CHAPTER 4

INITIAL FINDINGS

4.1 Introduction

This chapter introduced the initial results for BERT-based Semantic Similarity of Malaysia Legal Precedents study. It focuses on the presenting the comprehensive results from the development and evaluation of a BERT-based semantic similarity model for Malaysian legal texts. The objectives that had been discussed in the previous chapters are to investigate existing semantic similarity methods in legal NLP, develop and fine-tune a BERT-based model specifically for Malaysian legal contexts, and evaluate the model's performance through rigorous testing and visualization.

The analysis in this chapter following the complete pipeline from the data acquisition through model evaluation. It demonstrates that each phase of the methodology contributed to achieving a robust legal text similarity classification system. Therefore, the result shows the high effectiveness of transformer-based methods for understanding the semantic relationship in legal documents.

4.2 Data Sourcing and Acquisition Phase

4.2.1 Data Acquisition and Characteristics

This project used a Malaysian legal precedents dataset sourced from Kaggle, that included the comprehensive legal cases pairs for semantic similarity analysis:

Main characteristics of the Dataset:

- The dataset consists of Malaysian legal caselaw from various court of Malaysia.
- The dataset is pairs and have true label that annotated to give information about similarity between two cases.
- Each sentences have their own legal domain that help to train the model better.

This project used a Malaysian legal precedents dataset sourced from Kaggle, that included the comprehensive legal cases pairs for semantic similarity analysis:

- **Original dataset size**: 3,000 rows × 12 columns
- **Data source**: Kaggle legal text corpus
- **Format**: CSV file
- **Records successfully loaded**: 3,000 legal document pairs

Column Name	**Description & Distribution**
id	A unique identifier for each sentence pair (0 to 2999). Sequentially ordered, suggesting organized collection.
case1_id & case2_id	Unique case identifiers (e.g., ML-2024-2382). These follow a consistent alphanumeric format indicating state and year.
case1_text & case2_text	Legal sentence pairs, typically 237–430 characters in length. Sentences are descriptive and reflect real-world legal contexts.
case1_domain & case2_domain	Legal domains (e.g., criminal, property, family, contract). Help contextualize the source of each sentence.
true label	Ground truth for semantic similarity: 1 = similar, 0 = dissimilar.
is ambiguous	Boolean (True/False) indicating whether the similarity is ambiguous. Mostly False, used to mark edge cases.
complexity	Describes the difficulty level of the sentence pair (e.g., expert, intermediate, etc.).

Table 1 Analysis of each column in the dataset

Figure 1 below shows the distribution of the true labels which is 0 label is the higher with 1498 and 1 label is 1359 samples.

Figure 1 Distribution of True Labels

In Figure 2, the distribution of Case1 and Case2 domains shows that the number of each domain class is not far from each other, with the property law domain being the highest and the criminal domain being the lowest. This suggests a moderately uniform coverage of legal areas.

