LID VAN ASSOCIATIE KU LEUVEN

THOMAS MORE

DOCUMENTATION

Documentation Of Experiments Done To Achieve Final Model

Bachelor in Applied Computer Science.

Academiejaar 2024-2025

Campus: Geel

Emmanuel Akpandara

Introduction

The rapid growth of battery-related innovations has led to an influx of patent filings, making efficient classification of these documents crucial for research and development. **Umicore Belgium**, a leader in sustainable materials technology, requires an automated solution to manage and analyze patent data effectively. This project focuses on enhancing patent classification by transitioning from a traditional **Bag of Words (BoW)** approach to advanced **Large Language Models (LLMs)** like **PatentBERT and BatteryBert**, which better capture the semantic meaning of patent and battery related texts.

Conducted during my internship from **September 2024 to December 2024**, this research aims to address limitations in the current system by improving accuracy, in processing complex, domain-specific patent language. This work is essential to support strategic decision-making and innovation tracking within Umicore's battery materials division.

These experiments showcases a highlight of the gradual processes taken to refine the finetuning of the Bert model, each of these experiments were logged in **MLFlow** for analysis which introduced insights on steps to take going forward, furthermore, it aids the scientists and researchers in reproducing similar experiments with an end result in mind.

General:

Dataset Details: High Nickel Data

- **Total Training Samples:** 3885
- **Class Distribution:**
    - Class 0 (Potentially Not Relevant): 2881 samples
    - Class 1 (Potentially Relevant): 1004 samples
- **Random Weighted Sampling Utilized**

Doc HN PT 01

Basic Training Loop:

- **Model**: PatentBERT
- **Loss Function:** Cross-Entropy Loss
- **Optimizer**: AdamW
- **Learning Rate:** 0.001
- **Early Stopping:** Triggered on validation loss
- **Layers Trained:** Only the classifier layer
- **Batch Size:** 8
- **Epochs Completed:** 4

Results:

- Test Accuracy: 62%
- Test F1-Score: 64%
- Test Precision: 82%
- Test Recall: 62%
- PR Curve AUC: 55%

Analysis:

- **From the confusion matrix:**
    - 97% of relevant patents (Class 1) were correctly predicted (True Positives).
    - 50% of irrelevant patents (Class 0) were correctly predicted (True Negatives).
    - This indicates the model is biased towards predicting Class 1, making it more effective at identifying relevant patents.

Challenges Identified:

- Significant class imbalance in training data (Class 0 dominates).
- Training only the classifier layer limited the model's ability to generalize.
- Low PR curve AUC indicates difficulty maintaining high precision across different recall levels.

Future Steps:

1. Fine-Tune Full Model: Gradually unfreeze BERT layers to incorporate semantic embeddings.
2. Try Other Architectures: Experiment with models like BatteryBert**.**
3. Optimize Hyperparameters: Adjust batch size, learning rate

Doc HN PT 02

Basic Training Loop:

- **Model**: PatentBERT
- **Loss Function:** Cross-Entropy Loss
- **Optimizer**: AdamW
- **Learning Rate:** 0.001

- **Early Stopping:** Triggered on validation loss
- **Layers Trained:** Classifier and Pooler Layer
- **Batch Size:** 8
- **Epochs Completed:** 8

Results:

- Test Accuracy: 75%
- Test F1-Score: 76%
- Test Precision: 83%
- Test Recall: 75%
- PR Curve AUC: 63%

Analysis:

- **Confusion Matrix Observations:**
    - 90% of relevant patents (Class 1) were correctly predicted (True Positives).
    - 69% of irrelevant patents (Class 0) were correctly predicted (True Negatives).
- **Performance Improvements:**
    - Compared to Doc HN PT 01, the model's ability to correctly identify irrelevant patents (Class 0) improved by 19%, demonstrating better handling of class imbalance.
    - The PR Curve AUC increased to 63%, showing improved precision across a wider range of recall values, which indicates more balanced performance.
- **Class Bias:**
    - Although the model performs better overall, it still exhibits a slight bias toward predicting Class 1, making it more effective at identifying relevant patents than irrelevant ones.

Challenges Identified:

- Significant class imbalance in training data (Class 0 dominates).
- Training only the classifier layer limited the model's ability to generalize.

- Low PR curve AUC indicates difficulty maintaining high precision across different recall levels.

Key Differences from Doc HN PT 01:

1. **Layers Trained**: Training both the **Classifier Layer** and the **Pooler Layer** allowed the model to utilize richer semantic embeddings from the final BERT layers, resulting in a significant performance boost.
2. **Increase in Class 0 Recall**: The model demonstrated a notable improvement in its ability to identify irrelevant patents, balancing the precision-recall trade-off better than in Doc HN PT 01.
3. **Higher Metrics Across the Board**:
    - Accuracy increased by 13% (from 62% to 75%).
    - F1-Score improved by 12% (from 64% to 76%).
    - Precision for Class 1 increased slightly from 82% to 83%.
    - Recall for Class 1 improved from 62% to 75%.

Future Steps:

1. Unfreeze Additional Layers: Gradually unfreeze the BERT encoder layers to allow fine-tuning of deeper semantic representations.
2. Try Other Architectures: Experiment with models like BatteryBert**.**
3. Optimize Hyperparameters: Adjust batch size, learning rate

Doc HN PT 03

Basic Training Loop:

- **Model**: PatentBERT
- **Loss Function:** Cross-Entropy Loss

- **Optimizer**: AdamW
- **Learning Rate:** 0.001
- **Early Stopping:** Triggered on validation loss
- **Layers Trained:** Classifier , Pooler Layer, All layers from 7th epoch
- **Batch Size:** 8
- **Epochs Completed:** 10

Results:

- Test Accuracy: 73%
- Test F1-Score: 62%
- Test Precision: 53%
- Test Recall: 73%
- PR Curve AUC: 63%

Analysis:

- **Confusion Matrix Observations:**
    - Class 0 (Irrelevant Patents): 100% were correctly predicted (True Negatives).
    - Class 1 (Relevant Patents): 0% were correctly predicted (True Positives).
    - The model classified all patents as not relevant, regardless of their true label. This resulted in:
        - High Recall for Class 0 due to overemphasis on predicting irrelevance.
        - No Recall for Class 1, as no patents were classified as relevant.
- **Behavior After Unfreezing All Layers:**
    - Unfreezing all layers from the 7th epoch led to significant degradation in performance:
        - Training and Validation Accuracy dropped from 76% to around 50%.
        - Training and Validation Loss increased dramatically from 50% to around 80%.

