer achieved a significantly lower final validation perplexity (200.95) compared to the Bayesian MLM (4259857092.49). This suggests that the standard Transformer was less uncertain in its predictions on the validation set after pre-training.

The convergence behaviour also differed. The transformer's validation metrics, while still fluctuating, appeared somewhat more stable than the significant volatility observed in the Bayesian MLM's validation perplexity. The transformer architecture exhibits a smoother distribution in its loss and perplexity metrics, which can be attributed to the optimiser chosen (torch.optim.Adam). The gap between the training and validation loss stays relatively constant after epoch 18, which is a sign of convergence as well as good generalisation. This is the typical performance of a transformer architecture, primarily attributable to the multi-head self-attention. Training time: 2-3 days.

*Summary*

The implications of these comparative pre-training results are significant. The standard Transformer, with its lower validation perplexity, appears to have learned a more effective language model representation for the validation data compared to the Bayesian MLM. This improved pre-training performance on unseen data could potentially lead to faster convergence and higher performance during the subsequent task-adaptive pre-training step for the NER task. While the Bayesian MLM offers the advantage of uncertainty estimation, its pre-training performance, at least based on these metrics, appears to lag behind that of the standard Transformer on this specific dataset and task. The fine-tuning results will be crucial in determining whether the advantages of the Bayesian approach, in terms of uncertainty, outweigh the seemingly better representational learning of the standard Transformer during pre-training.

### C. Unbootstrapped NER

The XLM-RoBERTa model baseline sits at 94.2% validation accuracy. The unbootstrapped pre-training achieved stable convergence and excellent validation accuracy (92%). The Bayesian regularisation term (KL divergence) reduced dramatically, indicating efficient uncertainty calibration. However, class imbalance and unseen lexical patterns caused steep performance decay on the novel dataset — motivating the bootstrapped semi-supervised extension that followed. Results in Table I show the validation results of the NER model when it has not been bootstrapped. The model shows excellent overall performance. Although 93% ¡ 94.2%, which is the transformer baseline, we can confidently reject the null hypothesis that Bayesian methods cannot approximate the performance of the transformer. However, a class imbalance is present, with the B-PER label having the highest overall F1, precision, and recall scores. The model performs very poorly in the B-LOC category, and the macro-average indicates that the good performance on B-PER is inflating the predictive ability of this model. In general, the model appears to perform better on beginning tags than on inner tags.

### D. Bootstrapped NER

Validation with -10% dev data provides a detailed evaluation of the model's performance under low-resource constraints.

- **Precision:** The model demonstrates high precision for the 'O' label (0.96), indicating that when it predicts a token is not a named entity, it is very likely correct. Precision for entity labels varies, with 'I-PER' showing the highest precision (0.88) among entity tags, and 'B-ORG' having the lowest (0.63).
- **Recall:** Recall is also highest for the 'O' label (0.97), meaning the model correctly identifies most non-entity tokens. Among entity tags, 'B-DATE' has the highest recall (0.83). In contrast, 'I-DATE' and 'I-LOC' have the lowest recall (0.62), suggesting the model struggles to identify all instances of inside date and location entities.
- **F1-score:** The F1-score, a balance of precision and recall, provides a comprehensive measure of performance per label. The 'O' label achieves an excellent F1-score of 0.97. For named entities, F1-scores range from 0.66 ('B-ORG') to 0.82 ('B-DATE'). The relatively lower F1-scores for some tags, particularly 'B-ORG', 'B-PER', 'B-LOC', 'I-DATE', and 'I-LOC', highlight the challenges in accurately identifying the boundaries and types of all named entities, especially for less frequent or more ambiguous entities.
- **Support:** The support column shows the distribution of labels in the validation set. The 'O label has significantly higher support (31513) compared to the entity labels, which are minority classes.