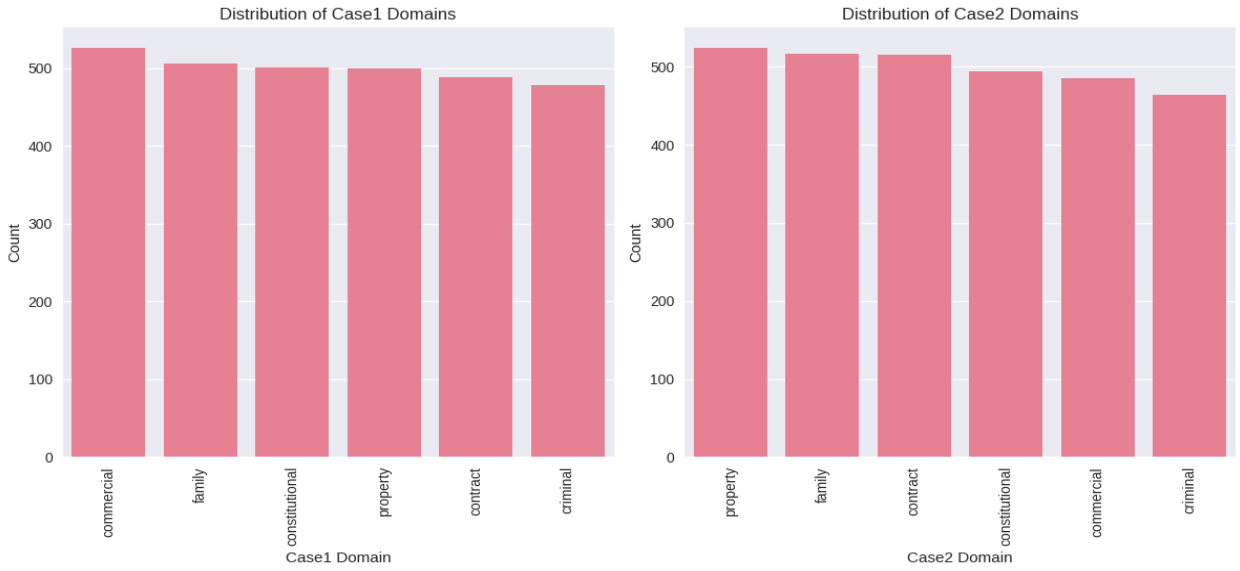


Figure 2 Distribution of Case1 and Case2 Domains

4.3 Data Cleaning and Preprocessing

Next, the process involved cleaning and preprocessing the dataset, where it went through a cleaning phase and the unnecessary columns were removed to gain better insights from the analysis

1. The legal text noise is cleaned while keeping useful punctuation, and filters out too-short or invalid entries.

```
def clean_text(text):
    """Clean legal text data"""
    if pd.isna(text) or text == '':
        return ""

    # Convert to string if not already
    text = str(text)

    # Remove extra whitespace
    text = re.sub(r'\s+', ' ', text).strip()

    # Keep legal punctuation and special characters
    # Only remove extreme special characters that might cause issues
    text = re.sub(r'[^\w\s\.,;:()\-\/&{}]', '', text)

    # Ensure minimum length
    if len(text) < 10:
        return ""

    return text
```

Figure 3 Cleaning process

2. The standardize domain function is used to clean and standardize legal domain names. First, the input is checked whether it missing or nan, and it return “unknown” if so. Then, it converts the input to lowercase and removes any extra spaces. The cleaned input is compared with the predefined dictionary of know legal domains. Therefore, it returns the standardize domain if it found.

```python
def standardize_domain(domain):
    """Standardize legal domain names"""
    if pd.isna(domain):
        return "unknown"

    domain = str(domain).lower().strip()

    # Domain mapping for consistency
    domain_mapping = {
        'commercial': 'commercial',
        'contract': 'contract',
        'property': 'property',
        'family': 'family',
        'criminal': 'criminal',
        'constitutional': 'constitutional',
        'tort': 'tort',
        'employment': 'employment',
        'intellectual_property': 'intellectual_property',
        'banking': 'banking_finance',
        'finance': 'banking_finance',
        'insurance': 'insurance',
        'tax': 'tax',
        'administrative': 'administrative',
        'civil': 'civil',
        'company': 'company_corporate',
        'corporate': 'company_corporate',
    }

    # Try to match with known domains
    for key, value in domain_mapping.items():
        if key in domain:
            return value

    return domain.replace(' ', '_')
```

Figure 3 Standardizing process

3. After the cleaning and standardizing the dataset, the duplicates data is removed. This is to ensure that the dataset is in better condition for better result when the training model started. Firstly, each pair is normalized by sorting the two values and removing duplicates based on the reverse sorted appearing, (e.g., (A, B) and (B, A)). This ensures only one unique representation of each case pair remains in the dataset.