- This suggests overfitting or catastrophic forgetting, where fine-tuning the lower BERT layers disrupted the pre-trained embeddings, resulting in a collapse of model performance.

Key Observations:

1. **Collapse in Class 1 Predictions**:
   - Despite high accuracy for irrelevant patents, the model completely failed to predict relevant ones, rendering it ineffective for use cases requiring recall or balance between the two classes.
2. **Impact of Layer Unfreezing**:
   - Unfreezing all layers without proper learning rate adjustments likely caused the model to overfit the training data while failing to generalize on the validation or test sets.

Future Steps:

1. **Refine Fine-Tuning Strategy**:
   - Gradually unfreeze deeper layers instead of all at once, starting with a few encoder layers and progressing only if performance improves.
   - Use smaller learning rates for pre-trained layers (e.g., differential learning rates) to prevent destabilization of pre-trained embeddings.
2. **Adjust Training Parameters**:
   - Introduce warmup steps in the learning rate schedule to stabilize training during fine-tuning.
   - Regularize the model using techniques like data, augmentation ,dropout and weight decay to prevent overfitting.

Conclusion:

This experiment highlights the risks of unfreezing all layers without careful optimization and showcases the delicate balance required when fine-tuning pre-trained models. While high accuracy for Class 0 might seem encouraging, the inability to predict Class 1 renders the model ineffective for real-world applications. Future iterations should focus on controlled

fine-tuning, addressing class imbalance, and improving generalization to achieve more reliable results.

Doc HN PT 04

Basic Training Loop:

- **Model**: PatentBERT
- **Loss Function:** Cross-Entropy Loss
- **Optimizer**: AdamW
- **Learning Rate:**
    - **Classifier and Pooler Layers:** 0.001
    - **Backbone Layers:** Adaptive, between 0.0001 and 0.00001 (adjusted using a custom LR reducer)**.**
- **Early Stopping:** Triggered on validation loss
- **Layers Trained:**
    - Classifier and Pooler layers initially
    - A percentage of the backbone layers were unfrozen iteratively when validation loss became stable (starting with 25%).
- **Batch Size:** 8
- **Epochs Completed:** 8

Results:

- Test Accuracy: 73%
- Test F1-Score: 62%
- Test Precision: 53%
- Test Recall: 73%
- PR Curve AUC: 36%

Analysis:

- **Confusion Matrix Observations:**

- Class 0 (Irrelevant Patents): 100% were correctly predicted (True Negatives).
- Class 1 (Relevant Patents): 0% were correctly predicted (True Positives).
- The model classified all patents as not relevant, regardless of their true label. This resulted in:
    - High Recall for Class 0 due to overemphasis on predicting irrelevance.
    - No Recall for Class 1, as no patents were classified as relevant.

Behavior After Layer Unfreezing:

- Starting from the 4th epoch, 25% of the frozen backbone layers were unfrozen iteratively whenever validation loss plateaued.
- **Observations:**
    - Training and Validation Accuracy dropped significantly (from 70% to 50%).
    - Training and Validation Loss increased (from 50% to 70%).
- **This suggests that:**
    - Fine-tuning a whole 25% of the back-bone destabilized pre-trained embeddings.
    - The model could not balance learning new representations while retaining prior knowledge.

Custom Functions Used:

- **Learning Rate Reducer** (custom_lr_reducer):
    - Dynamically reduced the learning rate for the backbone layers when validation loss plateaued.
    - Despite this, the model still failed to generalize for Class 1, indicating that the learning rate reduction alone was insufficient to stabilize training.
- **Layer Unfreezing Function** (unfreeze_layers):
    - In this case, **25% of the backbone layers unfrozen** iteratively.
    - Likely led to gradient instability, as deeper layers may require even smaller learning rates during fine-tuning.

Challenges Identified:

1. **Imbalanced Class Predictions**:
    - The model consistently defaulted to predicting all samples as Class 0, making it ineffective for detecting relevant patents (Class 1).
2. **Impact of Layer Unfreezing**:
    - While the unfreezing strategy attempted to iteratively train the backbone, it likely introduced gradient instability due to high learning rates for deeper layers.
3. **Low PR Curve AUC**:
    - A PR Curve AUC of **36%** shows poor precision-recall trade-offs, further confirming the model's inability to generalize well across classes.

Key Observations:

- While the custom unfreezing and learning rate adjustment sounded innovative, their implementation failed to improve performance.
- Performance degradation suggests that the backbone layer updates were too aggressive.

Future Steps:

- **Backbone Layers**: Very low learning rates (e.g., **0.00001–0.000001**).
- **Fine-Tune Fewer Layers**:
    - Instead of unfreezing **25%**, start with unfreezing just **1–2 encoder layers** closest to the classifier.

Conclusion

The initial implementation of PatentBERT revealed significant limitations in the model's ability to generalize, particularly for identifying relevant patents (Class 1). While irrelevant patents (Class 0) were classified with 100% accuracy, the complete failure to identify any relevant patents resulted in poor F1-score, precision, and recall for Class 1. The strategy of progressively unfreezing layers by 25% when validation loss plateaued led to instability, as

observed in the sharp drop in training and validation performance. This suggests that unfreezing large portions of the backbone layers without proper adjustments to learning rates can disrupt model convergence.