Accuracy (0.93): The high overall accuracy is primarily influenced by the model's strong performance on the majority of the O class. Macro Avg F1-score (0.76): The macro average F1-score, which treats all classes equally, provides a more representative measure of the model's performance across all entity types, suggesting room for improvement in handling the minority entity classes. Weighted Avg F1-score (0.93): The weighted average F1-score, weighted by class support, aligns closely with the overall accuracy and reflects the model's overall effectiveness, considering the class distribution.

As seen in Table II, model accuracy drops by 23% F1 score. The weighted average remains fairly high, but when looking at each of the classes, even classes with high support drop in their F1 scores. However, there is evidence of accuracy moving in the same direction for all of the classes. We hypothesise that by performing the masked language modelling task, the model is better able to map the relationships between variables in the network, which increases their covariance. When the

| Label | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| I-PER | 0.31 | 0.31 | 0.31 | 1803 |
| B-DATE | 0.17 | 0.24 | 0.20 | 1290 |
| B-LOC | 0.17 | 0.20 | 0.19 | 449 |
| O | 0.14 | 0.29 | 0.19 | 1739 |
| I-DATE | 0.31 | 0.31 | 0.31 | 708 |
| I-ORG | 0.38 | 0.26 | 0.31 | 371 |
| I-LOC | 0.88 | 0.79 | 0.83 | 47709 |
| B-ORG | 0.13 | 0.25 | 0.17 | 1226 |
| B-PER | 0.16 | 0.28 | 0.20 | 2105 |
| accuracy | | | 0.70 | 57400 |
| macro avg | 0.30 | 0.32 | 0.30 | 57400 |
| weighted avg | 0.76 | 0.70 | 0.73 | 57400 |

TABLE II
CLASSIFICATION REPORT ON THE TEST DATASET.

estimations are good, the entire system benefits, and vice-versa.

### *E. Data Augmentation: Generations*

Compared to the non-paraphrase augmented model, results show lower performance on all class metrics. There is a drop in the weighted average and macro-average, contradicting some of the conclusions we made in the previous stage. We hypothesise that introducing diversity into an unstable model only yields more instability. The I-LOC category has deteriorated the least, maintaining the models' ability to identify this class precisely. We hypothesise that by adding the prefixes in our paraphraser, we have increased the ability of the model to distinguish between classes based on the beginning of the word. However, this is only visible in that single category. The validation and test metrics follow a near identical pattern.

### *F. Data Augmentation: Paraphrases*

| Label | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| I-LOC | 0.83 | 0.88 | 0.86 | 47709 |
| O | 0.04 | 0.04 | 0.04 | 2105 |
| B-DATE | 0.03 | 0.01 | 0.02 | 1290 |
| I-PER | 0.02 | 0.02 | 0.02 | 1226 |
| B-ORG | 0.02 | 0.01 | 0.01 | 449 |
| I-DATE | 0.01 | 0.01 | 0.01 | 708 |
| B-PER | 0.01 | 0.00 | 0.00 | 371 |
| I-PER | 0.03 | 0.01 | 0.01 | 1803 |
| O | 0.03 | 0.02 | 0.02 | 1739 |
| accuracy | | | 0.74 | 57400 |
| macro avg | 0.11 | 0.11 | 0.11 | 57400 |
| weighted avg | 0.69 | 0.74 | 0.72 | 57400 |

TABLE III
CLASSIFICATION REPORT ON THE TEST DATASET.

