Management Summary: Hallucination Span Detection

Task Overview

The project addresses the challenge of detecting hallucination spans—false or unsupported information—within responses generated by multilingual large language models (LLMs). Using the Mu-SHROOM dataset (10 languages, including Mandarin and Finnish), the goal was to identify specific text spans containing inaccuracies. This task is critical for improving LLM reliability in global applications.

Key Challenges

1. **Multilingual Complexity**: Handling structurally diverse languages increased model training complexity.
2. **Tokenization Alignment**: Ensuring tokenized text matched original annotations after preprocessing (e.g., splitting "Fußball" into subwords like ["Fu", "##ß", "##ball"]).
3. **Variable Text Lengths**: Managing inconsistent input lengths while preserving query-response context.
4. **Label Consistency**: Mapping character-level annotations to token-level labels without misalignment.

Solution Implemented

Core Technical Approach: Supervised mBERT Model

The solution employed a multilingual BERT (mBERT) tokenizer to process queries and responses across diverse languages, ensuring compatibility with structurally complex texts like Finnish and Mandarin. To preserve contextual relationships between queries and their corresponding responses, input pairs were concatenated using special tokens `[CLS]` (to mark the start) and `[SEP]` (to separate query from response). Input sequences were standardized to a maximum length of 512 tokens via padding or truncation, with overflowed rows skipped entirely to avoid partial context loss. During training, the model utilized `CrossEntropyLoss`, ignoring labels for query and padding tokens (assigned `-100`), to focus exclusively on response token predictions. Optimization was performed using the AdamW algorithm, coupled with early stopping (3-epoch patience) to mitigate overfitting. Robustness was validated through 5-fold cross-validation, achieving **77% accuracy** and **67% recall**. For evaluation, token-level metrics (precision, recall, F1-score) were computed exclusively on response tokens (post-`[SEP]`), with post-processing logic to map predictions back to original text spans, ensuring alignment despite subword tokenization challenges (e.g., "1977" split into ["19", "77"]).

Future Recommendations

1. **Fine-Tuning**: Optimize mBERT with labeled training data to improve recall.
2. **Hybrid Pipelines**: Combine mBERT predictions with rule-based post-processing for span alignment.
3. **Multilingual Specialization**: Test language-specific models (e.g., XLM-R) for complex languages.
4. **Error Analysis**: Investigate high false negatives in punctuation and numeric tokens.

Code Implementation

The provided Python code implements the supervised mBERT workflow, including:

- **Custom Dataset Class**: Handles tokenization, padding, and label alignment.
- **Cross-Validation**: 5-fold training/evaluation to ensure generalizability.
- **Inference Pipeline**: Extracts response tokens and evaluates predictions.

Code Availability: The full implementation is provided for technical review, including training loops, token alignment logic, and evaluation scripts.