Evaluating Ensemble-Based Classifiers for Khasi Named Entity Recognition with Data Imbalance Constraints

Ransly Hoojon[1], Amitabha Nath[2*] and Saralin Lyngdoh[3]

[1,2] Information Technology Department, North Eastern Hill University, Umshing Mawkynroh, Shillong, 793022, Meghalaya, India

[3] Linguistic Department, North Eastern Hill University, Umshing Mawkynroh, Shillong, 793022, Meghalaya, India

Emails: ranslyh@gmail.com[1] , amitabha.me@gmail.com[2] , saralyngdoh@gmail.com[3]
* Corresponding Author

Abstract

Named Entity Recognition (NER), a core task of NLP, has been the subject of much research in resource-rich languages. However, little has been attempted with respect to low-resource languages such as Khasi. Based on our previous efforts, the existing works on Khasi NER applied transformer models such as RoBERTa, XLM-RoBERTa, and BERT-base-multilingual-cased using transfer learning (TL) and achieved an F1 score of 91.25% with RoBERTa. In this paper, ensemble methods are investigated to improve performance even further. We used a novel Khasi named entity recognition dataset annotated in span-based and converted to IOB2 formats for token-level predictions for both TL and ensemble methods. Ensemble methods, including majority voting, weighted voting, and stacking with logistic regression and XGBoost, were employed. Two settings were tested, one where two models are implemented (RoBERTa and XLM-RoBERTa) and the other with three models, with the addition of BERT-base-multilingual-cased, where these models act as learners. The ensemble of three models with XGBoost scored the highest F1 score of 94.06%, which is an improvement of +2.81% over the previous best. It also achieved significant improvements on rare entities, such as WORK_OF_ ART, LANGUAGE, ORDINAL, and EVENT, although the dataset was skewed toward the PERSON class. The results show how ensemble methods

can improve NER performance on imbalanced datasets and low-resource languages such as Khasi**.**

Keywords: Named Entity Recognition, Low-Resource Language, Khasi Language, Transformer-based Models, Ensemble Learning

1.0 Introduction

Named entity recognition (NER) whose task is to classify named entities (NEs) into predefined categories (Sharma *et al*., 2022) such as names of persons, organizations, and locations, among others (Li *et al*., 2022), from textual data serves as a one of the important aspect in the NLP pipeline processing, which plays a vital role in NLP applications including question answering (Diefenbach *et al*., 2018; Rogers *et al*., 2023; Lewis *et al*., 2019), information retrieval system (Gupta and Dixit, 2023; Eswaraiah and Syed, 2023), linking of entities (Al-Moslmi *et al*., 2020; Santana *et al*., 2023; Mulwad *et al*., 2023), and search engines (Riaz, 2018; Ssemwogerere *et al*., 2023; Orekhov *et al*., 2023). In resource-rich languages, NER benefits from abundant annotated datasets and relatively well-distributed classes, leading to strong results. Whereas, for a low-resource language like Khasi spoken by around one million people in North East India, the situation is markedly different. Besides the rise in digitizing textual data, unique challenges remain for entity extraction, such as low quality digitization, inconsistent spellings, limitations of OCR and the lack of textual data covering diverse topics, the challenges are further compounded with the unavailability of an annotated NER dataset where annotation has to be done manually from scratch, which introduces the problem of class imbalance, where frequent entities like GPE (9,436 instances) overshadow less common ones like WORK OF ART (15 instances), hindering the effectiveness of standard NER methods. Ambiguity and multi-word entities are another challenge when working with Khasi NER, e.g., “Ka Phan Nonglait,” which can be a location or a person’s name. Thus, contextual understanding of the language is needed to be able to

disambiguate and produce the correct label. An additional issue is the presence of nested entities, an entity inside another entity, e.g., “U Soso Tham Auditorium” where “U Soso Tham” is a person and “U Soso Tham Auditorium” can be labelled as FAC. In our previous studies, we utilized pre-trained models like RoBERTa, XLM-RoBERTa, and BERT-base-multilingual-cased for Khasi NER through transfer learning (TL) fine-tuned based on the annotated Khasi NER dataset consisting a total of 119,932 annotations (Hoojon, 2025), achieving some encouraging results of 91.25%, 91.17%, and 90.39%, respectively, however, these initiatives have drawbacks which fail to positively identify rare entity types, a limitation that is amplified by unbalanced datasets. This goes to show that relying solely on single models may not be a suitable approach, indicating the need for more creative approaches tailored to the specific requirements of low-resource situations. Thus, in this paper, we investigate a different method using ensemble techniques as a way to enhance the performance of the Khasi NER. Using our previously fine-tuned models, we combined them together, with the intention of leveraging their complementary advantages to boost predictive accuracy and address class imbalances. Our contributions in this paper include the following:

1. Showcasing the effectiveness of ensemble methods in improving NER outcomes for low-resource languages.
2. Substantially enhancing the identification of rare entities despite class imbalance.
3. Presenting a generalizable framework that can guide NER advancements for other under-resourced languages.

2.0 Related Work

The use of multiple models to take advantage of their strengths and produce better predictive performance has been a founding principle of machine learning and ensemble learning in particular, going back many years in applications such as classification, time series forecasting, and, more recently, NER. Ensemble methods, which combine models using

methods such as stacking and voting, overcome the shortcomings and limitations of a single classifier, such as overfitting, bias, and sensitivity to data scarcities. This section summarizes some of the more recent developments in ensemble learning, including their datasets, methods, results, and conclusions, to provide background for the proposed study.

Ensemble methodologies utilizing XGBoost are extensively examined within diverse application domains owing to their capability to capture complex nonlinear relationships, thereby significantly augmenting predictive accuracy. XGBoost is helpful in feature selection and gaining insight into an understanding of relationships in the data, as it can help find those features in the dataset that are most significant (Banga *et al*., 2021). XGBoost has also been used in stock market prediction (Naik and Mohan, 2021) and sales forecasting (Shilong *et al*., 2021). XGBoost is frequently applied to binary and multi-class classification tasks, and has been utilized, for example, for real-time accident detection (Parsa *et al*., 2020), customer churn prediction (Tang *et al*., 2020), credit risk analysis (Li *et al*., 2022; Liu *et al*., 2022; Wang *et al*., 2022), and medical diagnosis (Ogunleye and Wang, 2019). The class imbalance issue is dealt with through the modeling process via loss function weighting among other hyperparameters already available in XGBoost (Wang *et al*., 2020). Among others, XGBoost has been applied to class imbalance issues in medicine by (Létinier *et al*., 2021) for rare adverse drug reactions and to multiple imbalanced datasets (Zuech *et al*., 2021; Le *et al*., 2022; Mishra *et al*., 2021; Mushava and Murray, 2022).

The paper (Pavlyshenko, 2018) explores the use of multi-level stacking for time series predictions as well as logistic regression and supports the benefits of multi-layered structures for complicated predictions. This research utilizes three datasets published in Kaggle: the "Rossmann Store Sales" contains 1,115,000 records, "Grupo Bimbo Inventory Demand" covers 800,000 stores with sales data, and "Bosch Production Line Performance" has 1,183,747 records with imbalanced labels regarding manufacturing failure. In the case of time

series, first-level predictions obtained from first-level models such as GradientBoosting or ExtraTrees on the validation sets are incorporated as features into second-level models of the Lasso-type linear regression or Random Forests. Temporal integrity is maintained by not performing regular cross-validation but rather splitting based on time. In the case of logistic regression, stacking combines XGBoost classifiers with a second-level linear or Bayesian regression while using undersampling and one-hot encoding to tackle the class imbalance. "Grupo Bimbo" solution implements a three-level stack where the first and second levels are XGBoost and ExtraTrees, and the third layer is a weighted average. Stacking achieves an accuracy of 5-10% better than the accuracy of individual models, while the three-level Grupo Bimbo model reached a top score in Kaggle competitions with a log loss of 0.45. In the case of "Bosch", stacked XGBoost achieves an MCC of 0.35, which is higher than single XGBoost's MCC of 0.30. It is shown that stacking is capable of representing variations of parameters of the models and exploiting heterogeneity of the first-level predictions to be more robust, especially in the case of imbalanced data.

