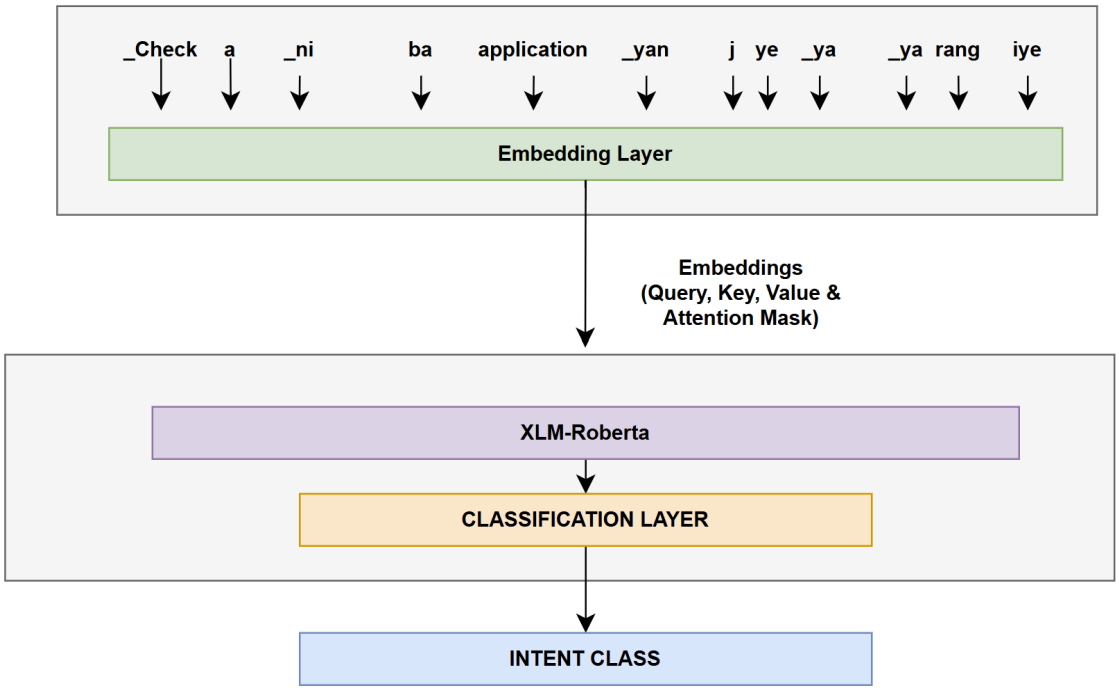
Intent Classification for Multilingual Voice AI: Written Summary

1. Overall Approach

I designed an intent classification solution for a multilingual Voice AI system, focusing on English (en), Kinyarwanda (rw), and mixed-language utterances. My approach is centred on fine-tuning XLM-RoBERTa (xlm-roberta-base), a pretrained multilingual Transformer, to classify user intents from ASR-transcribed utterances.



XLM-Roberta with a classification head for intent classification

Key steps in my workflow include:

1. **Data Exploration**:
    - I analysed the dataset's language distribution (rw: 306, en: 201, mixed: 193) and utterance lengths (mean ≈ 43 characters, max 71).
    - Visualised utterance length distributions to inform the max sequence length (64 tokens) for XLM-R.
2. **Preprocessing**:

    - Tokenized all utterances using XLMRobertaTokenizer.
    - Observed effective subword segmentation across English, Kinyarwanda, and slang/mixed utterances.
    - Retained all subword tokens without explicit language-specific normalization, leveraging XLM-R's shared vocabulary to handle low-resource languages and code-switching.
3. **Modeling**:
    - I implemented a PyTorch-based XLMRIntentClassifier: XLM-R encoder + dropout + linear classification head.
    - Predicted intents using the token embedding.
    - Trained with cross-entropy loss, small learning rate (2e-5), and linear warmup schedule over 5 epochs.
4. **Evaluation**:
    - Metrics: Overall accuracy, macro F1, and language-aware metrics.
    - Visualizations: Bar charts of macro F1 by language and confusion matrices to inspect intent-level errors.
    - Observed that English and mixed-language performance is strong, while Kinyarwanda remains challenging due to low-resource data.