Doc HN PT 04.1

Basic Training Loop:

- **Model**: PatentBERT
- **Loss Function:** Cross-Entropy Loss
- **Optimizer**: AdamW
- **Learning Rate:**
    - Classifier and Pooler Layers: 0.001
    - Backbone Layers: Adaptive (between 0.0001 and 0.00001) using a custom LR reducer.
- **Early Stopping**: Triggered based on validation loss**.**
- **Layers Trained:**
    - Initially, Classifier and Pooler Layers were trained.
    - 2.5% of the Backbone Layers were unfrozen iteratively when validation loss plateaued.
- **Batch Size:** 8
- **Epochs Completed:** 11

Results:

- **Test Accuracy**: 79%
- **Test F1-Score**: 80%
- **Test Precision**: 84%

- **Test Recall**: 79%
- **PR Curve AUC**: 70%

Analysis:

1. **Confusion Matrix Observations:**
    - **Class 0 (Irrelevant Patents):**
        - 77% correctly predicted (True Negatives).
    - **Class 1 (Relevant Patents):**
        - 85% correctly predicted (True Positives).
    - **Misclassifications:**
        - 23% of patents incorrectly classified as potentially relevant
        - 15% of patents incorrectly classified as potentially not relevant
2. **Behavior After Layer Unfreezing:**
    - Starting from the 4th epoch, 2.5% of frozen backbone layers were unfrozen iteratively whenever validation loss plateaued.
    - **Observations:**
        - **Training and Validation Accuracy:**
            - Increased from 74% (train) and 71% (val) to 86% (train) and 82% (val) by the 6th epoch.
            - Training accuracy stabilized at 89%, while validation accuracy remained between 79% and 82%.
        - **Training and Validation Loss:**
            - Decreased from 53% (train) and 55% (val) to 33.5% (train) and 44% (val) in the 6th epoch.
            - Training loss further decreased to 19% by the final epoch.
            - Validation loss increased temporarily to 57% (7th epoch) before stabilizing at 46% (10th epoch).

3. **Key Observations:**
    - Model performance showed significant improvement with finer granularity of layer unfreezing (2.5%).
    - Despite achieving a better balance in predicting both classes, the PR Curve AUC (70%) and misclassifications indicate potential for further tuning.
    - Slight instability in validation loss suggests that the model may still require finer learning rate adjustments or additional regularization e.g Data augmentation.

Custom Functions Used:

1. **Learning Rate Reducer (custom_lr_reducer):**
    - Dynamically adjusted the learning rate for the backbone layers when validation loss plateaued.
    - Helped stabilize training but could not fully prevent validation loss fluctuations.
2. **Layer Unfreezing Function (unfreeze_layers):**
    - Gradually unfroze 2.5% of the backbone layers at a time, reducing the risk of catastrophic forgetting.
    - This incremental approach improved stability compared to more aggressive approach used previously

Challenges Identified:

1. **Validation Loss Fluctuations:**
    - Temporary increases in validation loss during layer unfreezing indicate instability in generalization.
    - This might result from inappropriate learning rates or insufficient regularization when deeper layers were unfrozen.

2. **PR Curve AUC (70%):**

- Indicates the model struggles to differentiate between classes at certain thresholds, especially in edge cases.
- False positives remain relatively high (23%), suggesting misclassification of irrelevant patents as relevant.

Future Steps:

1. **Fine-Tune Learning Rates and Regularization:**

   Apply additional regularization techniques such as data augmentation, dropout or weight decay for the backbone.

2. **Refine Layer Unfreezing Strategy:**

   Explore unfreezing deeper layers in smaller and bigger increments e.g 5 and 1.5% to see how the Model performs.

3. Use **focal loss** or similar techniques to emphasize harder-to-classify samples, reducing false positives and negatives.

4. **Regular Monitoring of Metrics:**
   Use additional metrics (e.g., **Matthews Correlation Coefficient**) to better capture class-wise performance nuances.

Conclusion

By refining the training strategy, including a gradual **2.5% layer-unfreezing approach** and adaptive learning rate adjustments, the updated model demonstrated significant improvements in performance. A **test accuracy of 7G%** and an **F1-score of 80%** reflected better balance in predicting both **irrelevant (Class 0)** and **relevant (Class 1)** patents.
However, the persistence of **false positives (23%)** and occasional fluctuations in validation loss highlight areas for further optimization. This iteration demonstrated that smaller, incremental unfreezing of layers, combined with an adaptive learning rate reducer, stabilized training and validation.

Doc HN PT 05

Basic Training Loop:

- **Model**: PatentBERT
- **Loss Function:** Cross-Entropy Loss
- **Optimizer:** AdamW
- **Learning Rate:**
  - Classifier and Pooler Layers: 0.001
  - Backbone Layers: Adaptive (between 0.0001 and 0.00001) using a custom LR reducer.
- **Early Stopping:** Triggered based on validation loss**.**
- **Layers Trained:**
  - Initially, only Classifier and Pooler Layers were trained.
  - Half of the encoder layers were unfrozen at epoch 6.
- **Batch Size**: 8
- **Epochs Completed:** 9

Results:

- **Test Accuracy**: 73%
- **Test F1-Score:** 62%
- **Test Precision**: 53%
- **Test Recall**: 73%
- **PR Curve AUC:** 33%

Analysis:

1. **Confusion Matrix Observations:**
   - **Class 0 (Irrelevant Patents):**
     - 100% correctly predicted (True Negatives).
   - **Class 1 (Relevant Patents):**
     - 0% correctly predicted (True Positives).
   - **Misclassifications:**
     - All patents classified as irrelevant, resulting in:
       - High recall for Class 0 due to consistent predictions of irrelevance.
       - No recall for Class 1, as no patents were identified as relevant.

2. **Behavior After Layer Unfreezing:**
    - **Starting at epoch 5, half of the encoder layers were unfrozen.**
    - **Observations:**
        - **Training and Validation Accuracy:**
            - Dropped from 77% (train) and 72% (val) to 49% (train) and 48% (val) after unfreezing.
        - **Training and Validation Loss:**
            - Increased from 53% (val) at epoch 4 to 69% after unfreezing.
            - Training loss peaked at 76% and later settled at 71%.
        - Persistent instability was observed in both metrics after unfreezing the deeper layers.

Custom Functions Used:

1. **Learning Rate Reducer (custom_lr_reducer):**
    - Dynamically adjusted the learning rate for backbone layers when validation loss plateaued.
    - Slowed down the performance degradation but failed to fully stabilize the model.
2. **Unfreeze Half Encoder Layers (unfreeze_half_encoder_layers):**
    - Unfroze the last half of the encoder layers without modifying other layers.
    - Introduced significant gradient instability due to the large number of layers unfrozen simultaneously.

Challenges Identified:

1. **Gradient Instability:**
    - Unfreezing half of the backbone layers at once caused a sharp drop in accuracy and an increase in loss**.**
2. **Class Imbalance in Predictions:**
    - The model consistently predicted irrelevance (Class 0) for all patents, failing to generalize to Class 1.
3. **Validation Loss Plateau:**
    - Despite using a custom learning rate reducer, validation loss remained elevated after unfreezing layers.