To increase accuracy in classification proposed a probability-weighted voting ensemble (Rojarath and Songpan, 2020). The benchmarks employed in this work are the CPU dataset, with 8,192 instances and 21 features, Hepatitis, with 155 instances and 19 features, Ionosphere with 351 instances and 34 features, Sonar with 208 instances and 60 features, and Vote with 435 instances and 16 features; these datasets include binary and multiclass problems and are from UCI. They combine six base classifiers: Naïve Bayes, Bayesian Networks, Decision Trees, Multilayer Perceptrons, SVM, k-NN with six ensemble techniques: AdaBoost, Bagging, Stacking, Voting, Random Forests, Random Subspace. The new method proposes to weight classifiers according to the accuracy obtained in the training set, computing a weighted average of probabilities to predict a class. Among the experiments, the 3PW-Ensemble, which fuses three out of three classifiers, yields the maximum accuracy

rates of 85.36% over Random Forests 74.99% and other ensembles 4PW: 84.87%. Particularly for imbalanced datasets such as Hepatitis, precision and recall have significant improvements (80% F1 vs. 70% of Random Forests). The study finds probability-weighted voting to be beneficial to improve accuracy when votes are cast according to the true class conditional distributions and finds that it is a flexible technique useful for many classification problems. Performance, but, starts to deteriorate in cases of highly noisy data, indicating that robust to noise weighting schemes are required.

The study (Won and Martins, 2018) examines name recognition in historical text. The two historical corpora utilized are the Mary Hamilton Papers with 1,200 place entities and the Samuel Hartlib Papers with 400 place entities. Five different NER systems, Stanford NER, NER-Tagger, Edinburgh Geoparser, spaCy, and Polyglot, are tested and pooled using an ensemble NER system that uses votes per tool. The ensemble 2-3 votes obtain F1 scores of 70% in all scenarios, and the highest 73.3% F1 on Hartlib letters. Stanford NER is the best for Hartlib (F1 approximately 68%), while Polyglot performs the best for Hamilton (F1 approximately 65%). The ensemble's higher recall of 75% outperforms individual tools. (Ullah *et al*., 2024) addresses NER in a low-resource language, Urdu. UNER, which originally contains 50,692 tokens and includes person, location, organization and miscellaneous entities, has been augmented as UNER-II, which contains a total of 160,132 tokens and a total of 114,912 labeled entities. To supplement the dataset, CWEA produces synthetic entities (47,600 persons, 30,940 locations, 30,900 organization). Four transformer models, BERT-multilingual, RoBERTa-Urdu-small, BERT-base-cased, BERT-large-cased, as well as handwritten RNN and BiLSTM baselines, are all fine-tuned with a 70:10:20 train: validation: test split. BERT-multilingual reports a macro F1 of 98.2% on UNER-II, which is higher than RoBERTa-Urdu-small 88.4%, BERT-base- cased 90.8%, BERT-large-cased 91.6%, RNN 97%, and BiLSTM 96%. When evaluated on the original UNER it achieves

84.6% which is better than the previous RNN-based approach which scored 77%. Performance is greatly improved by CWEA as it diversifies the dataset and achieves error rates of 1-2% for its common entities. The types of errors encountered indicate that this can be improved by using better tokenization techniques, such as Word Piece, and more comprehensive gazetteers. The study highlights BERT-multilingual's adaptability to low-resource languages, with potential applications to Pashto and Persian.

In the medical domain, (Jin *et al*., 2025) tackles NER and relation extraction RE) problems. The four datasets employed are: BioRED (biomedical texts with 26,956 entities and 5,151 relations), RDD (rare disease texts with 4,256 entities and 1,957 relations), ADE (11,070 entities on adverse drug effects), and DIANN (1,656 mentions of disability annotations). PubMed-T5, COMCARE combines the PubMedBERT and PubMed T5 embedding with BiLSTM layers and a CRF to maintain sequence consistency and employs threshold 0.6 collaborative decision fusion which integrates two matrices of emissions. For RE, it uses BERT, PubMedBERT, and PubMed-T5, along with semantic chunking using 512 tokens with a stride of 128 tokens to address long-range dependencies while leveraging NER outputs to inform RE extraction. COMCARE's F1 scores are: 93.76% NER and 68.73% RE on BioRED, 77.86% NER and 86.79% RE on RDD, 82.48% NER on ADE, and 99.36% NER on DIANN; these scores exceed those of baselines such as BERT-BiLSTM-CRF (92.51% NER on BioRED) and LLMs such as GPT-4 (0% on DIANN). (Munthe, 2024) applies stacking to finance forecasting. The analysis employs 335 news records related to the trends of the Indonesian stock market Jakarta Composite Index which are classified as "Up" with 714 records and "Down" with 335 records, resulting in a dataset that has class imbalance. SMOTE creates synthetic "Down" samples in order to re-balance the data. It involves tokenization, stemming and TF-IDF vectorization. Predictions are then made by base models XGBoost, Random Forest and passed to meta models Logistic Regression and LSTM. Two stacking

techniques are explored, using either XGBoost/Random Forest with Logistic Regression or LSTM. Stacking with Logistic Regression obtains 86% accuracy, precision 85-86%, recall 84-87% and F1 85-86% outperforming Random Forest 85%, XGBoost 82%, Logistic Regression 81% and stacking with LSTM achived 83% accuracy. A ROC-AUC score of 0.91% indicates a strong discrimination. SMOTE mitigates majority class bias to enhance the prediction of the minority class. Relative to previous research, such as SVM performance at 84% and CNN-LSTM performance at 85%, this approach performs better given the simplicity and interpretability of Logistic Regression as the meta-model.

3.0 Methodology

The datasets are collected from various online sources, such as newspapers and books, with an effort to cover different ranges of topics, which are then cleaned, standardized for diacritics, and corrected for spelling errors, which are prevalent when downloading online content, and then the tasks of manual annotations are performed across 19 different entity types with a total of 30,776 annotated entities. Our annotators annotated the test set in flat format by labeling spans of entities with the following inputs: “text”: “Shillong”, “start”: 0, “end”: 8, “label”: “GPE”, which we converted to IOB2 format later. The conversion from flat to IOB2 is done by using B to denote the first token of a multi-token entity combined with the type of that entity, for example, B-PERSON for the beginning of a person’s name. Tokens that are inside an entity are labeled with I, for example, I-PERSON, and tokens that are not part of any entity are tagged with O, which stands for outside. This was done for the reasons of better interoperability, less ambiguous definitions of multi-token entities, and better metrics reporting on the level of single tokens. Table 1 shows the IOB2 tagging example for a Khasi sentence “U John na Shillong” and Table 2 provides an overview of the distribution of entities in the Khasi NER Test Dataset.