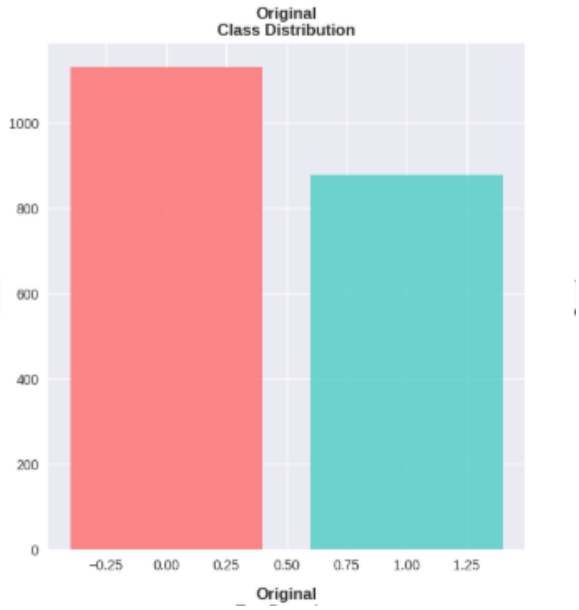
```
🔄 Removing semantic duplicates (reversed case pairs)...
  🔍 Found 229 rows that are part of duplicate pairs
  📄 Sample duplicate pairs:
    Pair 1 (ID: -6106279715210009412):
      Case1: Employment contract termination case where {employee} was dismissed without proper notice. The emplo...
      Case2: Sale and purchase agreement dispute regarding the {property_type} located at Kuala Lumpur. The purch...
    Pair 2 (ID: -1160084647589292402):
      Case1: Consumer protection claim under the Consumer Protection Act 1999. The consumer alleged {consumer_iss...
      Case2: Insurance claim dispute where the insurer denied liability for {claim_type}. The policy terms were i...
    Pair 3 (ID: 2491846763640360488):
      Case1: Religious law conflict with civil law regarding {religious_matter}. The court had to determine juris...
      Case2: Judicial review application against {authority} decision regarding {administrative_action}. The appl...
    Pair 4 (ID: -2562695766493108273):
      Case1: Federal-state jurisdiction dispute over {matter}. The case involved interpretation of the Ninth Sche...
      Case2: Federal-state jurisdiction dispute over {matter}. The case involved interpretation of the Ninth Sche...
  ✓ Removed semantic duplicates: 143 rows
  📊 Remaining rows: 2,857
```

Figure 4 Result for Removing Semantic Duplicates

4. After the above process, the process of balancing the data was begun. This process was done to ensure that the data is being proper balance for better result for the model training. Below are the graphs shows the imbalance class distribution before the balancing process. This will affect the result for the model training later. Therefore, this issue is addressed with proper balancing methods.

   Initial Class Distribution (Before Balancing):

   The dataset exhibited a moderate class imbalance requiring correction:



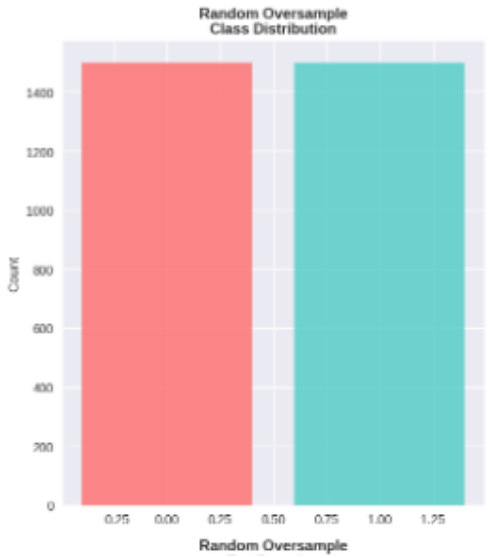
Class Label	Sample Count	Percentage	Description
Label 0 (Dissimilar)	1,498	52.43%	Non-similar legal precedent pairs
Label 1 (Similar)	1,359	42.57%	Similar legal precedent pairs
Total	**2,857**	**100.00%**	**Complete dataset**

Figure 5 Original Distribution of Class Label

These steps have 3 methods which are:

- random oversampling
- random under sampling
- combined approach

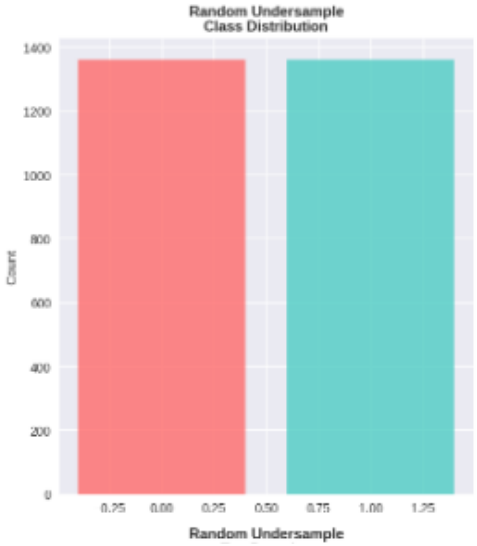
Firstly, the random oversampling process is done by balancing the class distribution by duplicating samples from the minority class. Therefore, it used "RandomOverSampler" to generate a dataset with equal class representation. Hence, increasing model exposure to underrepresented cases without losing any original data. Below is the result after using the random oversampling technique.



Class Label	Sample Count	Percentage	Description
Label 0 (Dissimilar)	1,498	50%	Non-similar legal precedent pairs
Label 1 (Similar)	1,498	50%	Similar legal precedent pairs
Total	**2,996**	**100.00%**	**Complete dataset**

Figure 6 Random Oversample Class Distribution

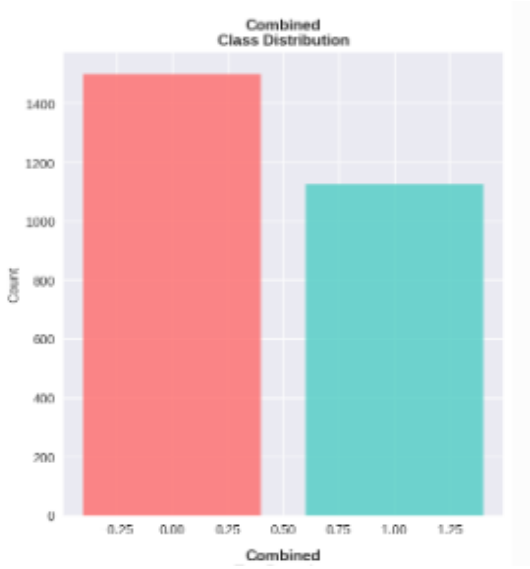
Besides, the random under sampling is another method that balancing the class distribution by duplicating samples from the majority class. Therefore, it used “RandomOverSampler” to generate a dataset with equal class representation. Therefore, the it reduced the dataset size while achieving class balance. This will result to speed up training but risks losing potentially valuable data. Below is the result after using the random under sampling technique.



Class Label	Sample Count	Percentage	Description
Label 0 (Dissimilar)	1,359	50%	Non-similar legal precedent pairs
Label 1 (Similar)	1,359	40%	Similar legal precedent pairs
Total	**2,718**	**100.00%**	**Complete dataset**

Figure 7 Random Under sample Class Distribution

The last methods are combined approach. This technique is technically combining the combined random under sampling of the majority class with oversampling of the minority class. For instances, the majority class was reduced to twice the size of the minority, while the minority was increased to 75% of the majority size. This hybrid method is used to support the useful majority examples while boosting minority representation. Below is the result after using the combined approach technique.



Class Label	Sample Count	Percentage	Description
Label 0 (Dissimilar)	1,480	50%	Non-similar legal precedent pairs
Label 1 (Similar)	1,211	40%	Similar legal precedent pairs
Total	**2,691**	**100.00%**	**Complete dataset**

Figure 8 Combined Approach of Class Distribution

4.3.1 Method Ranking

Therefore, the 3 methods were compared and evaluated using a custom quality score metric. Based on this, both Random Oversampling and Random Under sampling achieved the highest performance (1.000), while the Combined Approach followed with a slightly lower score (0.850). The score shows that the Random Oversampling and Random Under sampling achieved excellent score. Hence, it was decided that the **Random Oversampling** method was selected as the preferred balancing method due to its top performance (quality score: 1.000). Besides, it proven to enhance the minority class without discarding any original data. This preserves the full diversity of the dataset while addressing class imbalance effectively.