Future Steps:

1. **Incremental Unfreezing of Layers:**
    - Unfreeze smaller portions of the backbone
2. **Class-Weighted Loss Function:**

- Adjust the Cross-Entropy Loss with class weights to encourage better learning for Class 1.

3. **Data Augmentation:**
   - Enhance the dataset with data augmentation to improve class imbalance and generalization.

Conclusion:

The strategy of unfreezing half of the encoder layers simultaneously proved too aggressive, leading to significant gradient instability and poor generalization for Class 1 (relevant patents). While the learning rate reducer slowed the performance decline, it was insufficient to address the challenges introduced by abrupt unfreezing. The model completely failed to identify relevant patents, with a PR Curve AUC of only 33%. Future iterations should adopt a gradual layer-unfreezing strategy, incorporate class-weighted loss functions, these adjustments should balance predictions across classes, improve generalization, and stabilize training**.**

Observations so far

Classifier-Only Training Limits Learning; Pooler Fine-Tuning is Essential

Training only the classifier layer while keeping all the other layers of the BERT model frozen often results in suboptimal performance. This is because the learning capacity of the model is largely constrained when the deeper layers remain static. The frozen layers fail to adapt to the nuances of the new task, and the representations generated by the underlying transformer remain generic rather than task-specific.

To achieve better task adaptation and performance, it is crucial to unfreeze and fine-tune at least some of the other layers, such as the pooler layer. The pooler layer, being responsible for aggregating the sequence output into a fixed-size representation, plays a critical role in determining how well the model can interpret and represent input data for the specific downstream task. Training the pooler layer alongside the classifier allows the

model to adjust its high-level representations, thereby enhancing the overall learning process. It ensures that the model's deeper layers remain aligned with the specific requirements of the task while maintaining the generalization capability acquired during pre-training.

Low Learning Rates Mitigate Overfitting with Larger Unfrozen Backbones

While the classifier and pooler layers in a BERT model can benefit from a relatively high learning rate, such as 0.001, this approach becomes problematic when more layers in the model's backbone are unfrozen for training. The deeper layers of the transformer are highly sensitive to updates, and using a high learning rate, such as 0.001, can disrupt the pre- trained weights, leading to poor learning performance. This often manifests as low accuracies and high losses in both training and validation phases, as the model struggles to adapt effectively without destabilizing its pre-trained parameters.

Conversely, when a low learning rate, such as 0.00001, is applied to a substantial portion of the unfrozen backbone, the training process demonstrates stable and consistent improvements. Training loss decreases steadily, and training accuracy increases, reflecting effective learning of the task. However, the validation metrics often tell a different story. After an initial phase of improvement, validation loss tends to rise, and validation accuracy drops, indicating overfitting. The model memorizes the training data but fails to generalize to unseen data, a common challenge when a large portion of the model's capacity is leveraged during training.

This phenomenon underscores the importance of balancing the scope of fine-tuning with the appropriate learning rate. To prevent overfitting and ensure robust task-specific performance, it is often best to use a smaller learning rate alongside a selectively unfrozen backbone. By limiting the number of unfrozen layers, the model retains its generalization capabilities from pre-training while adapting sufficiently to the downstream task.

Imbalanced Data Can Favor Minority Class Despite Weighted Sampling

An imbalanced dataset, while often perceived as a challenge for classification tasks, does not always lead to poorer performance for the minority class. In some experiments, despite class 0 having nearly three times the number of samples as class 1, the model exhibited better performance on class 1. This finding challenges the assumption that class imbalance inherently disadvantages the minority class.

Upon closer inspection, most misclassifications by both older and newer versions of the model occurred in class 0. This phenomenon could be attributed to the sheer number of patents in class 0, which statistically increases the likelihood of misclassification occurring within this class. However, it raises an interesting contradiction: with 2,881 patents in class 0 compared to just 1,004 patents in class 1, it would seem logical for the model to perform better for class 0, given the abundance of training data and the use of a weighted sampler to mitigate class imbalance. Yet, this is not always the case.

One possible explanation lies in the distribution and complexity of features within each class. A larger class size can lead to higher intra-class variability, making it harder for the model to learn a consistent decision boundary. Meanwhile, the smaller class may inadvertently benefit from less variability, leading to more distinct and easily recognizable patterns during training. Additionally, the use of a weighted sampler could amplify the minority class's re presentation during training, further boosting its performance.

These observations highlight the nuanced dynamics of imbalanced datasets. Simply having more data for one class does not guarantee better performance for that class, and factors such as feature distribution, class variability, and sampling strategies play critical roles in shaping model outcomes.

.

Diving Deep into Encoder Layers

In the architecture of transformer-based models like BERT, the encoder layers play a crucial role in how the model processes and transforms input data. Understanding the behavior of these layers, especially when fine-tuning them for specific tasks, can provide valuable insights into model performance and efficiency.

Observations

The model consists of **24 total encoder layers** and each encoder layer is subdivided into **16** layers which I have grouped into **3** main sections based on their position within the model:

1. **10 Attention Layers:**
    - **6 Self-Attention Layers**: These layers are responsible for capturing relationships within the input sequence, allowing each token to attend to other tokens in the sequence. The self-attention mechanism enables the model to weigh the importance of various tokens relative to one another, which is key for contextual understanding.
    - **4 Output Layers:** These layers help refine the output of the attention mechanism, transforming the attention outputs into representations that can be passed to the subsequent layers**.**
2. **2 Intermediate Layers:** These layers typically consist of dense layers that help in transforming the attention outputs into a higher-dimensional space, allowing the model to learn more complex representations. They play an important role in enhancing the expressiveness of the model**.**
3. **4 Output Layers:** These layers consist of two dense layers followed by two layer normalization layers.
    - **Dense Layers:** These are responsible for further processing the information and learning more complex relationships within the data**.**
    - **Layer Norm Layers:** These help stabilize training by normalizing the output across the layers, ensuring consistent behavior and mitigating issues such as exploding or vanishing gradients**.**

Experiment Overview

Focus: The goal of the experiment is to assess how the model's behavior and performance change when unfreezing and fine-tuning different parts of the encoder layers.
Understanding the impact of this fine-tuning on both training efficiency and task

performance will provide valuable information into the optimal strategies for improving model accuracy and generalization.