Table 1: IOB2 Tagging

Token	IOB Tag
U	B-PERSON
John	I-PERSON
na	O
Shillong	I-GPE

Table 2: Entity Distribution with Descriptions and Examples

Entity	Count	Description	Example
GPE	9436	Countries, cities, states	India, Shillong,Delhi
PERSON	6046	People or fictional characters	U Soso Tham, John Smith
ORG	3675	Companies or institutions	Microsoft, FBI
NORP	3195	Nationalities or groups	Republican Party
LOC	2134	Non-GPE locations	Nile River
DATE	1849	Calendar dates	06/09/23
CARDINAL	1631	Numerals	2, Three , ar,lai
MONEY	623	Monetary amounts	$20,₹500, 10 tyngka
LAW	521	Legal documents	Roe v. Wade
TIME	388	Time periods	4 hours , 5 kynta, 10 baje
DAY	303	Days of the week	Monday, Sngi Nyngkong
PRODUCT	239	Physical goods	iPhone, Pizza
EVENT	191	Events or occurrences	FIFA World Cup
QUANTITY	171	Measurements	50kg, 1km
PERCENT	153	Percentages	80%
FAC	126	Facilities	Logan Airport
ORDINAL	82	Ordered numbers	9th, Ninth
LANGUAGE	48	Languages	English, Khasi
WORK_OF_ART	15	Titles of works	Mona Lisa

3.1 Base Models

In our ensemble experiment, we utilized the previously fine-tuned models, including RoBERTa-base, XLM-RoBERTa-base, and BERT-base-multilingual-cased, by combining them with the default Spacy NER layer (DSNER) for entity prediction, which was trained on the Khasi NER dataset featuring a training sample size of 10,756 examples that encompass a total of 119,932 entities (Hoojon, 2025). These transformer models were chosen to investigate

whether there is a notable difference in performance for Khasi NER between monolingual and multilingual approaches.

3.2 Ensemble Learning

In this work, we experimented with three ensemble learning techniques: Majority Voting, Weighted Voting, and Stacking using Logistic Regression and XGBoost, each aimed at utilizing model diversity while tackling the sparse and imbalanced dataset. These approaches are outlined below, adhering to the standard principles of ensemble learning (Ganaie *et al*., 2022).

3.2.1 MAJORITY VOTING

Majority Voting selects the label with the most votes from the models for each token. For a token i, let M = {m_1, m_2, . . . , m_k} denote the set of K models (K = 2 for the two-model ensemble, K = 3 for the three-model ensemble), and $y_m^{(i)} \in$ L be the predicted label by model m, where L = {B-PERSON, I-PERSON, O, B-GPE, . .} is the label set. The number of votes for label l ∈ L is:

$$V_l^{(i)} = \sum_{m \in M} 1(y_m^{(i)} = l) \quad (1)$$

The final prediction is:

$$\hat{y}^{(i)} = \arg\max_{l \in L} V_l^{(i)} \quad (2)$$

In case of ties when multiple labels have the same number of votes, we default to the RoBERTa prediction, as it is the model with the overall best performance. This method ensures robustness by favoring consensus, which is important for low-resource languages where individual models may overfit (Rahimi *et al*., 2019; Chen *et al*., 2022).

3.2.2 Weighted Voting

In weighted voting, the validation performance of each model is assigned a weight, where models with higher accuracy are given priority. We use F1 scores from a held-out Khasi validation set:$W_{RoBERTa}$ = 0.9125, $W_{XLM\text{-}RoBERTa}$ = 0.9117, and W_{BERT} = 0.9039 (for the three-model ensemble). The weighted score for label l ∈ L is:

$$W_l^{(i)} = \sum_{m \in M : y_m^{(i)} = l} w_m \tag{3}$$

The final prediction is:

$$\hat{y}^{(i)} = \arg\max_{l \in L} W_l^{(i)} \tag{4}$$

While this approach leverages the fine-tuned performance of all three models, it also takes advantage by incorporating cross-lingual insights from XLM-RoBERTa and BERT-base-multilingual-cased models. Weighted Voting is particularly effective as it balances model strengths without the need for additional training (Lima *et al*., 2023).

3.2.3 Stacking with Logistic Regression

Stacking trains, a meta-classifier to combine model predictions (Li, 2025), learning which model is most reliable for specific tokens. For token i, the feature vector is $(x^{(i)} = [y_{m_1}^{(i)}, y_{m_2}^{(i)}, \ldots, y_{m_K}^{(i)}]$, where predictions are encoded numerically using a Label Encoder (e.g., O → 0, B-PERSON → 1). A logistic regression (LR) meta-classifier, parameterized by θ, predicts the probability of each label:

$$P(y^{(i)} = l | x^{(i)}, \theta) = \frac{\exp(\theta_l \cdot \phi(x^{(i)}))}{\sum_{l' \in L} \exp(\theta_{l'} \cdot \phi(x^{(i)}))} \tag{5}$$

where,

- $\phi(x^{(i)})$ is the encoded feature vector.
- θ_l represents the parameters (weights) of the Logistic Regression model for label l, learned during training and
- $\exp(\theta_l \cdot \phi(x^{(i)}))$ computes an unnormalized score (logit) for label l.

The final prediction is:

$$\hat{y}^{(i)} = \arg\max_{l \in L} P(y^{(i)} = l | x^{(i)}, \theta) \quad (6)$$

The meta-classifier uses cross-entropy loss during training, which measures the difference between predicted probabilities and true labels by adjusting θ to maximize its potential to handle complex patterns.

3.2.4 XGBoost as a Meta-Classifier

XGBoost, which stands for Extreme Gradient Boosting, is used as the meta-classifier in our NER system to combine predictions of the base models, which include RoBERTa, XLM-RoBERTa and BERT-multiligual-cased, in a stacking ensemble. XGBoost is a scalable and efficient implementation of gradient boosting, which is good at picking up complex patterns and is well suited for combining the outputs of base models in NER (Bentéjac *et al*., 2021; Bartz-Beielstein *et al*., 2023; Biau and Cadre, 2021).

3.2.4.1 Xgboost

XGBoost develops a collection of decision trees, wherein the forecast for an input feature vector x_i (comprising encoded predictions from the base model for a token) is derived from the aggregated outputs across K trees. In the context of multi-class classification of IOB tags, such as “B-PERSON” or “O,” the score for a given class c is calculated as follows:

$$\hat{y}_{i,c} = \sum_{k=1}^{K} f_{k,c}(\mathbf{x}_i) \quad (7)$$

where $f_{k,c}(\mathbf{x}_i)$ is the score from the k^{th} tree for class c, and $\hat{y}_{i,c}$ is the raw score for instance i. Probabilities are obtained via the softmax function:

$$P(y_i = c|\mathbf{x}_i) = \frac{\exp(\hat{y}_{i,c})}{\sum_{j=1}^{C} \exp(\hat{y}_{i,j})} \quad (8)$$

where C is the number of IOB tags. The predicted class is:

$$\hat{c}_i = \arg\max_{c} P(y_i = c|\mathbf{x}_i) \quad (9)$$

Here, $\mathbf{x}_i$ comprises encoded IOB predictions from base models, and classes represent IOB tags for 19 entity types plus “O".

