**Language Learning for Grammar Feedback**

1. **Introduction**

This project focuses on building an advanced grammar correction system using state-of-the-art natural language processing (NLP) models. The system integrates multiple sequence-to-sequence (seq2seq) grammar correction models, namely T5, GEC, and Coedit, along with a classification model based on BERT. An ensemble strategy is applied to maximize prediction accuracy by leveraging the strengths of each model. The project addresses key grammar correction challenges such as sentence restructuring, subject-verb agreement, and preposition misuse.

1. **Dataset**

* **Source:** The dataset consists of sentences containing grammar mistakes paired with their corrected versions.
* **Preprocessing:**
  + Duplicate sentences and erroneous entries were removed.
  + Sentences were tokenized to ensure compatibility with model architectures.
  + Incorrect sentences were labeled as 0 and corrected ones as 1 for the classification model.
* **Structure:**
  + Each sentence-correction pair was formatted according to the requirements of each model.
  + For seq2seq models, the dataset included input and target fields.
  + For the classification model, it included text and label fields.

1. **Models Overview**

**3.1 T5 Model**

**Explanation of the T5 Model in the Code**

In this model, we implemented a grammar correction strategy using the T5 (Text-To-Text Transfer Transformer) model, which reformulates all NLP tasks as text-to-text problems. Instead of treating grammar correction as a classification task, T5 generates corrected sentences directly based on input prompts.

1. **Input Formatting Strategy**

The input sentence is prefixed with the instruction "fix grammar:" to guide the model in understanding the objective. This prompt helps T5 differentiate grammar correction tasks from other potential NLP tasks it might be fine-tuned on.

* Example Input: She go to school yesterday.
* Expected Output: She went to school yesterday.

This approach ensures that the model consistently interprets the task as grammar correction.

1. **Dataset Tokenization and Preprocessing**

Each sentence pair (incorrect and corrected) was tokenized using the T5 tokenizer. The tokenization step included:

* Padding: Sentences were padded to a maximum length of 64 tokens.
* Truncation: Longer sentences were truncated to fit the maximum token limit.
* Labels: Target sentences (corrected versions) were also tokenized and used as labels for training.

1. **Fine-Tuning Strategy**

The T5 model was fine-tuned on the prepared dataset with hyperparameters optimized for grammar correction:

* Epochs: 8
* Batch Size: 4
* Learning Rate: 5e-5
* Warmup Steps: 100
* Evaluation Strategy: After every epoch
* Metric Optimization: Minimize validation loss

The model was fine-tuned using the HuggingFace Trainer class, which simplified training and evaluation.

**4. Model Strengths and Advantages**

* Text-to-Text Flexibility: T5 excels at generating fluent and coherent corrected sentences.
* Context Awareness: It captures dependencies across words in a sentence effectively.
* Task Clarity: The "fix grammar:" prefix ensures consistent task interpretation.

**5. Model Limitations**

* Performance on Long Sentences: The token length constraint (64 tokens) can cause loss of contextual information.
* Overfitting on Small Datasets: Fine-tuning on smaller datasets may lead to overfitting, reducing generalizability.

**6. Summary in Simple Terms**

We passed grammatically incorrect sentences, prefixed with "fix grammar:", into the T5 model. Each sentence was tokenized, padded, and truncated before fine-tuning. During training, the model learned to predict corrected sentences. Its text-to-text architecture allowed it to generate high-quality grammar corrections, improving overall sentence fluency and correctness.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**3.2 GECTor Model**

**Explanation of the GEC Model in the Code**

In this model, we implemented a grammar correction strategy using the GECToR (Grammar Error Correction Transformer) model. GECToR focuses specifically on identifying and correcting grammatical errors in sentences with a seq2seq architecture.

**1. Input and Target Strategy**

The input consists of grammatically incorrect sentences, while the target contains the corrected versions. These pairs are used to train the model in a supervised learning setup.