Layer Segmentation: For the purposes of this experiment, the encoder layers are segmented into three distinct regions based on their proximity to the input and output**:**

- **Layers Closest to Input (0-7):** These layers are responsible for initially processing the raw input data. They capture lower-level features and representations, such as token embeddings and basic syntactic structures. Unfreezing these layers allows the model to adapt the basic features of the input data to the task at hand.
- **Middle Layers (8-15):** These layers capture more abstract and higher-level features, often reflecting deeper semantic relationships in the input sequence. Fine-tuning these layers can help the model focus on more complex task-specific patterns and improve its performance on downstream tasks**.**
- **Layers Closest to Output (16-23):** These layers refine the final representations and are responsible for task-specific outputs. Fine-tuning these layers has a direct impact on the model's performance on the target task. Unfreezing layers closer to the output typically allows the model to learn more detailed and task-specific features, but may risk overfitting if done excessively**.**

Key Points for Investigation:

1. **Impact of Unfreezing Different Layer Groups**: Investigating how unfreezing the layers closest to the input, middle, or output affects model performance can reveal which parts of the model need more fine-tuning for optimal task adaptation.
2. **Efficiency of Training:** By selectively unfreezing parts of the model, we can observe whether focusing on particular encoder layers leads to faster convergence and improved generalization or whether overfitting occurs when too many layers are unfrozen.
3. **Transfer of Knowledge:** Understanding how information flows through the layers and how different encoder sections contribute to model performance can provide deeper insights into the transfer of knowledge between pre-training and fine-tuning stages.

Layers Closest to Input: Experiment Details

Doc HN PT 05.3

Basic Training Loop:

Model: PatentBERT
Loss Function: Cross-Entropy Loss
Optimizer: AdamW

Learning Rate:

- Classifier and Pooler Layers: 0.001
- Backbone Layers: Adaptive (0.0001–0.00001) using a custom LR reducer.

- **Batch Size: 8**
- **Epochs Completed: 12**
- **Early Stopping:** Triggered based on validation loss.
- **Layer Training:**
  - Initially, only the Classifier and Pooler layers were trainable.
  - At epoch 6, encoder layers 0–7 were unfrozen.

Results

- Test Accuracy: **82%**
- Test F1-Score: **82%**
- Test Precision: **83%**
- Test Recall: **82%**
- Test PR Curve AUC: **6G%**
- Matthew Correlation Coefficient (Test MCC): **56%**

Analysis

1. **Confusion Matrix Observations:**

- **Class 0 (Irrelevant Patents):**
  - 84% correctly predicted (True Negatives).
- **Class 1 (Relevant Patents):**
  - 75% correctly predicted (True Positives).
- **Misclassifications:**
  - Class 0 misclassified as Class 1: 25%.

- Class 1 misclassified as Class 0: 16%.

2. Behavior After Layer Unfreezing

- **At Epoch 6:** Encoder layers 0–7 were unfrozen**.**
- **Observations:**
  - **Training Accuracy**: Increased from 78% to 96% by epoch 12.
  - **Validation Accuracy**: Improved to 81% by epoch 8, dipped to 76% by epoch 10, and rose slightly to 79% by epoch 12.
  - **Training Loss**: Decreased from 46% to 0.09% by epoch 12.
  - **Validation Loss**: Fluctuated, decreasing from 49% to 45% in epoch 7, but peaked at 71% by epoch 12.

Custom Functions and Techniques

1. Learning Rate Reducer (custom_lr_reducer)

- Dynamically adjusted learning rates for the backbone layers when validation loss plateaued.
- Helped slow down performance degradation but didn't completely eliminate validation loss fluctuations.

2. set_bert_layers_trainable(model, start_layer=0, end_layer=7, requires_grad=True)

- Used to set the required layers as trainable

Key Observations

1. Performance Trends: Unfreezing the input layers significantly improved training accuracy but led to overfitting, as indicated by rising validation loss.

Future Improvements:

- Use regularization more aggressively after unfreezing.

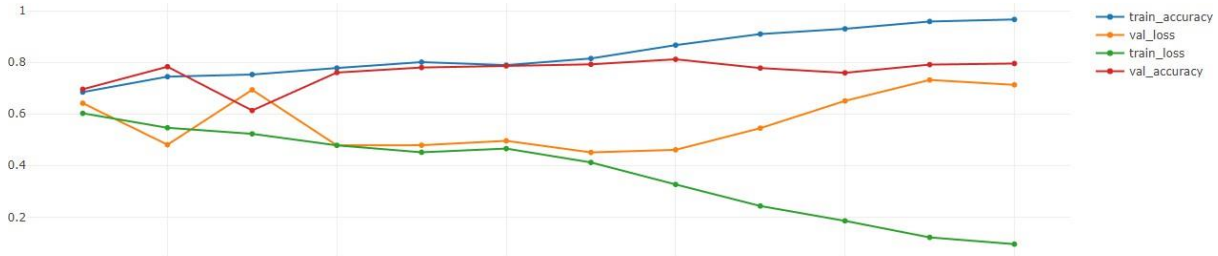
Conclusion

In this experiment, unfreezing the encoder layers closest to the input (layers 0-7) after six epochs improved training accuracy from 78% to 96%, but caused validation performance to fluctuate, with validation loss increasing from 45% to 71%, indicating potential

overfitting. The model achieved 82% accuracy, F1-score, precision, and recall, but struggled more with identifying relevant patents (75% accuracy) compared to irrelevant

ones (84%). The custom learning rate reducer helped stabilize training but didn't fully resolve validation loss fluctuations.

The trend to watch out for in this experiment are the differences in how the model converges or overfits via the losses and accuracy, how low/high can it go, at what point does it start to overfit?



From the chart, the behavior of the model can be seen right after the top encoder layers are unfrozen from the 6th epoch, loss improves slightly at the 7th but leaving it unfrozen for more epochs causes the model to quickly start memorizing or overfitting.

Middle Layers: Experiment Details

Doc HN PT 05.4

Basic Training Loop:

Model: PatentBERT
Loss Function: Cross-Entropy Loss
Optimizer: AdamW

Learning Rate:

- **Classifier and Pooler Layers:** 0.001
- **Backbone Layers**: Adaptive (0.0001–0.00001) using a custom LR reducer.
- **Batch Size:** 8

- **Epochs Completed**: 10
- **Early Stopping:** Triggered based on validation loss.
- **Layer Training:**
    - Initially, only the Classifier and Pooler layers were trainable.
    - At epoch 6, encoder layers 8–15 were unfrozen.