3.2.4.2 Training Objective

XGBoost optimizes an objective function balancing accuracy and complexity:

$$\text{Obj} = \sum_{i=1}^{n} \ell(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{10}$$

where $\ell(y_i, \hat{y}_i)$ is the multi-class log loss:

$$\ell(y_i, \hat{y}_i) = -\sum_{c=1}^{C} \mathbb{1}(y_i = c) \log(P(y_i = c|\mathbf{x}_i)) \tag{11}$$

and $\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$ regularizes the model, with T as the number of leaves, w_j as the leaf weights, γ as the leaf penalty, and λ as the L2 regularization parameter. Leaf weights are computed as:

$$w_j = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \tag{12}$$

where I_j is the set of instances in leaf j, $g_i = \frac{\partial \ell}{\partial \hat{y}_i}$ is the gradient, and $h_i = \frac{\partial^2 \ell}{\partial \hat{y}_i^2}$ is the Hessian (Biau and Cadre, 2021).

3.3 Algorithms

Algorithm 1: Data Preparation

Input: Dataset source with texts and entity annotations

1: Initialize D ← [], T ← []

2: **for** each pair in the parsed dataset **do**

3: **if** the pair has valid text and entities, then

4: Append pair to D, entities to T

5: **else**

6: raise error: “Invalid dataset structure.”

7: **end if**

8: **end for**

9: **return** D, T

Algorithm 1 shows how a dataset and entity annotations are preprocessed and validated to serve as the basis of the ensemble learning pipeline. It populates two lists: D, which holds the valid text-entity pairs, and T, which contains the corresponding entity labels, iterating through the data set while appending valid pairs to D and the respective entities to T, raising an error in case of invalid entries so as to ensure data integrity. This algorithm is key in the preparation of structured data for model predictions and ensemble techniques.

Algorithm 2: Model Loading and Prediction Generation

Input: Dataset D, configurations for base models

1: Initialize dictionaries M (models), P (predictions)

2: **for** each model name, path in configurations **do**

3: M[name] ← load model from path

4: **end for**

5: **for** each model name, model in M **do**

6: P[name] ← model predictions on texts from D

7: **end for**

8: **return** M, P

Algorithm 2 loads predefined base models and generates predictions on dataset D. It bootstraps dictionaries M and P for the models and their predictions, respectively, loading each model and using it for analyzing the texts in D to arrive at predictions. This process is critical in providing the initial predictions needed for majority voting, weighted voting, and stacking.

Algorithm 3: Stacking Meta-Classifier Training

Input: Dataset D, true labels T, models M, predictions P

1: Initialize lists X (stacking features), Y (true labels), L (label set)

2: **for** each example, index in D **do**

3: Extract text and true spans; process with reference model to get tokens

4: Convert true spans to labels → Y; add unique labels to L

5: **for** each model name in M **do**

6: Convert P[name][index] to labels or probabilities → X; update L

7: **end for**

8: **end for**

9: Fit label encoder on L, encode X and Y

10: Train meta-classifier (logistic regression or XGBoost) on encoded X, Y

11: **return** meta-classifier, label encoder

Algorithm 3 implements the stacking ensemble approach by transforming the base model predictions into features and training a meta-classifier, logistic regression, or XGBoost, which combines the base predictions into a single output. It initializes lists X for stacking features, the list of true labels Y, and the list of unique labels L, by iterating through each example in D to produce the true spans and transforming the predictions of the basemodel contained in P into features. A label encoder is fitted to encode X and Y, enabling the training of the meta-classifier on these features.

Algorithm 4: Chunking and Ensemble Prediction

Input: Dataset D, true labels T, models M, predictions P, meta-classifier, label encoder, predefined weights L = (l1, l2, l3), test dataset Tb

1: Initialize empty list C for labeled chunks; position ← 0

2: Split D into sentences; compute sentence embeddings S

3: **while** position < length of D **do**

4: Determine chunk end by semantic similarity in S and entity spans in T

5: Extract chunk text D[position:end]; tokenize using reference model

6: **For** each model, collect predictions from P and convert to token-level labels

7: Apply ensemble methods:

8: - Majority Voting: Select label with the most votes per token

9: - Weighted Voting: Select label with the highest weighted sum using L

10: - Stacking: Use a meta-classifier to predict labels from encoded predictions

11: Append (chunk text, final labels) to C

12: Update position to chunk end

13: **end while**

14: Repeat steps 2–12 for Tb to generate test predictions Ct

15: **return** C, Ct

Algorithm 4 employs the ensemble methods majority voting, weighted voting, and stacking by partitioning the dataset D into semantically related segments and assigning final labels to every token in each segment, and processing the test dataset T_b in a similar way. It maintains a list C of labeled chunks, computes sentence embeddings for segmentation, and for each chunk, it tokenizes the chunk text and collects predictions from P. The algorithm uses majority voting to compute the most common label, weighted voting where the weights are given to compute a weighted sum, and stacking, where it applies the meta-classifier to make the predictions. Labeled chunks are stored in C, and the process is repeated for T_b to produce test predictions C_t.

Algorithm 5: Evaluation and Output

Input: Chunks C, true labels T, test chunks Ct, test labels Tt

1: Validate that C and T correspond to the same text spans; if Ct and Tt are provided, validate similarly

2: Initialize an empty dictionary M for evaluation metrics

3: **if** T not in IOB2 format then

4: Convert T to IOB2 format by mapping true labels to chunk text spans

5: **end if**

6: **for** each chunk, true label in C, T **do**

7: Compute micro-averaged precision, recall, and F1 score for predicted vs. true IOB2 labels

8: Optionally compute per-entity-type scores (e.g., for “Person”, “Organization”)

9: Store scores in M (e.g., M[“precision”], M[“Person_f1”])

10: **end for**

11: **if** Ct and Tt are provided **then**

12: **if** Tt not in IOB2 format **then**

13: Convert Tt to IOB2 format

14: **end if**

15: **for** each chunk, true label in Ct, Tt **do**

16: Compute micro-averaged precision, recall, and F1 score

17: Store scores in M (e.g., M[“test_precision”])

18: **end for**

19: **end if**

20: **return** M (precision, recall, F1 scores for training and test data)

Here, Algorithm 5 constitutes the last component of the ensemble workflow, which aims to perform a quantitative analysis of the achieved predicted chunks against the ground-truth labels in order to validate the strength of the entity recognition and chunking pipeline. The

algorithm takes as input two main parameters, C, the predicted chunks that result from the ensemble techniques of Algorithm 4 (i.e., majority, weighted voting, stacking), and T, the true labels from the data preparation of Algorithm 1, which represent the ground-truth entity annotations. It optionally takes Ct (test predictions) and T_t (test true labels) for performance evaluation on unseen data. The algorithm starts by checking if the predicted chunks in C align with the text spans in T (and chunks in C_t and spans in T_t, if applicable). A new empty dictionary M is initialized as a place to store metrics for evaluation. If T (or T_t) is not in IOB2 format, it is first converted to IOB2 by tagging spans of text covered by chunks with the corresponding IOB2 tags, as it matches the predicted in C, which are also IOB2. The core evaluation iterates over each chunk and true labels, averaging precision, recall and F1 at the level of individual characters for the purpose of describing the accuracy of predicted IOB2 labels in comparison to true labels using the micro-averaging method. Per-entity type scores (for example "Person" or "Organization") can optionally be computed as well. These metrics are recorded in M with keys like M["precision"] or M["Person_f1"]. When C_t and T_t are available for test data, a parallel computation of the same metrics is carried out and stored separately (for example, M["test_precision"]). The algorithm produced an output M, encapsulating a comprehensive set of performance metrics. The following formulas were used to calculate precision, recall, and F1 score (Jurafsky and Martin, 2020):

$$\text{Precision} = \frac{TP}{TP + FP} \tag{13}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{14}$$