Selected Class Distribution (After Balancing using Random Oversampling):

Class Label	Sample Count	Percentage	Description
Label 0 (Dissimilar)	1,498	50%	Non-similar legal precedent pairs
Label 1 (Similar)	1,498	40%	Similar legal precedent pairs
Total	**2,996**	**100.00%**	**Complete dataset**

Table 2 Selected class distribution (Random Oversample)

4.3.2 EDA

Exploratory Detail Analysis is conducted after the data cleaning and processing done. This is to understand the dataset for more accurate used in model development. For instance, EDA is the process of analyzing and summarizing the dataset to uncover the main characteristics. Therefore, it will be represented with visualization for more meaningful insight. Furthermore, it helps in understanding the data structure, spotting patterns and detecting the anomalies. Therefore, the quality of overall dataset will be focused on before the application of machine learning.

Therefore, the dataset that has been cleaned and processed earlier will be analyzed using EDA. The analysis is useful for further actions, specifically for the model development phase. Below is the visualization of the EDA that has been conducted on the cleaned dataset and after the balancing process.

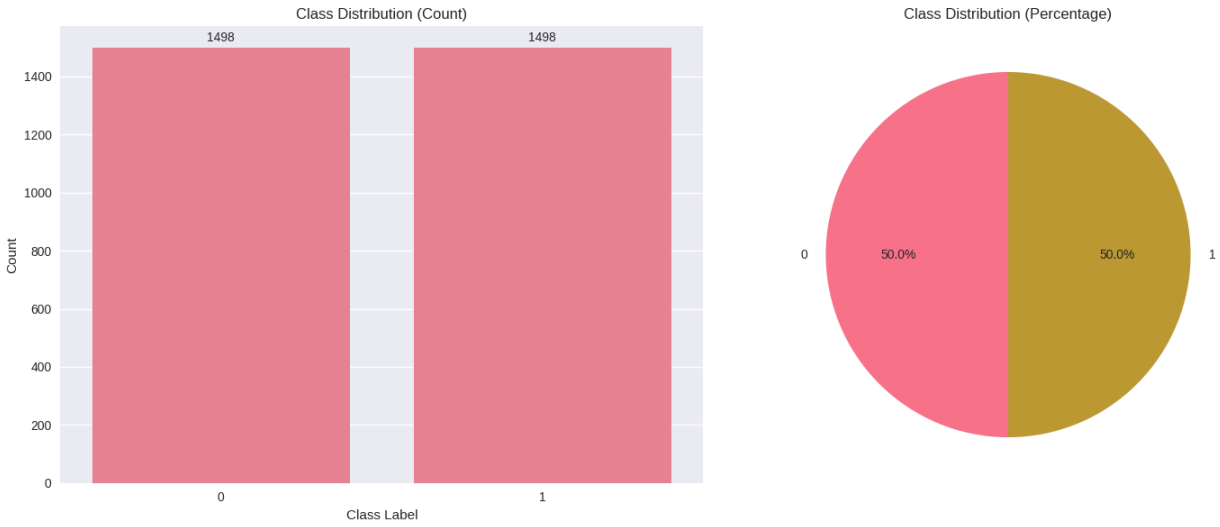


Figure 9 Finalized Class Distribution

The analysis of the class distribution count has been conducted, and the results show that the dataset is balanced, with 1,498 samples evenly distributed between the "similar" and "not similar" classes. Each class represents 50% of the data. This balance is important to avoid classification bias in machine learning models and to ensure more accurate results.

The figure 10 below shows the distribution of text lengths after the balancing process. It illustrates the distribution of text lengths for Case 1 and Case 2 after applying random oversampling. For

instance, the lengths of both texts are relatively similar, with a peak around 270–280 characters. Additionally, the maximum length exceeds 400 characters, but only for a few samples.