Compared to the non-paraphrase augmented model, the paraphrase-augmented model performs lower on all class metrics. The drop in the weighted average and macro-average contradict some of the conclusions we made in the previous stage. We hypothesise that introducing diversity into an unstable model only yields more instability. The I-LOC category has deteriorated the least, maintaining the models' ability to identify this class precisely. We hypothesise that by adding the prefixes in our paraphraser, we have increased the ability of the model to distinguish between classes based on the beginning of the word. However, this is only visible in that single category. The validation and test (in Table III follow a near identical pattern.

| Label | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| B-PER | 0.04 | 0.04 | 0.04 | 2105 |
| B-DATE | 0.03 | 0.01 | 0.02 | 1290 |
| B-ORG | 0.02 | 0.02 | 0.02 | 1226 |
| B-LOC | 0.02 | 0.01 | 0.01 | 449 |
| I-DATE | 0.01 | 0.01 | 0.01 | 708 |
| I-LOC | 0.83 | 0.88 | 0.86 | 47709 |
| I-ORG | 0.01 | 0.00 | 0.00 | 371 |
| I-PER | 0.03 | 0.01 | 0.01 | 1803 |
| O | 0.03 | 0.02 | 0.02 | 1739 |
| accuracy | | | 0.74 | 57400 |
| macro avg | 0.11 | 0.11 | 0.11 | 57400 |
| weighted avg | 0.69 | 0.74 | 0.72 | 57400 |

TABLE IV
CLASSIFICATION REPORT ON TEST WITH GENERATED DATASET.

Table IV shows the test metrics on NER for the low-resource bootstrap self-supervised learning loop. Although the overall accuracy is still fairly high, the ability of the model to make any meaningful associations is being deteriorated by the noise being introduced. Engaging or disabling MC dropout in the paraphrases does not help in standardising the non-standard way that grammar is being generated by the paraphraser. The model is struggling to make distinctions between variables in the network.

## X. DISCUSSION

The experimental results demonstrate that the proposed hybrid NER model is capable of performing named entity recognition on the Shona dataset. The integration of k-means clustering features with word embeddings provides additional information to the LSTM layer, potentially aiding in the identification of entities. The Bayesian Linear layer introduces uncertainty estimation, which could be valuable for downstream tasks or for identifying low-confidence predictions.

While the model achieves high overall accuracy and performs very well on non-entity tokens, there are clear opportunities for improving its performance on specific entity types, particularly those with lower recall or F1-scores. When applying the Bayesian LSTM model without pre-training we reaped short training times and decent performance, however this performance drops to 76% (-17% F1 score) when evaluated against novel data. Looking at the macro-average also shows that each class has significantly deteriorated in the face of the novel data. Applying bootstrap techniques in the pre-trained weights showed signs of improving the class imbalances while still keeping the overall accuracy high in all categories, even improving those in under-supported classes. When evaluated against novel data, the model F1 score drops by 20% but maintains good class representation for each of the categories. Data augmentation strategies do not erode overall accuracy but increase class imbalances, showing over-fitting towards classes that occur most frequently. In, general O tags were difficult for the model to identify, as well as the inner tags. For paraphrasing and future work the data supports leaning against

the tendency to identify classes starting with capitalisation as key feature.

## XI. Conclusion and Future work

The study attempts to address agglutination in a low-resource African language, revealing that dominant models do not work well for this domain. Through review of the literature and related studies, it was discovered that little to no standardised methods for computational modelling exist. Subsequently, an exploration for candidate architectures led to experimentation with bootstrap methods and their associated Bayesian architectures. Through the collection of cross-entropy loss metrics, as well as precision, accuracy, and recall on NER, results revealed insights into the behaviour of Bayesian architectures for language modelling. Our findings revealed that there is instability in the model, which can be somewhat overcome with regularisation; however, this does not prevent it from degrading by 13-20% F1 scores when tested on novel data.

Future work should explore techniques to enforce agglutinative behaviour. Without engagement with stakeholders in linguistics, it appears to be an insurmountable barrier to overcome. Classification tasks centre around sophisticated predictions that address class imbalance, and perfecting the scalable model that this study proposes may lend towards achieving that objective. Investigation of bootstrap techniques does show promise for use in low-resource settings; however, insufficient research exists that explores the use of statistical methods for Language modelling. This study comprises the first available open-source Bayesian LSTM for NLP, to our knowledge. As such, considering that we have reduced training times without significantly deteriorating performance, this study aims to create a foundation for future development in this exciting area.

