# Contextual Language Understanding with Transformer Models

## Testing

Testing is critical to validating the accuracy, robustness, and fairness of the transformer-based model.  
  
We start by evaluating the model on validation and test sets using a range of metrics. For classification tasks, we compute accuracy, precision, recall, and F1 score. For generative tasks, BLEU, ROUGE, and perplexity scores are used. Confusion matrices and precision-recall curves provide deeper insights into performance across classes.  
  
Cross-validation is used to ensure the model generalizes well to unseen data. Stratified sampling ensures balanced representation of labels in each fold. We perform statistical significance testing, such as t-tests or bootstrap analysis, to confirm improvements over baselines.  
  
Robustness testing involves feeding the model adversarial or noisy inputs to see how performance degrades. We test edge cases, long input sequences, and unusual sentence structures. Fairness testing evaluates whether the model performs equitably across different demographic groups, using disaggregated metrics.  
  
We conduct error analysis by examining misclassified examples. This helps identify systematic errors or ambiguous cases. Model interpretability tools like LIME, SHAP, or attention heatmaps help explain predictions.  
  
The testing phase ensures the model is not only accurate but also resilient, fair, and explainable. Findings from this phase feed back into development for iterative improvement.  
  
  
Testing is critical to validating the accuracy, robustness, and fairness of the transformer-based model.  
  
We start by evaluating the model on validation and test sets using a range of metrics. For classification tasks, we compute accuracy, precision, recall, and F1 score. For generative tasks, BLEU, ROUGE, and perplexity scores are used. Confusion matrices and precision-recall curves provide deeper insights into performance across classes.  
  
Cross-validation is used to ensure the model generalizes well to unseen data. Stratified sampling ensures balanced representation of labels in each fold. We perform statistical significance testing, such as t-tests or bootstrap analysis, to confirm improvements over baselines.  
  
Robustness testing involves feeding the model adversarial or noisy inputs to see how performance degrades. We test edge cases, long input sequences, and unusual sentence structures. Fairness testing evaluates whether the model performs equitably across different demographic groups, using disaggregated metrics.  
  
We conduct error analysis by examining misclassified examples. This helps identify systematic errors or ambiguous cases. Model interpretability tools like LIME, SHAP, or attention heatmaps help explain predictions.  
  
The testing phase ensures the model is not only accurate but also resilient, fair, and explainable. Findings from this phase feed back into development for iterative improvement.  
  
  
Testing is critical to validating the accuracy, robustness, and fairness of the transformer-based model.  
  
We start by evaluating the model on validation and test sets using a range of metrics. For classification tasks, we compute accuracy, precision, recall, and F1 score. For generative tasks, BLEU, ROUGE, and perplexity scores are used. Confusion matrices and precision-recall curves provide deeper insights into performance across classes.  
  
Cross-validation is used to ensure the model generalizes well to unseen data. Stratified sampling ensures balanced representation of labels in each fold. We perform statistical significance testing, such as t-tests or bootstrap analysis, to confirm improvements over baselines.  
  
Robustness testing involves feeding the model adversarial or noisy inputs to see how performance degrades. We test edge cases, long input sequences, and unusual sentence structures. Fairness testing evaluates whether the model performs equitably across different demographic groups, using disaggregated metrics.  
  
We conduct error analysis by examining misclassified examples. This helps identify systematic errors or ambiguous cases. Model interpretability tools like LIME, SHAP, or attention heatmaps help explain predictions.  
  
The testing phase ensures the model is not only accurate but also resilient, fair, and explainable. Findings from this phase feed back into development for iterative improvement.