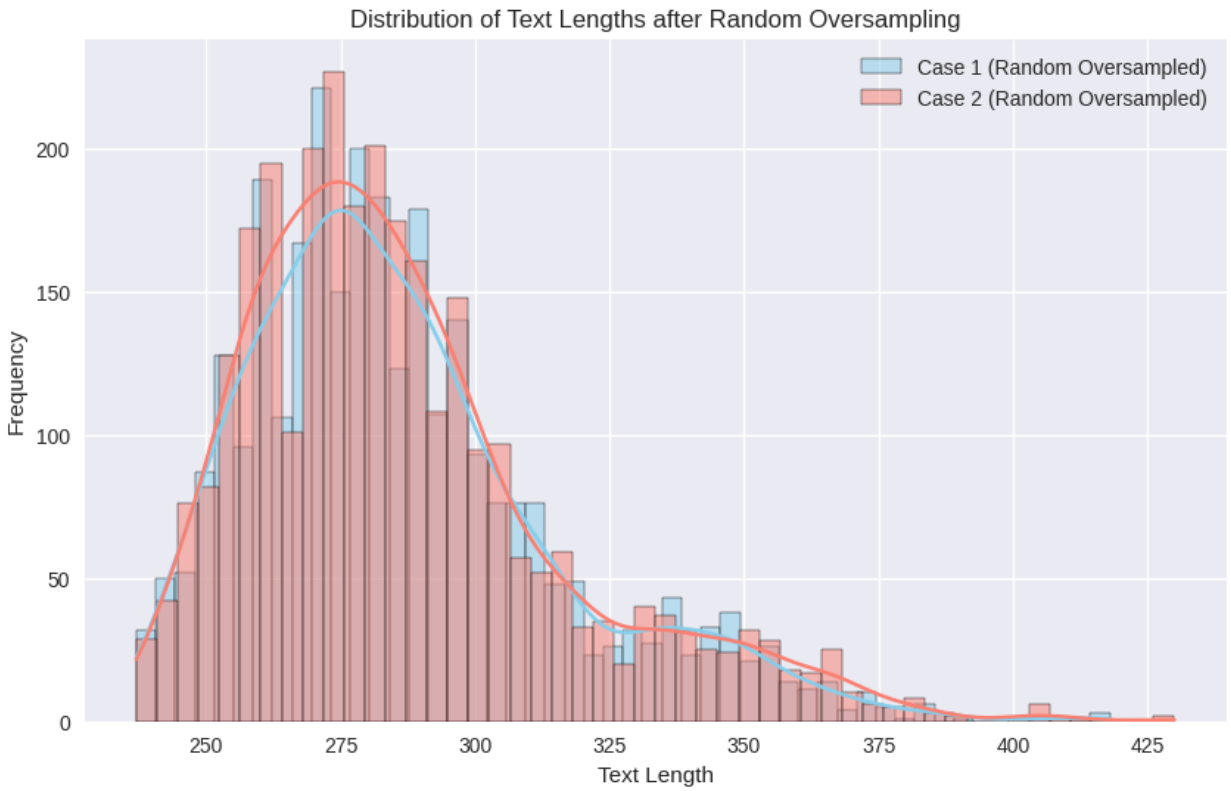


Figure 10 Distribution of Text Lengths After Random Oversampling

4.4 Model Development

The finalized dataset of 2,996 entries, which had been cleaned and preprocessed, was first loaded to be split into training, validation, and test sets. The proportion of each set were 2396 (80%) for training, and 300 (10%) each for validation and testing. Therefore, it will then be wrapped inside “InputExample” objects with the corresponding similarity labels 0 or 1. The DataLoader was set with a batch size of 16 that will be used during training and evaluation.

Next, the selected model which is a pre-trained Sentence-BERT model was fine-tuned using “CosineSimilarityLoss”. This tools is suitable for the semantic similarity tasks. Then the “EmbeddingSimilarityEvaluator” was employed on the validation set. This is to ensure the performance of the model being critically evaluated and the best performing model was saved based on the validation scores.

Moreover, the hyperparameter tuning was set to the model with three epochs. This epoch was evaluated with intermediate evaluations and it performed every 500 steps. Therefore, the best checkpoints which is the highest score will be saved as the final model.