## XII. Acknowledgements

This work was undertaken in the Distributed Multimedia CoE at Rhodes University, with financial support from Telkom SA. The authors acknowledge that opinions, findings and conclusions or recommendations expressed here are those of the author(s) and that the above-mentioned sponsor accepts no liability whatsoever in this regard.

## XIII. Appendix

Github repository. [2].

[2]Bootstrap Shona LM: https://github.com/dshumbanhete/Bootstrap_Shona_LM

## References


**Adelani, D. I., Abbott, J., Neubig, G., D'souza, D., Kreutzer, J., Lignos, C., Palen-Michel, C., Buzaaba, H., Rijhwani, S., Ruder, S.** *et al.* Masakhaner: Named entity recognition for african languages. *Transactions of the Association for Computational Linguistics*, 9:1116–1131, 2021.

**Adelani, D. I., Neubig, G., Ruder, S., Rijhwani, S., Beukman, M., Palen-Michel, C., Lignos, C., Alabi, J. O., Muhammad, S. H., Nabende, P.** *et al.* Masakhaner 2.0: Africa-centric transfer learning for named entity recognition. *arXiv preprint arXiv:2210.12391*, 2022.

**Bhagavatula, M., GSK, S., and Varma, V.** Named entity recognition an aid to improve multilingual entity filling in language-independent approach. In *Proceedings of the first workshop on Information and knowledge management for developing region*, pages 3–10. 2012.

**Chien, J.-T. and Ku, Y.-C.** Bayesian recurrent neural network for language modeling. *IEEE transactions on neural networks and learning systems*, 27(2):361–374, 2015.

**Efron, B.** Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics: Methodology and distribution*, pages 569–593. Springer, 1992.

**Fabbri, A. R., Han, S., Li, H., Li, H., Ghazvininejad, M., Joty, S., Radev, D., and Mehdad, Y.** Improving zero and few-shot abstractive summarization with intermediate fine-tuning and data augmentation. *arXiv preprint arXiv:2010.12836*, 2020.

**Feng, S. Y., Gangal, V., Wei, J., Chandar, S., Vosoughi, S., Mitamura, T., and Hovy, E.** A survey of data augmentation approaches for nlp. *arXiv preprint arXiv:2105.03075*, 2021.

**Jawad, Z. N. and Balázs, V.** Machine learning-driven optimization of enterprise resource planning (erp) systems: a comprehensive review. *Beni-Suef University Journal of Basic and Applied Sciences*, 13(1):4, 2024.

**Kachru, B. B.** World englishes and english-using communities. *Annual review of applied linguistics*, 17:66–87, 1997.

**Nekoto, W., Marivate, V., Matsila, T., Fasubaa, T., Kolawole, T., Fagbohungbe, T., Akinola, S. O., Muhammad, S. H., Kabongo, S., Osei, S.** *et al.* Participatory research for low-resourced machine translation: A case study in african languages. *arXiv preprint arXiv:2010.02353*, 2020.

**Şahin, G. G. and Steedman, M.** Data augmentation via dependency tree morphing for low-resource languages. *arXiv preprint arXiv:1903.09460*, 2019.

**Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I.** Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

**Xue, B., Yu, J., Xu, J., Liu, S., Hu, S., Ye, Z., Geng, M., Liu, X., and Meng, H.** Bayesian transformer language models for speech recognition. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7378–7382. IEEE, 2021.

**Yao, Y., Jin, W., and Ravi, S.** Labeling without seeing? blind annotation for privacy-preserving entity resolution. *arXiv preprint arXiv:2308.03734*, 2023.

**Zaman, M., Saha, S., Zohrabi, N., and Abdelwahed, S.** Uncertainty estimation in power consumption of a smart home using bayesian lstm networks. In *2022 IEEE International Symposium on Advanced Control of Industrial Processes (AdCONIP)*, pages 120–125. IEEE, 2022.