2. Key Assumptions

1. Each utterance has one dominant intent.
2. Transcription errors do not drastically alter intent semantics.
3. XLM-R's multilingual embeddings generalize across languages, including code-switched text.
4. Short utterances (<64 tokens) are sufficient to capture user intent.
5. Fine-tuning for a small number of epochs (5 in this case) provides a reasonable baseline, especially under limited compute constraints.

3. Trade-Offs and Limitations

Aspect	Trade-Off / Limitation	Rationale
Model Size	xlm-roberta-base is large, ~270M params	Balances accuracy with latency; can deploy on GPU/CPU
Training Epochs	Only 5 epochs	Quick experimentation; Kinyarwanda performance may improve with more epochs
Max Sequence Length	64 tokens	Captures nearly all utterances; truncates extremely long ones
Low-resource language	Kinyarwanda macro F1 ~0.42	Data scarcity limits performance; the model alone cannot fully compensate

4. Notes for Reviewers

- XLM-R tokeniser is language-agnostic and handles subword tokenisation for English, Kinyarwanda, and code-switched/slang utterances.
- Evaluation explicitly includes language-aware metrics, which are critical for fairness and public-service deployment.
- Observed gaps in Kinyarwanda performance justify targeted data collection and augmentation.
- Training pipeline includes gradient clipping, warmup scheduler, and device-agnostic execution, reflecting production-awareness.
- Confusion matrices and macro F1 guided insights into intent overlap, ASR noise impact, and code-switching robustness.

5. Responses to Assignment Sections

5.1 Part 2 – Data Strategy

- Cleaned ASR transcripts with minimal preprocessing to preserve code-switching and morphology.
- Handled transcription errors via subword tokenisation (XLM-R) without explicit correction.
- Addressed small/imbalanced datasets via class weighting and language-aware oversampling.
- Plan to improve data quality over time through labelling feedback loops.

5.2 Part 3 – Evaluation

- Metrics: accuracy, macro F1, language-aware accuracy/F1.
- Robustness checks: code-switching, informal phrasing, low-confidence ASR outputs.
- Failure mode analysis via confusion matrices, especially for Kinyarwanda.
- Suitability: model is deployable as a baseline, with fallback mechanisms for low-confidence predictions.

5.3 Part 4 – Production & MLOps

- Model versioning: save checkpoints per best validation macro F1.
- Monitoring: track accuracy and macro F1 over time; watch drift in low-resource languages.
- Real-time integration: tokenise utterance → XLM-R → classifier → intent → action.

- Fallback strategies: flag predictions with low softmax confidence for human review or repeated ASR capture.

5.4 Part 5 – Ethics & Public-Sector Considerations

- Bias/Fairness: explicitly measured performance by language to detect inequities.
- Explainability: subword-level token embeddings and confusion matrices help explain model predictions.
- Responsible AI: ensures low-resource languages are not ignored, mitigating unfair service delivery.

6. Key Observations and Next Steps

- English and mixed-language utterances are handled effectively; Kinyarwanda performance requires more data or targeted augmentation.
- Subword tokenisation allows robust handling of code-switching and slang, critical for Voice AI.
- Future improvements include:
    - Extended training epochs
    - Confidence-based fallback logic
    - Incremental data labelling for Kinyarwanda
    - Production monitoring pipelines

Overall Conclusion:
I successfully implemented a multilingual intent classification system that leverages XLM-R's pretrained representations to handle code-switching, informal phrasing, and low-resource Kinyarwanda utterances. The design explicitly addresses fairness, language imbalance, evaluation transparency, and production considerations, providing a strong foundation for a real-world Voice AI deployment.