$$F_1 = 2 \cdot \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{15}$$

4.0 Experimental Setup

To improve our predictions and make them more reliable, we used ensemble methods in two settings: one with two models (RoBERTa and XLM-RoBERTa-base) and another with three

models (RoBERTa, XLM-RoBERTa-base, and BERT-base-multilingual-cased). We tested these on a dataset with 2,675 examples from 19 different categories, totaling 30,766 labeled entities. These pre-trained models were fine-tuned for the Khasi NER task. The goal of the experiment was to evaluate the different models both separately and in combination by means of several ensemble techniques: majority voting, weighted voting, and stacking via meta-classifiers such as logistic regression (LR) and XGBoost. For each model, the input text was processed by passing it through the corresponding tokenizers of the models, and the predictions were made at the span level, which were then converted to predictions at the token level by using character offsets. For the ensemble methods, we used three different strategies for aggregating predictions of the models. In cases of ties, the selection would default to the label suggested by RoBERTa, which is the best-performing model. In weighted voting, the predictions were aggregated according to the accuracy of each model, with weights equal to 0.9125, 0.9117, and 0.9039 for RoBERTa, XLM-RoBERTa, and BERT-multilingual, respectively. The stacking meta-classifiers, which were trained on the predictions of the base models, provide predictions for the final token label. Evaluation is done on standard NER metrics precision, recall, and F1 score calculated at the token level. Figure 1 through 4 shows the system setup for the ensemble method.