Example Input: She go to school yesterday.

Expected Output: She went to school yesterday.

**2. Dataset Tokenization and Preprocessing**

Each sentence pair (incorrect and corrected) was tokenized using the GECToR tokenizer. Tokenization steps included:

Padding: Sentences were padded to a maximum length of 64 tokens.

Truncation: Longer sentences were truncated.

Label Mapping: Corrected sentences were mapped as target labels.

**3. Fine-Tuning Strategy**

* Epochs: 3
* Batch Size: 2
* Gradient Accumulation Steps: 4
* Precision: Mixed Precision (fp16=True)

**4. Model Strengths and Advantages**

* Excels at explicit grammar corrections.
* Handles shorter sentences effectively.

**5. Model Limitations**

* Struggles with sentence restructuring.
* Limited performance on longer sentences.

**6. Summary in Simple Terms:** The GECToR model tokenizes and processes incorrect sentences to predict corrected outputs. Its seq2seq architecture is optimized for identifying and fixing grammar errors efficiently.

**3.3 Coedit Model**

**Explanation of the Coedit Model in the Code**

In this model, we implemented a grammar correction strategy using the Coedit (Collaborative Editing Transformer) model. Coedit is specifically designed for collaborative editing tasks and excels at maintaining sentence fluency while addressing grammatical issues.

**1. Input and Target Strategy**

The input consists of grammatically incorrect sentences, while the target contains the corrected versions. These pairs are used to train the model in a supervised learning setup.

Example Input: He go to the market.

Expected Output: He goes to the market.

**2. Dataset Tokenization and Preprocessing**

Each sentence pair (incorrect and corrected) was tokenized using the Coedit tokenizer. Tokenization steps included:

Padding: Sentences were padded to a maximum length of 64 tokens.

Truncation: Longer sentences were truncated.

Label Mapping: Corrected sentences were mapped as target labels.

**3. Fine-Tuning Strategy**

The Coedit model was fine-tuned on the prepared dataset with hyperparameters optimized for grammar correction:

Epochs: 15

Batch Size: 8

Learning Rate: 2e-5

Weight Decay: 0.01

Evaluation Strategy: After every epoch

Fine-tuning was done using the HuggingFace Trainer class to ensure efficient training and evaluation.

**4. Model Strengths and Advantages**

Maintains sentence fluency effectively.

Performs well in subtle grammar refinements.

Handles diverse sentence structures.

**5. Model Limitations**

Prone to overcorrection in ambiguous contexts.

Limited performance on highly complex grammar errors.

**6. Summary in Simple Terms**

The Coedit model tokenizes incorrect sentences and predicts corrected outputs while preserving sentence fluency. Its collaborative editing architecture ensures smooth and natural grammar corrections.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**3.4 BERT Classification Model**

**Explanation of the BERT Classification Model in the Code**

In this model, we implemented a grammar validation strategy using the BERT (Bidirectional Encoder Representations from Transformers) model. The BERT model acts as a classifier to determine whether a sentence output by the seq2seq models is grammatically correct or not.

**1. Input and Label Strategy**

The input consists of sentences from the seq2seq models, while labels indicate whether the sentence is correct (1) or incorrect (0).

Example Input: She goes to the school.

Label: 1 (Correct)

Example Input: She go to the school.

Label: 0 (Incorrect)

**2. Dataset Tokenization and Preprocessing**

Each sentence was tokenized using the BERT tokenizer. Tokenization steps included:

Padding: Sentences were padded to a maximum length of 64 tokens.

Truncation: Longer sentences were truncated.

Label Mapping: Sentences were paired with their corresponding binary labels.

**3. Fine-Tuning Strategy**

The BERT classification model was fine-tuned on the labeled dataset with hyperparameters optimized for accuracy:

Epochs: 5

Batch Size: 8

Learning Rate: 5e-5

Evaluation Strategy: After every epoch

Fine-tuning was performed using the HuggingFace Trainer API.