Furthermore, the “EmbeddingSimilarityEvaluator” also conducted the evaluation on the test. For instances, the best model showed a Pearson correlation score of 0.6648 and Spearman correlation of 0.53304. The final model scored 0.6641 for Pearson correlation and 0.5362 for Spearman correlation. Therefore, the final model was selected based on the Pearson correlation as it is the primary metric for the semantic similarity tasks. Hence, the final model with best-performing checkpoint saved as Bert-sbert-legal that will be use for the evaluation’s visualization.

Hence, it concluded that the model demonstrated a good level of semantic correlation with a Pearson score of 0.665. It shows that the model is able to effectively capturing the semantic similarity between the legal sentence pairs. This making it suitable for the legal NLP tasks and application.

4.5 Model Evaluation and Visualization

After the training, the final evaluation was conducted on a test set that had been split earlier. The test set consist of 3oo sentence pairs and was processed using the fine-tuned model. Therefore, the predictions were obtained by calculating the cosine similarity between embedded representations of each sentence pair. For instances, the predicted score was ranged from -.0.187 to 0.996. This range will be compared with the true label range from 0 to 1.

1. Regression Metrics

To assess the model's capability in capturing semantic similarity as a continuous measure, several regression metrics were computed:

- Pearson Correlation: 0.6641
- Spearman Correlation: 0.5362
- R^2 Score: 0.4278
- Root Mean Squared Error (RMSE): 0.3782
- Mean Absolute Error (MAE): 0.2670

Above result shows that the model demonstrates a moderate to good correlation. For instances, the model prediction was a reasonably low error, which means it is reliable for legal NLP tasks. Besides, the

Pearson correlation of 0.6641 suggests that the model is able to know and capture the semantically similar legal sentences.

2. Classification Metrics

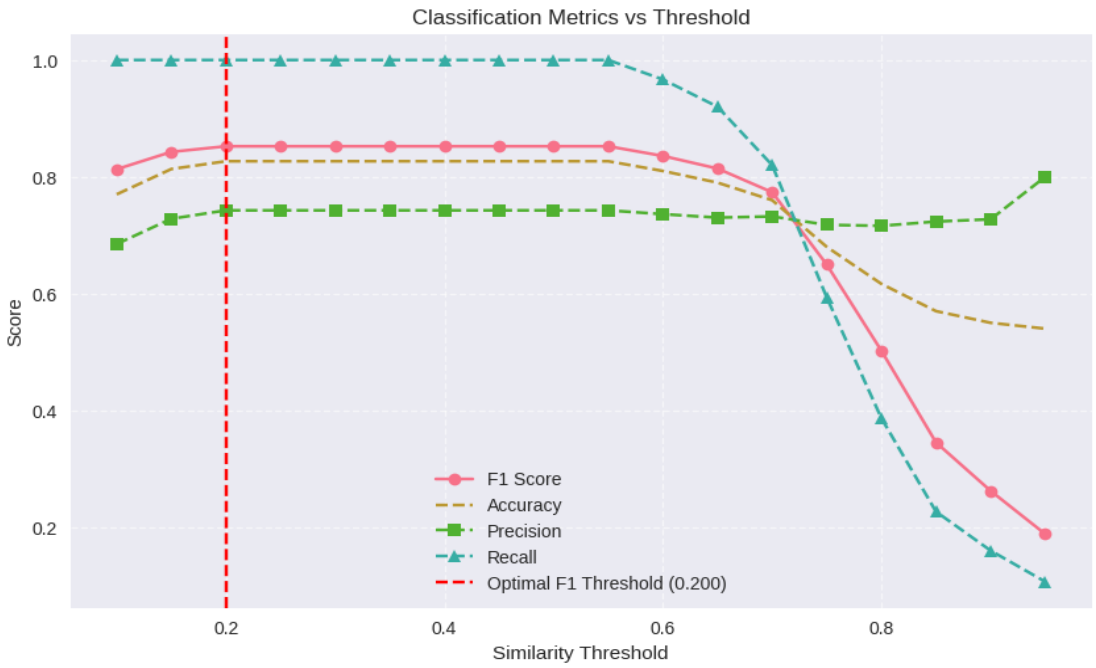
Then, it was evaluated as a binary classification problem. For instance, the sentence pairs such as similar or dissimilar were classified based on the thresholds. Below are the three thresholds that were tested:

Threshold	Accuracy	Precision	Recall	F1 Score
0.5	0.8267	0.7426	1.0000	0.8523
0.6	0.8100	0.7360	0.9667	0.8357
0.7	0.7600	0.7321	0.8200	0.7736

The result shows that the 0.5 thresholds showed the highest F1 score of 0.8523. This provides that the balance trade-off between the precision and recall. Besides, the recall of 1.0000 indicated that the model successfully identified all actual similar pairs.

3. Optimal Threshold Selection

Therefore, to obtain the optimal thresholds that effectively classified the pairs, a fine-grained search from 0.1 to 0.95 was conducted. This involved in steps of 0.05 to get the best result. Hence, it was found that the optimal thresholds were 0.200 which yielded the highest F1 score of **0.8523**.



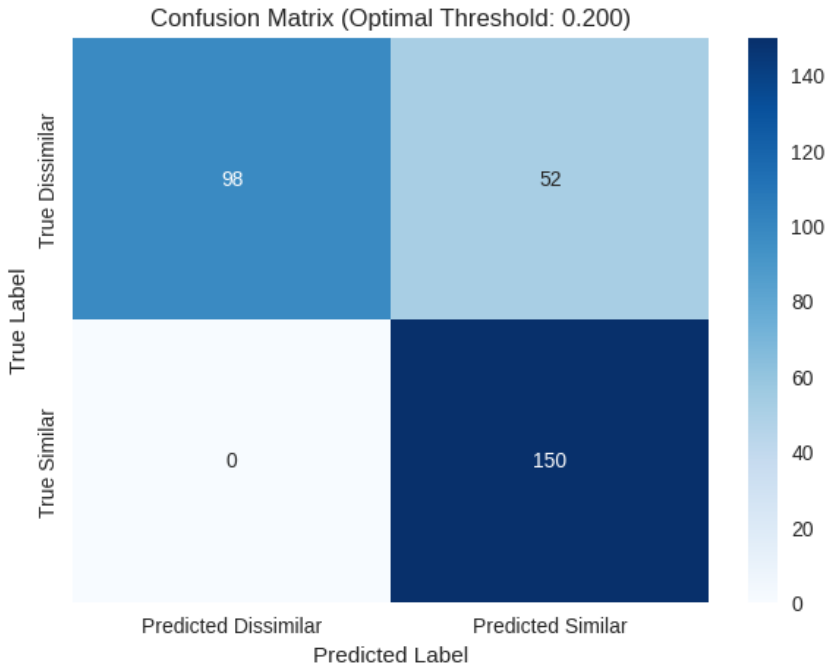
Class	Precision	Recall	F1-Score	Support
Dissimilar	1.00	0.65	0.79	150
Similar	0.74	1.00	0.85	150
Accuracy			0.83	300
Macro Avg	0.87	0.83	0.82	300
Weighted Avg	0.87	0.83	0.82	300

Figure 11 Classification Metrics vs Threshold

It shows that the performance confirms the model is well-tuned to favor recall. Moreover, it still maintains the high precision and making it highly suitable for Legal NLP tasks. This is because it requires a high recognition of true semantic similarity to avoid significant implications.

4. Confusion Matrix

Figure 12 Confusion Matrix illustrates the model's binary classification performance at the optimal threshold of 0.200. Therefore, it shows that the number of the correctly and incorrectly classified sentence pairs as ether similar or dissimilar:



	Predicted Similar	**Predicted Dissimilar**
True Similar	150 (TP)	0 (FN)
True Dissimilar	52 (FP)	98 (TN)

Figure 12 Confusion Matrix

4.6 Conclusion

The sentence-BERT model that was fine-tuned with Malaysian legal case pairs shows a strong performance in identifying semantic similarity. This can prove by the Pearson correlation of 0.6641 and the F1 score of 0.8523 at the optimal classification threshold. Next chapter will be the future work and conclusion.