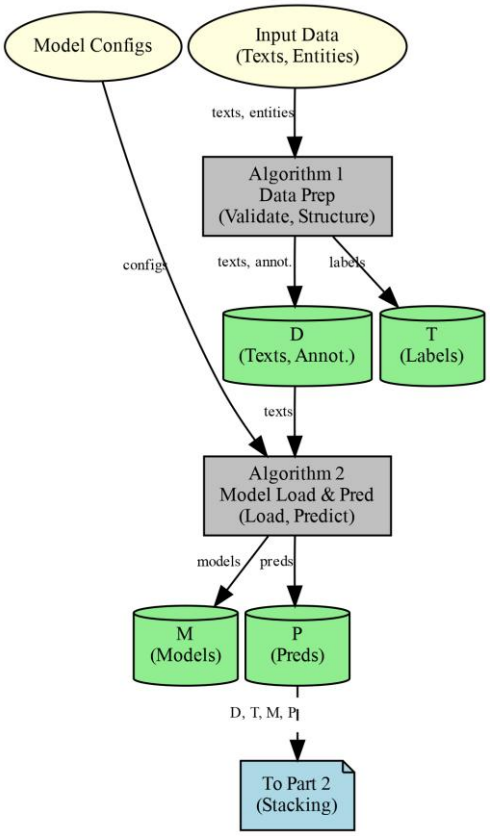

Figure 1: Text–entity preprocessing and validation

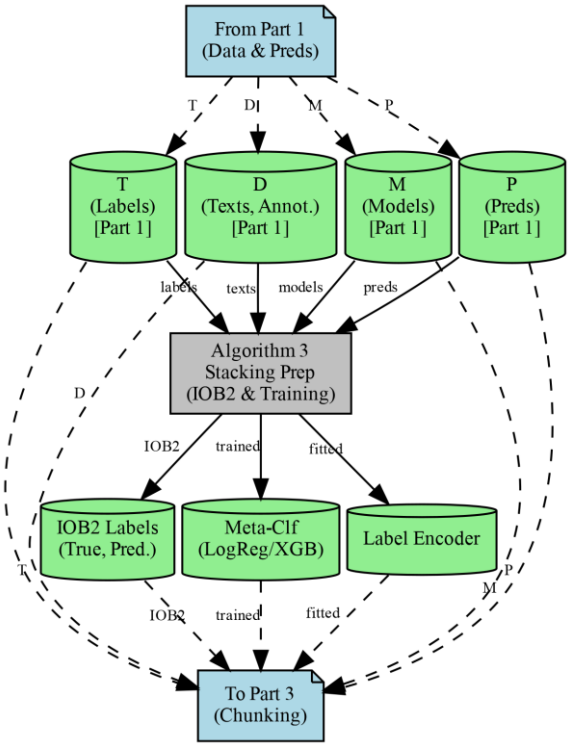

Figure 2: Base model loading and prediction generation pipeline

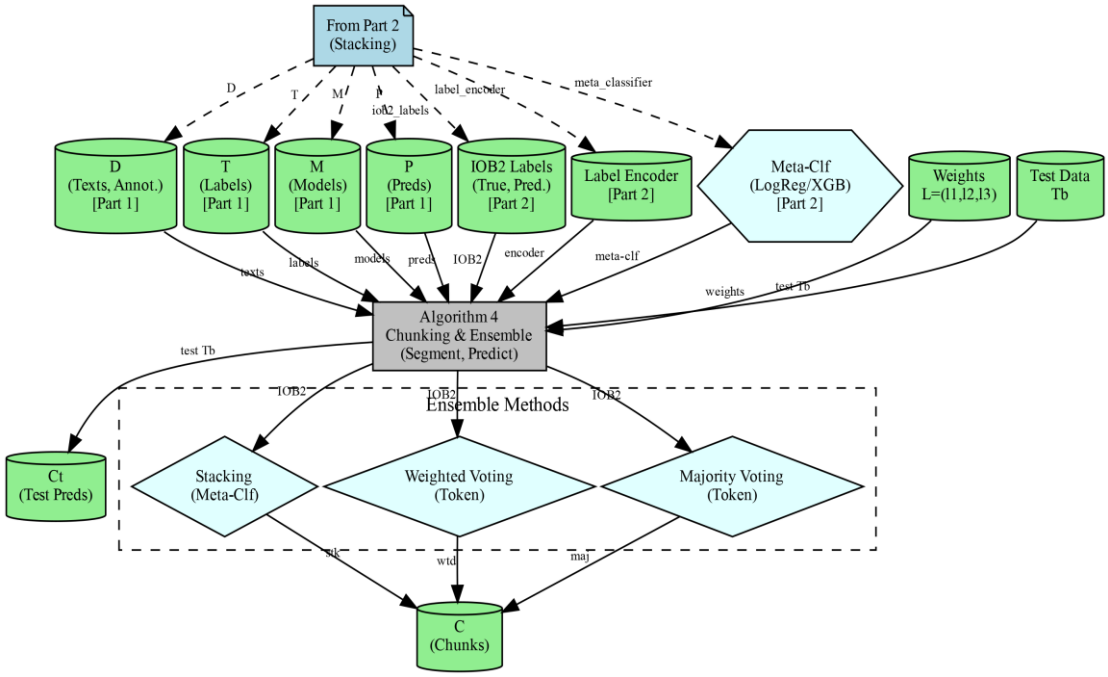

Figure 3: Ensemble labeling with segmentation and voting methods

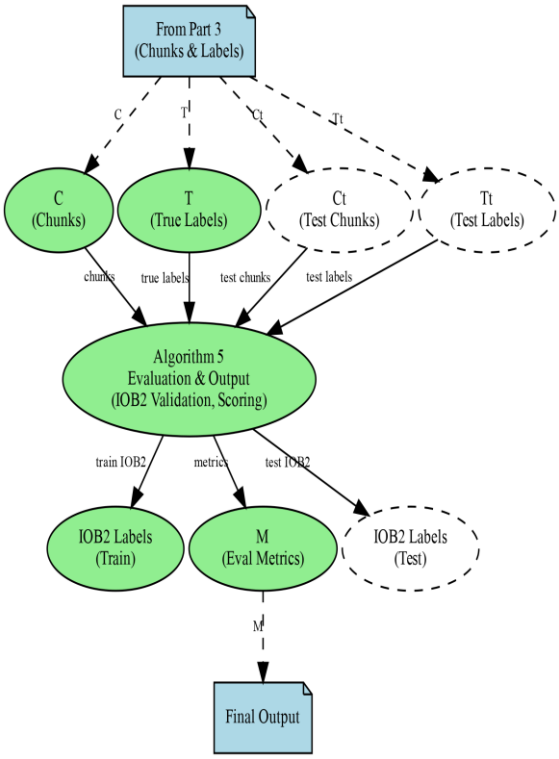

Figure 4: Evaluation of ensemble predictions using precision, recall, and F1

5.0 Results and Discussion

Table 3: Model descriptions

Abbreviation	Description
RoBERTa	Robustly optimized BERT pretraining approach
XLM-R	Cross-lingual RoBERTa
BERT-M	Multilingual BERT
Maj-2	Majority Voting with 2 models
Wgt-2	Weighted Voting with 2 models
Stk-2 (LR)	Stacking Ensemble with 2 models (Logistic Regression)
Stk-2 (XGBoost)	Stacking Ensemble with 2 models (XGBoost)
Maj-3	Majority Voting with 3 models
Wgt-3	Weighted Voting with 3 models
Stk-3 (LR)	Stacking Ensemble with 3 models (Logistic Regression)
Stk-3 (XGBoost)	Stacking Ensemble with 3 models (XGBoost)

Referring to Table 3, we evaluated the performance of the ensemble techniques on Khasi NER dataset by comparing with the baseline models. From the results, it is evident that 2-model (RoBERTa and XLM-RoBERTa) and 3-model (RoBERTa, XLM-RoBERTa, and BERT-multilingual-cased) ensembles for Named Entity Recognition (NER) outperform individual baseline models considerably, but also indicate that their effectiveness is dependent on the specific ensemble approach, as well as on the entity type. The baseline models, RoBERTa with F1-score 91.25%, XLM-RoBERTa with F1-score 91.17%, and BERT-multilingual-cased with F1-score 90.39%, have already excellent results, with RoBERTa reaching the best performance overall. In the 2-model ensembles, majority voting (Maj-2) and weighted voting (Wgt-2) show the same performance with an F1-score of 93.29% (+2.04% respect to RoBERTa), and XGBoost stacking (Stk-2 XGBoost) outperforms both with an F1-score of 93.65% (+2.40%). Logistic regression stacking (Stk-2), on the other hand, performs very poorly with F1-score of 54.19% (-37.06%) and proves to be an inappropriate meta-learner. The 3-model ensembles perform even better, with majority and weighted voting (Maj-3/Wgt-3) obtaining an F1-score of 93.54% (+2.29%) and XGBoost stacking (Stk-3 XGBoost) reaching the highest F1-score of 94.06% (+2.81%) as an attestation of XGBoost's

ability to exploit complementary predictions provided by other models. Stk-3 with logistic regression, still remains ineffective with F1-sore of 62.87% (-28.38%). For high frequency entities like PERSON with F1-score of 96.44% to 96.86%, GPE with F1-score of 94.19% to 94.64% and DATE with F1-score of 97.47% to 98.49% the per-entity analysis shows good performance, with Stk-3 XGBoost prevailing in all cases. Rare entities such as WORK_OF_ART with F1-score of 34.29% to 38.36%, PRODUCT with F1-score of 62.47% to 68.81% and FAC with F1-score of 66.99% to 77.79% all have low F1 scores due to support, which ranged from 57 to 355, where Stk-3 XGBoost only performed slightly better. Logistic regression fails across multiple entities (e.g., F1-score of 0.0% for FAC, PRODUCT), highlighting its limitations. The 3-model ensembles marginally outperform than the 2-model ones, especially for entities like PERCENT with F1-score of 95.45% vs. 94.01% and FAC with F1-score of 77.79% vs. 73.94%, suggesting that BERT-multilingual-cased makes complementary predictions even if it is a weaker baseline. Micro-averaged F1-scores (93.26%-94.02%) reflect the prevalence of entities with high support, whereas macro-averaged F1-scores (84.48% -86.69%) suggest that Stk-3 XGBoost performs better at classifying rare entities. In view of these results, we can state that the XGBoost stacking, especially in the case of 3 models, is the best approach to improve NER accuracy, but majority as well as weighted voting are also simpler, yet successful, alternatives. Table 4 compares the performance of the various ensemble approaches with the three base models in terms of overall precision, recall, and F1 scores.

Table 4: Overall Performance Comparison of Individual and Ensemble Models

Method	**Models Included**	**Precision**	**Recall**	**F1**	**Others vs. RoBERTa**
RoBERTa	RoBERTa	91.12	91.39	91.25	---
XLM-R	XLM-RoBERTa	90.55	91.79	91.17	-0.08
BERT-M	BERT-multilingual-cased	90.24	90.54	90.39	-0.86
Maj-2	RoBERTa, XLM-RoBERTa	92.88	93.7	93.29	+2.04

Wgt-2	RoBERTa, XLM-RoBERTa	92.88	93.7	93.29	+2.04
Stk-2(LR)	RoBERTa, XLM-RoBERTa	54.74	53.65	54.19	-37.06
Stk-2 (XGBoost)	RoBERTa, XLM-RoBERTa	93.58	93.71	93.65	+2.4
Maj-3	RoBERTa, XLM-R, BERT-M	93.36	93.72	93.54	+2.29
Wgt-3	RoBERTa, XLM-R, BERT-M	93.36	93.72	93.54	+2.29
Stk-3 (LR)	RoBERTa, XLM-R, BERT-M	63.74	62.02	62.87	-28.38
Stk-3 (XGBoost)	RoBERTa, XLM-R, BERT-M	94.2	93.92	94.06	+2.81

For prediction on new sentences, we intentionally selected rare entities to test the performance of the various ensemble methods:

Sentence 1 features the rare entity *Ka Akor Kaba Tam Bynta-III* (name of a book), a WORK_OF_ART. Only the 2-model and 3-model Stacking with XGBoost accurately classified this entity. No other ensemble methods, were able to identify it. For instance, the phrase "*la pyllait paidbah ia ka kot [Ka Akor Kaba Tam Bynta-III] da u Kongsan Rangbah…"* shows a true positive only when stacked with XGBoost. This performance demonstrates the higher capability of the XGBoost model, when compared to simpler ensemble strategies, to understand the contextual subtleties required for identifying complicated multi-token named entities.

Sentence 2 also investigates the distribution of the ordinal word *ba ar* (meaning second), which is labeled with the ORDINAL category, e.g., '*...la tip kum ka bneng '[ba ar]*...' and *'..ka bneng [ba ar] sha pyrthei...'*. All of the ensemble methods correctly classified them except for 2-model and 3-model (stk-2(LR) and stk-3(LR)).

In Sentence 3, we tested with the entity LANGUAGE, e.g., *'...ka Jaka Hikai Ktien [Khasi] ne ka Khasi Learning Centre.....ka Kim kam nang ban kren [Khasi]…'.* All ensemble methods, 2-model and 3-model variants of majority, weighted voting, as well as XGBoost stacking, were able to successfully classify every instance of the positive class. The performance is relatively consistent in this case, which indicates that the ensemble models are strong when

rare items occur in very indicative contexts, such as language names with the verb *"ktien/kren"* which means *"language/speak"*, with the exception of logistic stacking (stk-2(LR) and stk-3(LR)), which failed to identify the entity correctly. Plots of various model performances over all ensemble settings and per-entity predictions are shown in Figure 5 and Figure 6.