**4. Model Strengths and Advantages**

High accuracy in identifying valid corrections.

Effective handling of sentence-level context.

Robust classification performance.

**5. Model Limitations**

Misclassifications may occur in highly ambiguous sentences.

Performance depends on the diversity of training data.

**6. Summary in Simple Terms**

The BERT classification model evaluates the outputs from seq2seq models and assigns confidence scores based on grammatical correctness. It serves as the final validation layer in our grammar correction pipeline, ensuring the most reliable corrected sentence is selected.

**4. Ensemble Strategy**

**Explanation of the Ensemble Strategy in the Code**

We implemented a unique ensemble strategy to unify the outputs of three grammar correction models: GECToR, Coedit, and BART. Instead of following a fixed sequence for applying these models, we explored six different combinations of their order to determine the best correction for a given sentence. Each corrected sentence was evaluated using a BERT-based classification model, which scored each output for correctness. The final correction was selected based on the highest classification confidence score. This approach ensured a fair evaluation and improved overall accuracy by leveraging the strengths of each model. Here's a breakdown of what happens:

**1. Model Combination Strategy**

* The sentence is passed through GECToR, Coedit, and BART in six different orders:
  1. GECToR → Coedit → BART
  2. GECToR → BART → Coedit
  3. Coedit → BART → GECToR
  4. Coedit → GECToR → BART
  5. BART → GECToR → Coedit
  6. BART → Coedit → GECToR
* These combinations ensure that we explore all possible orderings, as each model might perform differently depending on its position in the sequence.

**2. Scoring the Outputs**

* Each of the six outputs generated by the different model orders is passed through a **BERT-based classification model**.
* The classification model evaluates the **correctness of each output** and assigns a score to indicate its confidence in the prediction.

A black background with white text

Description automatically generated

**3. Selecting the Best Output**

* After scoring all six outputs, we select the one with the **highest classification score**.
* This ensures that the final output is validated and represents the most reliable correction.

A black background with white text

Description automatically generated

**What We Achieved with This Approach**

1. **Diverse Evaluation:** By exploring multiple sequences of the models, we reduce bias and increase the chance of finding the most accurate correction.
2. **Objective Scoring:** The classification model acts as an independent judge, validating each output without bias toward any specific model.
3. **Dynamic Selection:** Instead of relying on a fixed model order, the system dynamically selects the most effective sequence for each sentence.
4. **Improved Accuracy:** This approach leverages the strengths of all three models and maximizes their collective performance.
5. **Results**

After thoroughly evaluating each model and their combined performance in the ensemble strategy, we achieved the following results:

T5 Model Accuracy: 85%

GEC Model Accuracy: 72%

Coedit Model Accuracy: 70%

BERT Classification Model F1 Score: 92%

BERT Classification Model Accuracy: 90%

Final Ensemble Accuracy: 96%

These results demonstrate the effectiveness of combining seq2seq grammar correction models with a validation layer using a classification model. The ensemble strategy successfully optimized the strengths of each individual model, resulting in improved accuracy and robust performance across various grammar correction tasks.

1. **Conclusion**

In this project, we successfully developed a grammar correction system by combining seq2seq models (T5, GEC, Coedit) with a classification layer using the BERT model. Each model contributed unique strengths, and their integration in an ensemble strategy allowed us to achieve a final accuracy of 96%. The T5 model excelled in generating fluent corrections, GEC effectively addressed explicit grammatical errors and Coedit ensured contextual coherence. The BERT classification model served as a robust validation layer, ensuring the reliability of the final output.

This approach not only demonstrated the potential of ensemble strategies in NLP tasks but also provided insights into model dependencies and performance limitations. Future improvements, such as domain-specific fine-tuning, enhanced dataset diversity, and multilingual support, could further elevate the system's capabilities. This project establishes a strong foundation for building scalable and reliable grammar correction tools suitable for real-world applications.