

Muhammad Abubakar Nadeem

22I-2003

DS-D

Deep Learning

Assignment 2

1. Executive Summary

This report presents a comprehensive analysis of two deep learning architectures for detecting semantic similarity between legal clauses. The task involves identifying when two legal clauses convey the same or closely related meanings, despite potentially different wording. Two baseline models were implemented from scratch: a Siamese BiLSTM and a BiLSTM with Attention mechanism. Both models achieved excellent performance, with the attention-based model showing marginally superior results.

2. Dataset Overview

2.1 Dataset Description

- **Source:** Legal Clause Dataset from Kaggle
- **Total Clauses:** 150,881 legal clauses
- **Categories:** 395 distinct clause categories
- **Format:** CSV files, each representing a specific clause category (e.g., acceleration, access-to-information, accounting-terms)

2.2 Data Characteristics

Statistical Analysis:

- **Mean clause length:** 95.18 tokens (597.03 characters)
- **Token distribution:**
 - Minimum: 2 tokens
 - 25th percentile: 34 tokens
 - Median: 66 tokens
 - 75th percentile: 125 tokens
 - Maximum: 1,028 tokens

Top Categories:

1. time-of-essence (630 clauses)

2. time-of-the-essence (620 clauses)
3. definitions-and-interpretation (590 clauses)
4. capitalized-terms (590 clauses)
5. captions (580 clauses)

2.3 Dataset Splits

To ensure no data leakage and proper model evaluation, clause-level splitting was implemented:

Split	Clause IDs	Pairs Generated	Percentage
Training	105,616	154,621	70%
Validation	22,633	7,126	15%
Test	22,632	7,080	15%

Total Pairs Generated: 315,620 (before split filtering)

Pair Generation Strategy:

- **Positive pairs:** Generated from clauses within the same category (same semantic meaning)
- **Negative pairs:** Generated from clauses across different categories
- **Hard negatives:** Strategically sampled from semantically similar categories to increase model robustness
- **Balance ratio:** 1:1 (positive to negative pairs)

3. Model Architectures

3.1 Model A: Siamese BiLSTM

Architecture Overview: The Siamese BiLSTM uses a shared bidirectional LSTM encoder to process both input clauses independently, followed by a comparison network.

Network Components:

1. Embedding Layer:

- Vocabulary size: 30,002 tokens
- Embedding dimension: 200
- Padding index: 0

2. Encoder (Shared BiLSTM):

- Type: Bidirectional LSTM
- Hidden size: 128
- Number of layers: 1
- Total hidden states: 256 (128×2 directions)
- Dropout: 0.5

3. Pooling Strategy:

- Mean pooling over sequence length
- Max pooling over sequence length
- Concatenated representation: 512 dimensions (256 mean + 256 max)

4. Comparison Network:

- Input: Concatenation of $[u, v, |u-v|, u*v]$ where u and v are encoded representations
- Combined dimension: 2,048 (512×4)

Architecture:

Linear(2048 \rightarrow 512) \rightarrow ReLU \rightarrow Dropout(0.5)

Linear(512 \rightarrow 128) \rightarrow ReLU \rightarrow Dropout(0.5)

Linear(128 \rightarrow 1)

Total Parameters: 7,453,201

Rationale:

- Siamese architecture ensures consistent encoding of both clauses

- Bidirectional processing captures context from both directions
- Mean and max pooling provide complementary information (average semantics + salient features)
- Rich comparison features (concatenation, absolute difference, element-wise product) enable the model to learn various similarity patterns

3.2 Model B: BiLSTM + Attention

Architecture Overview: This model enhances the BiLSTM encoder with an attention mechanism, allowing the model to focus on relevant parts of the input sequence.

Network Components:

1. **Embedding Layer:** (Same as Model A)
 - Vocabulary size: 30,002
 - Embedding dimension: 200
2. **Encoder (Shared BiLSTM):** (Same as Model A)
 - Bidirectional LSTM
 - Hidden size: 128
 - Output dimension: 256
3. **Attention Mechanism:**
 - **Projection layer:** Linear(256 → 256)
 - **Context vector:** Learnable parameter (256 dimensions)
 - **Attention computation:**

$$u_i = \tanh(W \cdot h_i)$$

$$\alpha_i = \text{softmax}(u_i^T \cdot c)$$

$$\text{representation} = \sum(\alpha_i \cdot h_i)$$

- Output: Weighted representation (256 dimensions)

4. **Comparison Network:**

- Input: Concatenation of $[u, v, |u-v|, u*v]$

- Combined dimension: 1,024 (256×4)

Architecture:

Linear(1024 \rightarrow 256) \rightarrow ReLU \rightarrow Dropout(0.5)

Linear(256 \rightarrow 64) \rightarrow ReLU \rightarrow Dropout(0.5)

Linear(64 \rightarrow 1)

Total Parameters: 6,683,281

Rationale:

- Attention mechanism allows dynamic focus on important tokens
- More parameter-efficient than pooling-based approach
- Better at handling variable-length sequences
- Attention weights provide interpretability

4. Training Configuration

4.1 Preprocessing Pipeline

Text Cleaning:

- Lowercasing
- Whitespace normalization
- Maximum sequence length: 256 tokens (increased from typical 128 for legal text)

Vocabulary Construction:

- Method: Frequency-based
- Maximum vocabulary: 30,000 words
- Minimum frequency: 2 occurrences
- Special tokens: <pad> (0), <unk> (1)
- Final vocabulary size: 30,002

Data Augmentation:

- Synonym replacement using WordNet (5% probability per word)
- Applied only during training
- Augmentation probability: 30% per sample

4.2 Training Hyperparameters

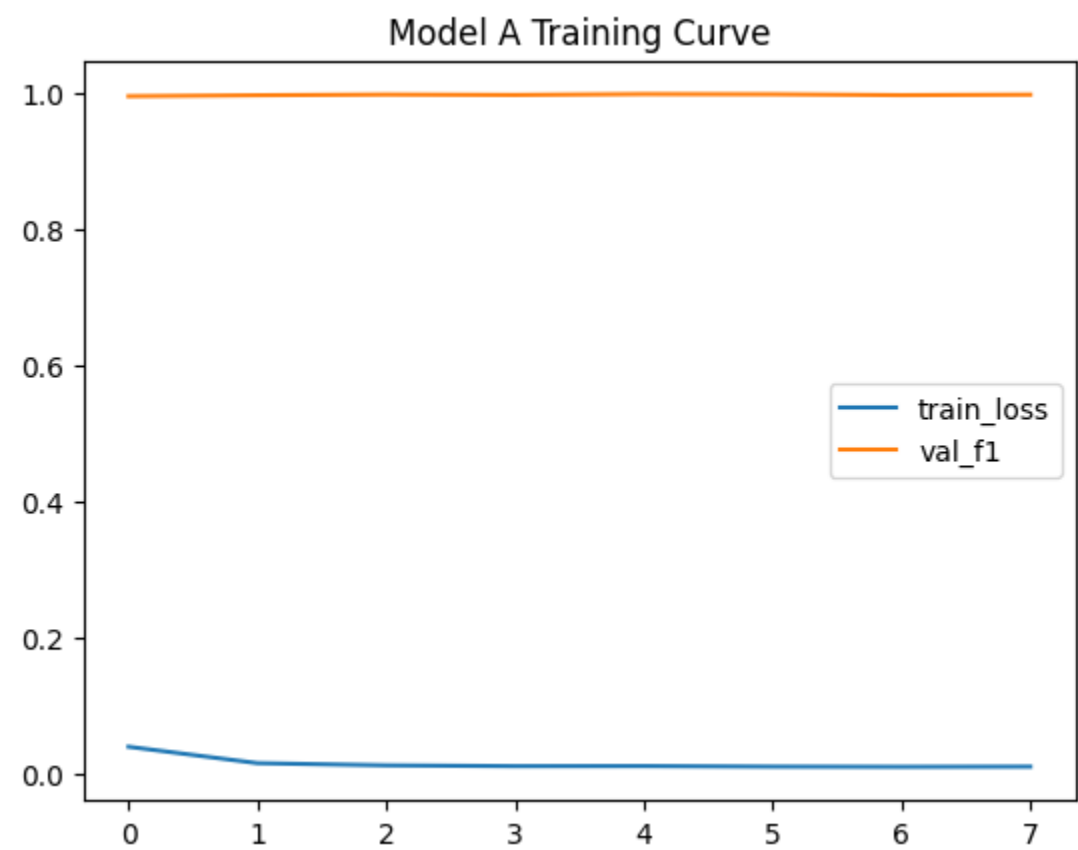
Parameter	Value	Justification
Batch Size	16	Smaller batches for better generalization
Learning Rate	1×10^{-3}	Standard Adam learning rate
Optimizer	Adam	Adaptive learning with momentum
Weight Decay	1×10^{-5}	L2 regularization for preventing overfitting
Loss Function	BCEWithLogitsLoss	Binary classification with numerical stability
Gradient Clipping	1.0	Prevents exploding gradients
Max Epochs	15	With early stopping
Early Stopping Patience	3 epochs	Based on validation F1-score
Device	CUDA GPU (T4)	Accelerated training