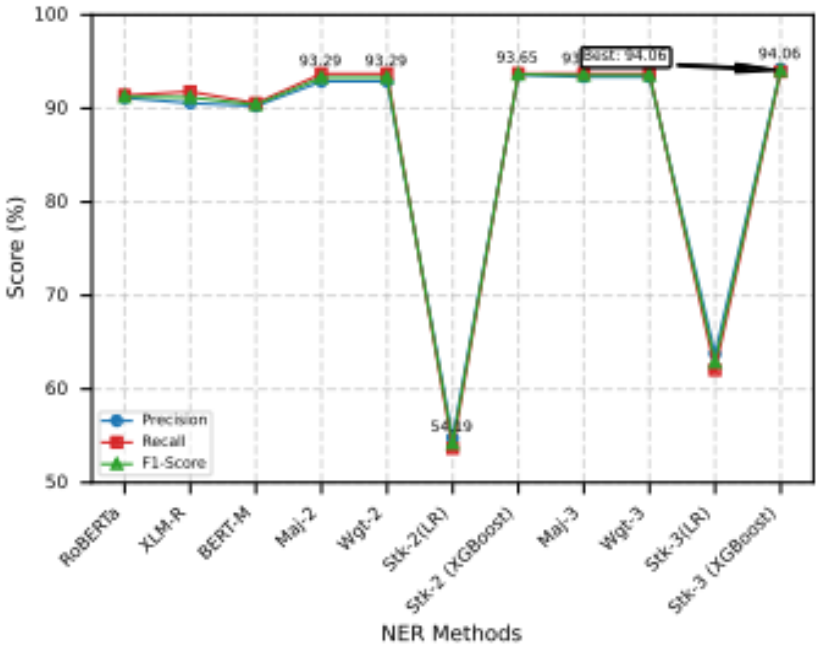

Figure 5: Evaluation metrics across ensemble and baseline models

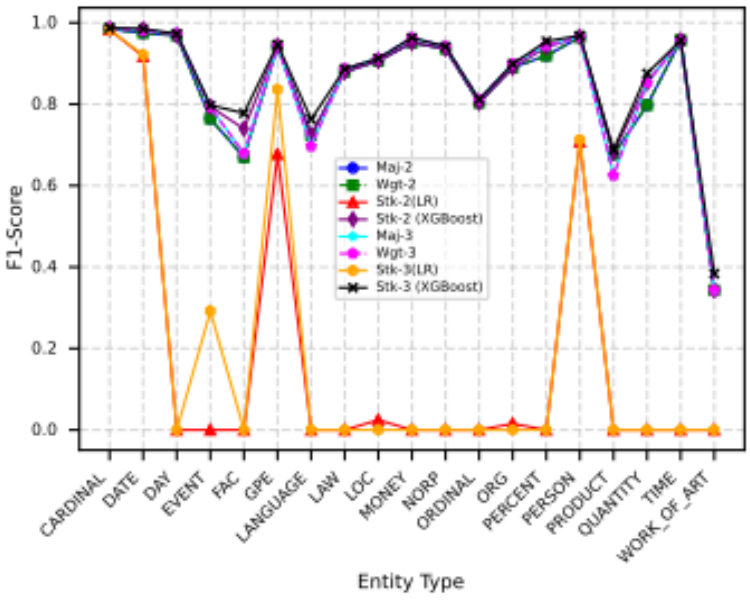

Figure 6: Entity-wise precision, recall, and F1 of ensemble models

6.0 Conclusion

In conclusion, this paper aims to investigate whether ensemble learning methods can help to improve NER system performance on the Khasi language. Our ensemble learning approach is based on our previous work on individual transformer models, RoBERTa, XLM-RoBERTa, and BERT-multilingual-cased trained on Khasi NER datasets. The experiment showed improved performance over individual models; RoBERTa, the strongest baseline model with an F1-score of 91.25%, was outperformed by XGBoost as a meta-learner using the three base learners, with an improvement of +2.81%. This experiment also showed an improvement in detecting rare and under-represented entities such as WORK_OF_ART, LANGUAGE, ORDINAL, and EVENT, even though the dataset is skewed towards other classes such as GPE, PERSON, ORG, NORP, LOC, DATE, and CARDINAL entities. Findings also show that logistic regression as a meta-learner performs very poorly, and other limitations, such as an imbalanced dataset and overdependence on other high-resource languages as foundational models. Our future goal is to increase the dataset size, covering a wider range of topic domains with well-balanced classes, adopting data augmentation, and exploring more sophisticated meta-learners or methods that are best suited to the language. Overall, our experimental approach shows that ensemble learning is a promising and scalable strategy for improving NER performance in low-resource languages such as Khasi.

REFERENCES

Sharma, A., Chakraborty, S., Kumar, S. *et al.* (2022). Named entity recognition in natural language processing: a systematic review. In: *Proceedings of the Second Doctoral Symposium on Computational Intelligence*. Singapore: Springer, pp. 817–828. https://doi.org/10.1007/978-981-16-3647-9_59

Li, J., Sun, A., Han, J. and Li, C. (2022). A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1), pp. 50–70. https://doi.org/10.1109/tkde.2020.2981314

Diefenbach, D., Lopez, V., Singh, K. and Maret, P. (2018). Core techniques of question answering systems over knowledge bases: a survey. *Knowledge and Information Systems*, 55(3), pp. 529–569. https://doi.org/10.1007/s10115-017-1100-y

Rogers, A., Gardner, M. and Augenstein, I. (2023). QA dataset explosion: a taxonomy of NLP resources for question answering and reading comprehension. *ACM Computing Surveys*, 55(10), pp. 1–45. https://doi.org/10.1145/3560260

Lewis, P., Oğuz, B., Rinott, R., Riedel, S. and Schwenk, H. (2019). MLQA: evaluating cross-lingual extractive question answering. *arXiv*. Available at: https://arxiv.org/abs/1910.07475

Gupta, V. and Dixit, A. (2023). Recent query reformulation approaches for information retrieval system – a survey. *Recent Advances in Computer Science and Communications*, 16(1), pp. 94–107. https://doi.org/10.2174/2666255815666220404091920

Eswaraiah, P. and Syed, H. (2023). An efficient ontology model with query execution for accurate document content extraction. *Indonesian Journal of Electrical Engineering and Computer Science*, 29(2), pp. 981–989. https://doi.org/10.11591/ijeecs.v29.i2.pp981-989

Al-Moslmi, T., Ocaña, M.G., Opdahl, A.L. and Veres, C. (2020). Named entity extraction for knowledge graphs: a literature overview. *IEEE Access*, 8, pp. 32862–32881. https://doi.org/10.1109/ACCESS.2020.2973928

Santana, B., Campos, R., Amorim, E., Jorge, A., Silvano, P. and Nunes, S. (2023). A survey on narrative extraction from textual data. *Artificial Intelligence Review*, 56(8), pp. 8393–8435. https://doi.org/10.1007/s10462-022-10338-7

Mulwad, V., Finin, T., Kumar, V.S., Williams, J.W., Dixit, S. and Joshi, A. (2023). A practical entity linking system for tables in scientific literature. *arXiv*. https://doi.org/10.48550/arXiv.2306.10044

Riaz, K. (2018). Improving search via named entity recognition in morphologically rich languages: a case study in Urdu. PhD thesis, University of Minnesota. Available at: https://hdl.handle.net/11299/195403