Results

- Test Accuracy: **81%**
- Test F1-Score: **81%**
- Test Precision: **82%**
- Test Recall: **81%**
- Test PR Curve AUC: **66%**
- Matthew Correlation Coefficient (Test MCC): **54%**

Analysis

1. **Confusion Matrix Observations:**

- **Class 0 (Irrelevant Patents):**
    - 84% correctly predicted (True Negatives).
- **Class 1 (Relevant Patents):**
    - 72% correctly predicted (True Positives).
- **Misclassifications:**
    - Class 0 misclassified as Class 1: 28%.
    - Class 1 misclassified as Class 0: 16%.

2. **Behavior After Layer Unfreezing**

- **At Epoch 6:** Encoder layers 8–15 were unfrozen**.**
- **Observations:**
    - **Training Accuracy**: Increased from 78% to 91.7% by epoch 10.
    - **Validation Accuracy**: Improved from 75% to 80% by epoch 9, dipped to 75% in epoch 10.

- **Training Loss**: Decreased from 46% to 21% by epoch 10.
- **Validation Loss**: Fluctuated, decreasing from 51% to 48% in epoch 8, but peaked at 55% by the 10th epoch.

Custom Functions and Techniques

1. Learning Rate Reducer (custom_lr_reducer)

- Dynamically adjusted learning rates for the backbone layers when validation loss plateaued.
- Helped slow down performance degradation but didn't completely eliminate validation loss fluctuations.

2. set_bert_layers_trainable(model, start_layer=8, end_layer=15, requires_grad=True)

- Used to set the required layers as trainable

Key Observations

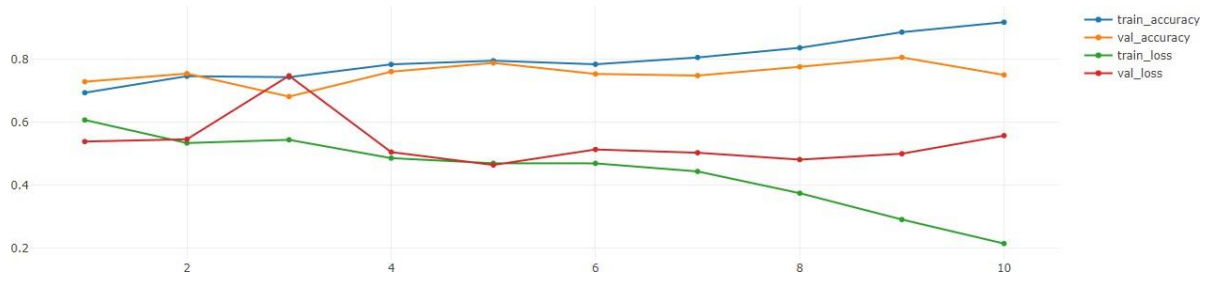
Performance Trends: Unfreezing the middle layers significantly improved training accuracy but led to overfitting, as indicated by rising validation loss.

The model is slower to overfit

Future Improvements:

- Use less layers to see how it affects overfitting
- Try freezing and unfreezing other layers in conjunction

Conclusion:



The difference between this chart and the former is slightly different, in this chart starting from the 6th epoch, the model is slower to converge, i.e the losses are slower to react and descend when the middle layers have been unfrozen, in similar fashion, it is also slower to overfit. From epoch 6-8 the general movement is downwards before it starts to overfit. The same could be said with the accuracies, it starts to slowly increase before reducing at the 9th epoch. These 2 charts could tell that the deeper layers are faster to converge and overfit in comparison to the layers close to the output.

The experiment demonstrated that unfreezing encoder layers 8 –15 at epoch 6 significantly improved training accuracy (from 78% to 91.7%) but led to overfitting, as seen in fluctuating validation loss and a decline in validation accuracy by epoch 10. The test results, including an F1-score of 81% and MCC of 54%, indicate effective task performance but highlight challenges in distinguishing between classes, particularly for minority class instances.

Custom techniques like dynamic learning rate adjustment helped stabilize performance but did not fully resolve overfitting. Future experiments should in which unfreezing fewer layers, varying layer ranges, and employing additional regularization techniques to achieve better generalization could be explored.

Layers Closest to Output: Experiment Details

Doc HN PT 05.5

Basic Training Loop:

Model: PatentBERT
Loss Function: Cross-Entropy Loss
Optimizer: AdamW

Learning Rate:

- **Classifier and Pooler Layers:** 0.001
- **Backbone Layers**: Adaptive (0.0001–0.00001) using a custom LR reducer.
- **Batch Size:** 8

- **Epochs Completed**: 13
- **Early Stopping:** Triggered based on validation loss.
- **Layer Training:**
    - Initially, only the Classifier and Pooler layers were trainable.
    - At epoch 6, encoder layers 16–23 were unfrozen.

Results

- Test Accuracy: **80%**
- Test F1-Score: **81%**
- Test Precision: **82%**
- Test Recall: **80%**
- Test PR Curve AUC: **67%**
- Matthew Correlation Coefficient (Test MCC): **55%**

Analysis

1. **Confusion Matrix Observations:**

    - **Class 0 (Irrelevant Patents):**
        - 81% correctly predicted (True Negatives).
    - **Class 1 (Relevant Patents):**
        - 78% correctly predicted (True Positives).
    - **Misclassifications:**
        - Class 0 misclassified as Class 1: 22%.
        - Class 1 misclassified as Class 0: 19%.

2. Behavior After Layer Unfreezing

- **At Epoch 6:** Encoder layers 8–15 were unfrozen**.**
- **Observations:**
    - **Training Accuracy**: Increased from 79% to 83% by 8. Increased to 95.7% by 13th epoch.
    - **Validation Accuracy**: Improved from 78.4% to 83% by epoch 8, dipped to 75.7% in epoch 12, Increased a bit to 79% by epoch 13.
    - **Training Loss**: Stayed around 45% from 6th to 7th epoch, Decreased steadily to 12% by 13th epoch.
    - **Validation Loss**: Stayed around 49% and 50% between 6th and 7th epoch, decreased to 43.5% by 8th epoch, Increased slowly to 57.8% by 11th epoch, rose to 80% by epoch 12 and dropped to 72% at 13th epoch.