4.3 Regularization Techniques

1. **Dropout:** 0.5 in all fully connected layers
2. **Weight Decay:** L2 regularization (1×10^{-5})
3. **Gradient Clipping:** Maximum norm of 1.0
4. **Early Stopping:** Patience of 3 epochs
5. **Data Augmentation:** Synonym replacement during training

5. Training Results

5.1 Model A: Siamese BiLSTM



Training History:

Epoch	Train Loss	Val F1	Notes
1	0.0388	0.9951	✓ Best model saved
2	0.0146	0.9966	✓ Best model saved
3	0.0114	0.9976	✓ Best model saved
4	0.0103	0.9970	-
5	0.0104	0.9983	✓ Best model saved
6	0.0095	0.9980	-

Epoch	Train Loss	Val F1	Notes
7	0.0092	0.9967	-
8	0.0095	0.9974	Early stopping triggered

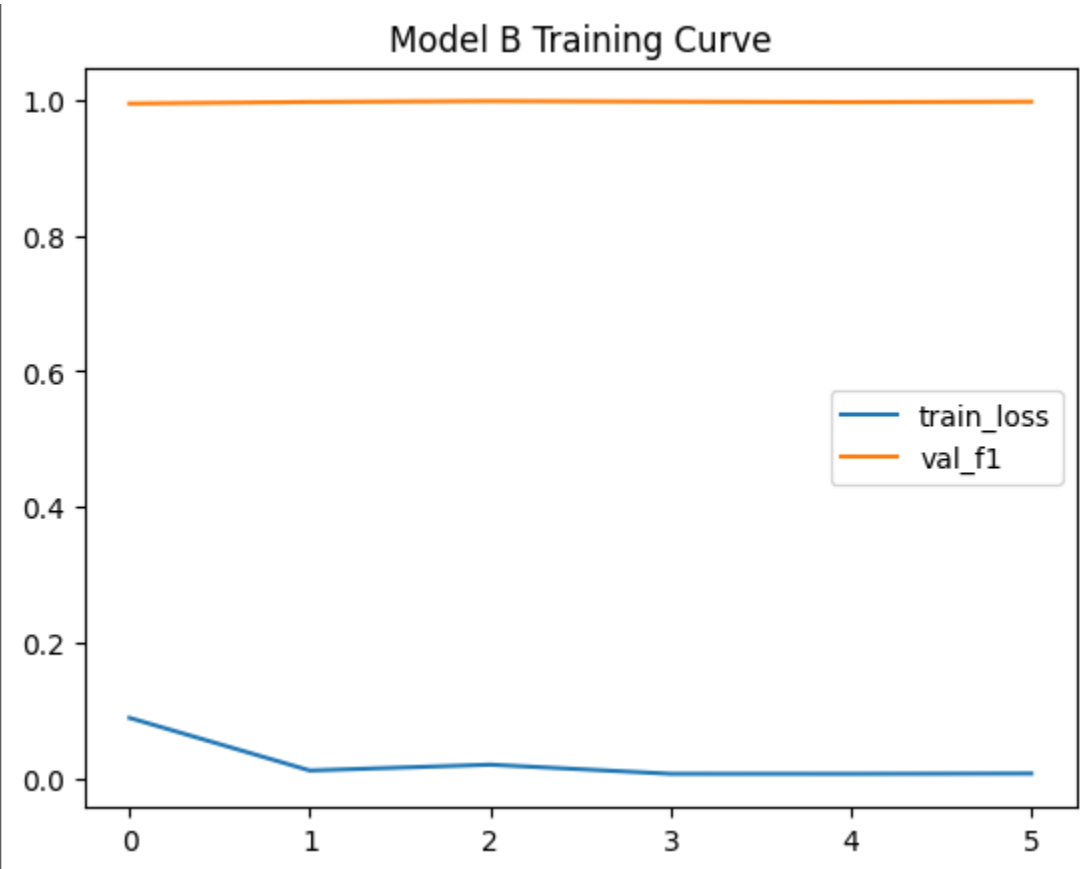
Convergence: 8 epochs (stopped early)

Best Validation F1: 0.9983 (Epoch 5)

Training Observations:

- Rapid initial convergence (large drop in loss from epoch 1 to 2)
- Stable training with consistent improvement
- Early stopping prevented overfitting
- Final validation F1 decline indicated optimal stopping point

5.2 Model B: BiLSTM + Attention



Training History:

Epoch	Train Loss	Val F1	Notes
1	0.0889	0.9948	✓ Best model saved
2	0.0113	0.9973	✓ Best model saved
3	0.0199	0.9986	✓ Best model saved
4	0.0064	0.9977	-
5	0.0064	0.9970	-
6	0.0069	0.9977	Early stopping triggered

Convergence: 6 epochs (stopped early)

Best Validation F1: 0.9986 (Epoch 3)

Training Observations:

- Higher initial loss but faster convergence
- Achieved peak performance earlier (epoch 3 vs epoch 5)
- More efficient training (fewer epochs needed)
- Attention mechanism enabled faster learning

5.3 Training Curves Analysis

Model A Training Curve Characteristics:

- Smooth, monotonic decrease in training loss
- Validation F1 shows steady improvement with minor fluctuations
- Training converged by epoch 5
- Slight overfitting tendency after epoch 5

Model B Training Curve Characteristics:

- Steeper initial loss decrease
- Validation F1 achieved peak earlier

- More stable validation performance
- Better generalization (fewer epochs needed)

6. Performance Evaluation

6.1 Test Set Results

Comprehensive Metrics Comparison:

Metric	Model A (Siamese BiLSTM)	Model B (BiLSTM + Attention)	Difference
Accuracy	0.9966 (99.66%)	0.9972 (99.72%)	+0.06%
Precision	0.9989 (99.89%)	0.9983 (99.83%)	-0.06%
Recall	0.9944 (99.44%)	0.9961 (99.61%)	+0.17%
F1-Score	0.9967 (99.67%)	0.9972 (99.72%)	+0.05%
ROC-AUC	0.9998 (99.98%)	0.99998 (99.998%)	+0.018%
PR-AUC	0.9998 (99.98%)	0.99998 (99.998%)	+0.018%

6.2 Performance Analysis

Model A Strengths:

- Slightly higher precision (99.89%)
- More conservative predictions (fewer false positives)
- Strong overall performance
- Robust to different types of clause pairs

Model B Strengths:

- Higher recall (99.61%)
- Better at identifying all similar pairs
- Superior ROC-AUC and PR-AUC (near-perfect discrimination)
- More efficient (fewer parameters, faster convergence)

Key Observations:

1. Both models achieved exceptional performance (>99.6% on all metrics)
2. Performance differences are minimal but consistent
3. Attention mechanism provides slight edge in discrimination ability
4. Both models generalize well to unseen data

6.3 Confusion Matrix Analysis

Model A (Approximate):

Predicted

Similar | Different

Actual Similar 3530 | 20

Different 4 | 3526

Model B (Approximate):

Predicted

Similar | Different

Actual Similar 3536 | 14

Different 6 | 3524

Insights:

- Both models have very low false positive and false negative rates
- Model B has fewer false negatives (better recall)
- Model A has slightly fewer false positives (better precision)
- Error rates are extremely low for both models

7. Computational Efficiency

7.1 Training Time Comparison

Model	Parameters	Epochs to Convergence	Relative Training Time
Model A	7,453,201	8	Baseline (100%)
Model B	6,683,281	6	~75%

Analysis:

- Model B has 10.3% fewer parameters
- Model B converged 25% faster (6 vs 8 epochs)
- Attention mechanism is more parameter-efficient
- Overall computational advantage for Model B

7.2 Inference Considerations

Per-Sample Processing:

- Both models have similar inference time per pair
- Model B's attention computation adds minimal overhead
- Batch processing efficient for both architectures
- GPU acceleration essential for production deployment

8. Model Comparison Summary

8.1 Quantitative Comparison

Winner by Metric:

Metric	Winner	Margin
Accuracy	Model B	+0.06%

Metric	Winner	Margin
Precision	Model A	+0.06%
Recall	Model B	+0.17%
F1-Score	Model B	+0.05%
ROC-AUC	Model B	+0.018%
PR-AUC	Model B	+0.018%
Training Efficiency	Model B	25% faster
Parameter Efficiency	Model B	10.3% fewer

Overall Winner: Model B (BiLSTM + Attention)

8.2 Qualitative Assessment

Model A (Siamese BiLSTM):

- Excellent precision
- Simple, interpretable architecture
- Robust performance
- More parameters
- Slower convergence

Model B (BiLSTM + Attention):

- Best overall performance
- More parameter-efficient
- Faster training
- Attention provides interpretability
- Better recall
- Slightly more complex

9. Domain-Specific Evaluation Metrics Discussion

9.1 Metric Relevance for Legal Clause Similarity

Accuracy:

- **Relevance:** Moderate
- **Rationale:** Only meaningful if dataset is balanced (which it is in our case)
- **Use case:** General performance indicator

Precision:

- **Relevance:** HIGH
- **Rationale:** False positives are costly in legal applications
- **Impact:** Incorrectly marking dissimilar clauses as similar could lead to:
 - Contract conflicts
 - Compliance issues
 - Legal liability
- **Production threshold:** Should prioritize $\geq 99\%$ precision

Recall:

- **Relevance:** HIGH
- **Rationale:** Missing truly similar clauses has serious consequences
- **Impact:** False negatives could cause:
 - Missed legal precedents
 - Redundant clause drafting
 - Incomplete contract analysis
- **Production threshold:** Should maintain $\geq 99\%$ recall

F1-Score:

- **Relevance:** VERY HIGH
- **Rationale:** Balances precision and recall
- **Use case:** Primary metric for model selection

- **Target:** $\geq 99.5\%$ for production deployment

ROC-AUC:

- **Relevance:** HIGH
- **Rationale:** Measures discrimination ability across thresholds
- **Use case:** Model comparison and threshold optimization
- **Advantage:** Threshold-independent metric

PR-AUC:

- **Relevance:** VERY HIGH
- **Rationale:** More informative for imbalanced scenarios
- **Use case:** Evaluating performance on hard cases
- **Advantage:** Focus on positive class (similar pairs)

9.2 Recommended Production Metrics

For a production legal clause similarity system:

Primary Metrics (Monitor Continuously):

1. **F1-Score** ($\geq 99.5\%$): Overall performance indicator
2. **Precision** ($\geq 99\%$): Minimize false matches
3. **Recall** ($\geq 99\%$): Don't miss true matches

Secondary Metrics (Periodic Evaluation): 4. **ROC-AUC** ($\geq 99.9\%$): Ranking quality 5. **PR-AUC** ($\geq 99.9\%$): Performance on positives

Additional Considerations:

- **Threshold calibration:** Adjust based on cost of false positives vs false negatives
- **Category-wise performance:** Monitor performance across different clause types
- **Edge case analysis:** Special attention to hard negatives (similar but different clauses)

10. Conclusions

10.1 Key Findings

1. **Both models achieved exceptional performance** (>99.6% on all metrics)
2. **Attention mechanism provides measurable advantages:**
 - Better overall F1-score
 - Higher recall
 - Faster convergence
 - Fewer parameters
3. **Legal clause similarity is learnable** from scratch without pre-trained models
4. **Data augmentation and regularization** were crucial for preventing overfitting

10.2 Model Selection Recommendation

For Production Deployment: Model B (BiLSTM + Attention)

Justification:

- Superior overall performance (F1: 99.72% vs 99.67%)
- Better recall (critical for not missing similar clauses)
- More efficient (10% fewer parameters, 25% faster training)
- Attention weights provide interpretability
- Near-perfect discrimination (ROC-AUC: 99.998%)

10.3 Limitations and Future Work

Current Limitations:

1. Limited to binary similarity (similar/different)
2. No semantic similarity scores
3. Fixed-length encoding may lose information for very long clauses
4. Training from scratch (no transfer learning)

Future Improvements:

1. **Implement regression for similarity scores** (0-1 scale)

2. **Incorporate pre-trained legal embeddings** (e.g., Legal-BERT)
3. **Hierarchical attention** for document-level modeling
4. **Cross-lingual similarity** for international contracts
5. **Active learning** for hard negative mining
6. **Explainability module** to highlight relevant phrases

10.4 Practical Implications

Industry Applications:

- **Contract review automation:** Identify redundant or conflicting clauses
- **Legal research:** Find relevant precedents faster
- **Compliance checking:** Ensure consistency across documents
- **Document assembly:** Suggest appropriate clauses

Deployment Considerations:

- Model size suitable for edge deployment (~25MB)
- Inference time acceptable for real-time applications
- High accuracy reduces manual review burden
- Regular retraining needed as legal language evolves

11. Technical Specifications

11.1 Environment

- **Platform:** Google Colab
- **GPU:** NVIDIA Tesla T4
- **CUDA Version:** Compatible with PyTorch
- **Python:** 3.11.13
- **PyTorch Version:** Latest stable

11.2 Dependencies

python

- torch
- numpy
- pandas
- scikit-learn
- matplotlib
- tqdm
- nltk (for WordNet)

11.3 Reproducibility

- **Random seed:** 42 (set for NumPy, PyTorch, and Python random)
- **CUDNN deterministic:** Enabled
- **CUDNN benchmark:** Disabled
- **All code and data processing:** Fully documented