Ssemwogerere, R., Sajo, A.A., Mutwalibi, N. and Mzee, A.K. (2023). A survey about the application of artificial intelligence in search engines. In: *Handbook of Research on AI Methods and Applications in Computer Engineering*. Hershey, PA: IGI Global, pp. 161–178. https://doi.org/10.4018/978-1-6684-6937-8.ch008

Orekhov, S., Godlevsky, M., Malyhon, H. and Goncharenko, T. (2023). A new method of search engine optimization based on semantic kernel idea. In: *Advances in Artificial Systems for Medicine and Education VI*. Cham: Springer, pp. 67–77. https://doi.org/10.1007/978-3-031-24468-1_7

Hoojon, R. (2025). Optimizing low-resource Khasi NER through transfer learning. In: *Proceedings of the International Conference on North East India AI Summit (NEIAIS-2025)*. Manipur, India

Banga, A., Ahuja, R. and Sharma, S.C. (2021). Performance analysis of regression algorithms and feature selection techniques to predict PM2.5 in smart cities. *International Journal of System Assurance Engineering and Management*, pp. 1–14

Naik, N. and Mohan, B.R. (2021). Novel stock crisis prediction technique: a study on Indian stock market. *IEEE Access*, 9, pp. 86230–86242. https://doi.org/10.1109/ACCESS.2021.3088999

Shilong, Z. *et al.* (2021). Machine learning model for sales forecasting by using XGBoost. In: *IEEE International Conference on Consumer Electronics and Computer Engineering*. pp. 480–483

Parsa, A.B., Movahedi, A., Taghipour, H., Derrible, S. and Mohammadian, A.K. (2020). Toward safer highways: application of XGBoost and SHAP for real-time accident detection. *Accident Analysis & Prevention*, 136, p. 105405. https://doi.org/10.1016/j.aap.2019.105405

Tang, Q., Xia, G., Zhang, X. and Long, F. (2020). A customer churn prediction model based on XGBoost and MLP. In: *International Conference on Computer Engineering and Application*. pp. 608–612. https://doi.org/10.1109/ICCEA50009.2020.00133

Li, Y., Stasinakis, C. and Yeo, W.M. (2022). A hybrid XGBoost-MLP model for credit risk assessment. *Forecasting*, 4(1), pp. 184–207

Liu, J., Zhang, S. and Fan, H. (2022). A two-stage hybrid credit risk prediction model. *Expert Systems with Applications*, 195, p. 116624. https://doi.org/10.1016/j.eswa.2022.116624

Wang, K., Li, M., Cheng, J., Zhou, X. and Li, G. (2022). Research on personal credit risk evaluation based on XGBoost. *Procedia Computer Science*, 199, pp. 1128–1135. https://doi.org/10.1016/j.procs.2022.01.143

Ogunleye, A. and Wang, Q.G. (2019). XGBoost model for chronic kidney disease diagnosis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 17(6), pp. 2131–2140

Wang, C., Deng, C. and Wang, S. (2020). Imbalance-XGBoost: leveraging weighted and focal losses. *Pattern Recognition Letters*, 136, pp. 190–197. https://doi.org/10.1016/j.patrec.2020.05.035

Létinier, L. *et al.* (2021). Artificial intelligence for unstructured healthcare data. *Clinical Pharmacology & Therapeutics*, 110(2), pp. 392–400. https://doi.org/10.1002/cpt.2266

Zuech, R., Hancock, J. and Khoshgoftaar, T.M. (2021). Detecting web attacks using random undersampling. *Journal of Big Data*, 8(1), p. 75

Le, T.T.H., Oktian, Y.E. and Kim, H. (2022). XGBoost for imbalanced multiclass classification-based intrusion detection systems. *Sustainability*, 14(14), p. 8707

Mishra, M., Patnaik, B., Bansal, R.C., Naidoo, R., Naik, B. and Nayak, J. (2021). DTCDWT-SMOTE-XGBoost-based islanding detection. *IEEE Systems Journal*, 16(2), pp. 2008–2019

Mushava, J. and Murray, M. (2022). A novel XGBoost extension for credit scoring. *Expert Systems with Applications*, 202, p. 117233. https://doi.org/10.1016/j.eswa.2022.117233

Pavlyshenko, B. (2018). Using stacking approaches for machine learning models. In: *IEEE International Conference on Data Stream Mining and Processing*. pp. 255–258. https://doi.org/10.1109/DSMP.2018.8478522

Rojarath, A. and Songpan, W. (2020). Probability-weighted voting ensemble learning. *Journal of Advances in Information Technology*, 11(4), pp. 217–227. https://doi.org/10.12720/jait.11.4.217-227

Won, M. and Martins, B. (2018). Ensemble named entity recognition. *Frontiers in Digital Humanities*, 5. https://doi.org/10.3389/fdigh.2018.00002

Ullah, F., Gelbukh, A., Zamir, M., Riverón, E. and Sidorov, G. (2024). Enhancement of named entity recognition in low-resource languages. *Computers*, 13(10), p. 258. https://doi.org/10.3390/computers13100258

Jin, M., Choi, S.M. and Kim, G.W. (2025). COMCARE: a collaborative ensemble framework. *Electronics*, 14, p. 328. https://doi.org/10.3390/electronics14020328

Munthe, I. (2024). Implementation of stacking technique combining machine learning and deep learning algorithms. *Journal of Applied Data Sciences*, 5, pp. 2079–2091. https://doi.org/10.47738/jads.v5i4.421

Ganaie, M.A., Hu, M., Malik, A.K., Tanveer, M. and Suganthan, P.N. (2022). Ensemble deep learning: a review. *Engineering Applications of Artificial Intelligence*, 115, p. 105151. https://doi.org/10.1016/j.engappai.2022.105151

Rahimi, A., Li, Y. and Cohn, T. (2019). Massively multilingual transfer for NER. *arXiv*. Available at: https://arxiv.org/abs/1902.00193

Chen, Y., Zhong, R., Zha, S., Karypis, G. and He, H. (2022). Meta-learning via language model in-context tuning. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*. pp. 719–730. https://doi.org/10.18653/v1/2022.acl-long.53.

Lima, K.A. *et al.* (2023). A novel data and model centric artificial intelligence approach for Bengali NER. *PLOS ONE*, 18(9), p. e0287818. https://doi.org/10.1371/journal.pone.0287818

Li, Y. (2025). Enhanced logistic regression using stacking algorithm. *Highlights in Science, Engineering and Technology*, 136, pp. 1–11. https://doi.org/10.54097/xmphgz15

Bentéjac, C., Csörgő, A. and Martínez-Muñoz, G. (2021). A comparative analysis of gradient boosting algorithms. *Artificial Intelligence Review*, 54(3), pp. 1937–1967. https://doi.org/10.1007/s10462-020-09896-5

Bartz-Beielstein, T., Chandrasekaran, S. and Rehbach, F. (2023). Case study II: tuning of gradient boosting (XGBoost). In: *Hyperparameter tuning for machine and deep learning with R*. Singapore: Springer, pp. 221–234. https://doi.org/10.1007/978-981-19-5170-1_9

Biau, G. and Cadre, B. (2021). Optimization by gradient boosting. In: *Advances in Contemporary Statistics and Econometrics*. Cham: Springer, pp. 23–44. https://doi.org/10.1007/978-3-030-73249-3_2

Jurafsky, D. and Martin, J.H. (2020). *Speech and language processing*. 3rd ed. Stanford: Stanford University