Custom Functions and Techniques

1. Learning Rate Reducer (custom_lr_reducer)

- Dynamically adjusted learning rates for the backbone layers when validation loss plateaued.
- Helped slow down performance degradation but didn't completely eliminate validation loss fluctuations.

2. set_bert_layers_trainable(model, start_layer=16, end_layer=23, requires_grad=True)

- Used to set the required layers as trainable

Key Observations

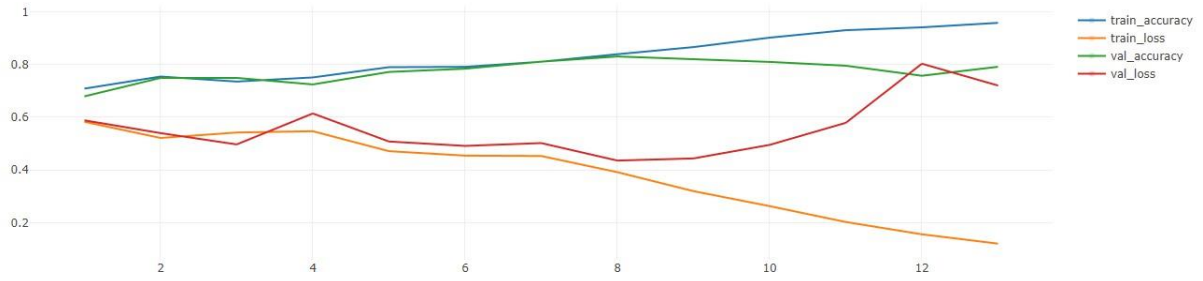
- Model is much slower to overfit
- Validation and training loss increase together and then decrease after unfreezing, showing similar trends before overfitting later.

Future Improvements

1. **Selective Unfreezing:**
   Experiment with unfreezing fewer layers closer to the output (e.g., layers 20–23) to reduce overfitting and validation loss fluctuations.

2. **Dynamic Fine-Tuning Strategies:**
   Combine freezing and unfreezing across different epochs or apply progressive layer unfreezing to enhance generalization.

Conclusion



In this scenario, the model exhibits delayed overfitting following the unfreezing of layers, as reflected in the similar trends observed in training and validation loss between the 6th and 8th epochs. This alignment suggests stability in the learning process and an enhanced capacity for the model to generalize effectively as training loss decreases. The results indicate that unfreezing layers closer to the output introduces minimal disruption to the model's learned representations. When changes occur, they remain sufficiently controlled to enable generalization without leading to overfitting.

Learned Hypothesis

The behavior of the model when unfreezing different encoder layer groups highlights that the depth of unfreezing within the BERT architecture has a profound impact on the trade- off between convergence speed and generalization:

1. **Layers Closest to Input (0–7):** Unfreezing these layers accelerates convergence in training but introduces a significant risk of overfitting. These layers encode fundamental input-level features, and changes here can drastically alter the model's learned representations, leading to instability in validation performance and poor generalization.

2. **Middle Layers (8–15):** These layers represent the core of the model's feature extraction capability. Unfreezing them slows down both convergence and overfitting, enabling more controlled learning. However, this segment still exhibits a risk of overfitting after extended training, although it is slower compared to the input layers.
3. **Layers Closest to Output (16–23):** Unfreezing these layers causes the least disruption to the model's representations, resulting in a slower overfitting process and better stability in training and validation trends. These layers handle task- specific transformations, and changes here primarily refine the model's outputs without drastically altering its overall feature space.

General Hypothesis:

Unfreezing deeper layers closer to the input causes greater sensitivity to overfitting, as these layers govern foundational feature extraction. Conversely, unfreezing layers closer to the output has a minimal destabilizing effect and allows for controlled fine-tuning, making it a safer strategy for achieving generalization without compromising stability. Middle layers offer a balance but require careful monitoring to prevent overfitting during extended training.

This pattern suggests that selective and progressive unfreezing, combined with adaptive learning rates, can optimize the trade-off between generalization and overfitting across different encoder layer groups.

Experiment 5.5.2: Unfreezing Encoder Layers 18–20

Overview

This experiment was part of a series evaluating fine-tuning BERT layers incrementally. For **Experiment 5.5.2**, the focus was on unfreezing encoder layers 18–20 at the 6th epoch while keeping other layers frozen, to observe the effects on task-specific performance and training stability. This approach aimed to strike a balance between introducing representational flexibility and minimizing destabilization.

Selection Rationale:

Experiment 5.5.2 was chosen as the representative configuration because it produced some of the best results in terms of accuracy, F1-score, and MCC. Similar experiments with other layer ranges, such as **16–18 (Experiment 5.5.1)** and **21–23 (Experiment 5.5.3)**, were conducted but did not achieve comparable stability or performance metrics.

Results Summary

- **Test Metrics:**
  - **Accuracy:** 83%
  - **F1-Score:** 83%
  - **Precision:** 84%
  - **Recall:** 83%
  - **MCC (Matthew's Correlation Coefficient):** 58%
  - **PR Curve AUC:** 67%
  - **ROC Curve AUC:** 87%
- **Confusion Matrix Analysis:**
  - **Class 0 (Irrelevant Patents):**
    - **Correctly identified:** 86% (True Negatives)
    - **Misclassified**: 14%
  - **Class 1 (Relevant Patents):**
    - **Correctly identified:** 75% (True Positives)
    - **Misclassified:** 25%

Training and Validation Behavior

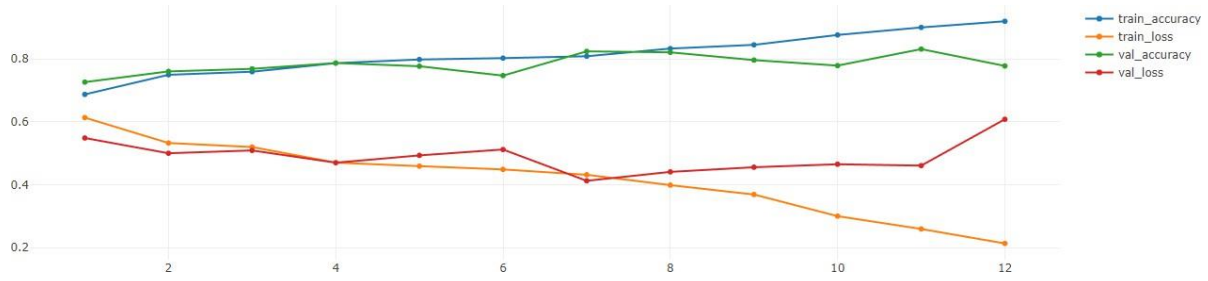
1. **Training Accuracy:**
   - Gradually increased, reaching **G2%** by the 12th epoch.
   - The steady improvement indicates a controlled learning process without abrupt overfitting tendencies.
2. **Validation Accuracy:**
   - Rose to a peak of **82.4%** at the 7th epoch.
   - Declined to **77.8%** by the 10th epoch, rebounded to **83%** at the 11th, and then dropped sharply to **77.7%** at the 12th epoch.
   - This pattern reflects minor instability after prolonged training but suggests the model's ability to recover generalization temporarily.
3. **Training Loss:**
   - Consistently decreased to **21%** by the last epoch, signifying effective optimization.
4. **Validation Loss:**
   - Decreased from **51%** to **41%** by the 7th epoch, showing improved generalization during early fine-tuning.

- Increased gradually to **46%** at the 11th epoch and spiked to **60.8%** at the 12th epoch, signaling overfitting as the model memorized patterns rather than generalizing.

Key Observations:

- Unfreezing layers 18–20 allowed fine-tuning of task-specific output layers while maintaining the stability of foundational representations.
- The resulting performance metrics (Accuracy: 83%, F1-Score: 83%, MCC: 58) indicate strong generalization compared to experiments involving broader unfreezing ranges, where overfitting occurred more rapidly.
- The validation trends suggest that unfreezing only a subset of output- adjacent layers (18–20) led to a more stable training process compared to larger ranges (e.g., 16–23).
- However, slight overfitting appeared after the 11th epoch, evidenced by increased validation loss and declining validation accuracy.
- Class 0 (irrelevant patents) showed higher recognition rates (**86%**) compared to Class 1 (**75%**). This indicates a class imbalance in model sensitivity, where the model is more confident in identifying irrelevant patents.
- The ROC Curve AUC of **87%** suggests that the model is effective at distinguishing between the two classes, with relatively high recall (**83%**) and precision (**84%**).
- The gradual changes in accuracy and loss curves highlight the controlled impact of unfreezing layers 18–20. Compared to experiments with larger unfreezing ranges, this configuration proved less prone to destabilizing the model's learned representations.

Conclusion



Experiment 5.5.2 was chosen as the most effective configuration from a series of experiments with different layer ranges. **Unfreezing encoder layers 18–20** struck a good balance between the model's ability to adapt its internal representations to the task and its

capacity to maintain consistent performance on unseen data.. This setup produced one of the highest-performing models, achieving an accuracy of 83%, F1-Score of 83%, and MCC of 58%. The controlled training dynamics and improved stability make this configuration a promising approach for fine-tuning tasks requiring minimal overfitting and high precision.

Gradient Boost Ensemble Model Experiment

Model : XGBoost

Description : Trained on inputs which are the probabilities of the training data for the Old Model and a new PatentBert Model (notebook)

Results Summary

- **Test Metrics:**
  - **Accuracy:** 82%
  - **F1-Score:** 70%
  - **Precision:** 64%
  - **Recall:** 77%
  - **MCC (Matthew's Correlation Coefficient):** 58%
  - **PR Curve AUC:** 71%
  - **ROC Curve AUC:** 87%
- **Confusion Matrix Analysis:**
  - **Class 0 (Irrelevant Patents):**
    - **Correctly identified:** 84% (True Negatives)
    - **Misclassified**: 14%
  - **Class 1 (Relevant Patents):**
    - **Correctly identified:** 77% (True Positives)
    - **Misclassified:** 25%

Key Observations

1. **Performance Metrics:**
   - The **Accuracy** of the ensemble model (82%) indicates strong overall classification performance.
   - **ROC Curve AUC (87%)** and **PR Curve AUC (71%)** highlight good discrimination ability and performance in handling imbalanced data.

- Despite the high accuracy, the **F1-Score (70%)** reflects room for improvement in balancing precision and recall, particularly for relevant patents.

2. **Class-Specific Insights:**
   - **Class 0 (Irrelevant Patents):**
     - Achieved high **True Negative Rate (84%)**, demonstrating the ensemble model's ability to filter out irrelevant patents effectively.
     - Relatively low misclassification rate (14%) for this class.
   - **Class 1 (Relevant Patents):**
     - While the **True Positive Rate (77%)** shows the model's ability to correctly identify relevant patents, the **Precision (64%)** indicates challenges in avoiding false positives.
     - The higher misclassification rate (25%) for this class suggests potential improvements in recognizing subtle patterns for relevant patents.
3. **Ensemble Effectiveness:**
   - Combining the Old Model and the PatentBERT model through XGBoost led to a notable improvement in metrics such as ROC Curve AUC (87%).
   - The integration effectively leveraged the strengths of both models, achieving better generalization and robustness compared to individual model performance.

Conclusion

The Gradient Boost Ensemble model successfully combined the outputs of the Old Model and the new PatentBERT model, achieving strong overall classification performance, particularly in distinguishing irrelevant patents. However, while the model demonstrated excellent discrimination capability (as indicated by ROC AUC), challenges remain in improving precision and reducing misclassification for relevant patents.

Model	Training			Validation				Exp. Link
	Recall%	Precision%	Accuracy%	Recall%	Precision%	Accuracy%	MCC %	
Baseline model (Old)	88	92	88	78	84	78		Link
PatentBert (**HNPT 05.3.2**)	96.525	96.54	96.525	81	84	81	57	Link
BatteryBert (**HNBT 05.3.2**)	90.6	90.7	90.6	80	82	80	53	Link
Patent Bert-Weighted loss 1/2	83	83.7	83	81	83	81	57	Link
Battery Bert Weighted loss 1/2	69	72	69.67	78	76	78	38	Link
Ensemble	99.7	86.2	95.7	77	64	82	58	Link

Key Takeaways:

PatentBERT with the weighted loss shows a balance between recall, precision, and MCC, making it a robust choice for scenarios where generalization is critical.

BatteryBERT models underperform compared to their **PatentBERT** counterparts, both with and without weighted loss.

The ensemble model, while strong in training, sacrifices validation precision, indicating potential overfitting or misalignment in integrating predictions from the base models.