Scientific Abstract Classifier using a SetFit and Contrastive Learning Fine-tuned Language Model

Jonathan Setiawan

Ritsumeikan University

January 13, 2026

1. Introduction

1.1 Project Overview

This is a project conducted as a part of the Artificial Intelligence class requirements for the final project which is to create a scientific abstract classification with a fine-tuned language model that is able to classify a given abstract into one of three defined classes or domains. For this project, the classes are split into bioinformatics, neuroscience, and materials science. Furthermore, this project will use SetFit (Sentence Transformer Fine-tuning) which is a state-of-the-art (SOTA) framework that combines contrastive learning with the concept of fine-tuning to train a model with reasonable accuracy without the need for large volumes of training data. The base model used in this project is SciBERT which is a BERT variant that is specifically pre-trained on millions of scientific publications from Semantic Scholar that provides domain specific vocabulary and contextual understanding that are crucial for abstract classification.

In addition to the new fine-tuning framework, this project also proposes a new method of data augmentation by introducing "context injection," which improves the training samples with domain-specific hints extracted through TF-IDF analysis. This method is similar to how humans may use a thesaurus to find synonyms to better understand a word which is through other similar words being used as examples which is the inspiration behind context injection. In more detail, this augmentation method identifies keywords that are distinct to each scientific domain and appends contextual information to the original abstracts which helps the model be able to recognize and build that sense of pattern recognition better. This project also includes a 5-fold cross validation, metrics, and other visualizations to support the results that were found through the span of the project.

1.2 Motivation

Text classification is one of the domains of NLP that is growing rapidly in relevance due to the growth of Large Language Models (LLMs). In academic context in particular, the ability to classify papers based on their domain is a very important task as most researchers often struggle to figure out how to narrow their search when they are just starting out. Moreover, managing millions of publications annually through manual classification is incredibly impractical and a huge waste of resources that could be optimized using technology. Specifically, scientific text classification presents a unique challenge for NLP systems as they are often dense in highly specialized vocabulary, domain-specific jargon, and

complicated subtle semantic nuances that make it stand out among the other fields of text classification.

Traditional machine learning approaches for text classification such as the ones based on TF-IDF features with Support Vector Machines (SVMs), often struggle to comprehend the subtle nuances in scientific writing.Although deep learning models such as BERT (Bidirectional Encoder Representations from Transformers) have revolutionized NLP, they are impractical for domain-specific text classification as they would require more computational resources and amount of labeled data for accurate fine-tuning. This presents a problem for researchers or institutions that have limited access to high-performing computers or a large repertoire of datasets. The SetFit framework attempts to address this issue by allowing these institutions to create high performing models on less data.

2. Literature Review

2.1. Traditional Text Classification Approaches

Text classification has been a fundamental task in NLP for decades that started out from manually creating and defining the features or statistical model. The Bag-of-Words (BoW) model for example, represents documents as simply unordered collections of word frequencies but was used as the foundation for many future approaches. Term Frequency-Inverse Document Frequency (TF-IDF) soon came as a direct competitor schematic that evaluates word importance by considering the frequency of that word appearing within a particular document and their rarity across the corpus which successfully enables it to identify specific vocabulary for classification tasks (Salton & Buckley, 1988).

Other machine learning algorithms including Naive Bayes, Support Vector Machines (SVMs), and Random Forests were proposed as alternatives when combined with TF-IDF features for document classification. In particular, SVMs stood out with its performance by finding optimal hyperplanes to identify classes in a high-dimensional feature space. However, all of these approaches still treat words as independent tokens and fail to understand contextual word relationships, semantics, and complicated syntax in writing. This limitation became ever so more evident in scientific text classification where the jargons are domain-specific and often heavily relied on context.

2.2. Transformer-based Language Models

Transformer architecture based language models were first introduced by Vaswani et al. (2017) which marked a paradigm shift in NLP through its novel concept of self-attention mechanism. Up until that point, neural networks process sequences in order, but transformers compute attention weights in parallel across all positions in a sequence which allow the model to comprehend long-range dependencies better. This self-attention mechanism also allows each token to attend to every other token in the input which equates to learning contextual representations that dynamically adapt based on the surrounding text.

BERT (Bidirectional Encoder Representations from Transformers) was introduced by Devlin et al. (2019) and applied the transformer architecture to create deep bidirectional language models through pre-training on massive text corpus. BERT worked by using a transformer's encoders to read a specific text from both ways which means that it would understand the context read from both ways, this contributed to its tasks which was to be able to guess hidden words and next sentence prediction. In technical terms, BERT's pre-training objectives of Masked Language Modeling (hidden words) and Next Sentence Prediction allowed the model to learn from a rich environment which can be further fine-tuned for downstream tasks on smaller task-specific datasets such as text classification. With how BERT became, other variants of it subsequently emerged such as SciBERT (Beltagy et al., 2019), which was pre-trained on 1.14 million scientific papers from Semantic Scholar. SciBERT utilizes vocabulary that is taken specifically from scientific literature which results in its better performance in scientific NLP tasks compared to the general task BERT model, hence giving it the SciBERT name.

2.3. Contrastive Learning and SetFit Framework

In recent times, Contrastive Learning is a contemporary technique that was developed for learning meaningful representations by training the models to distinguish similar and unlike samples, hence the contrastive nature of the name. Fundamentally, this method starts with creating pairs or groups of examples, where the model learns to bring representations of similar samples in the embedding space while pushing away samples that are not similar. For reference, Sentence-BERT (Reimers & Gurevych, 2019) modified this approach for sentence embeddings by using siamese and triplet network structures with BERT which resulted in the ability for BERT to classify dynamically better in the training phase.

This ultimately leads us to the SetFit framework (Tunstall et al., 2022) which is a recent breakthrough in few-shot text classification by combining the contrastive learning with fine-tuning concepts. SetFit starts with fine-tuning a pre-trained sentence transformer model using contrastive learning on generated sentence pairs from the labeled data. Then, it trains the classification head on the embeddings that were made from the previous fine-tune result. This expedited the competitiveness for smaller models to compete with large language models and requiring significantly less labeled data and computational resources. The model also uses a concept known as cosine similarity loss during contrastive training which tells the model to create embeddings where the angular distance between vectors reflects the similarity of the word which connected yet another piece of the puzzle.

3. Methodology

The following diagram illustrates the complete system pipeline from data collection to the final classification:

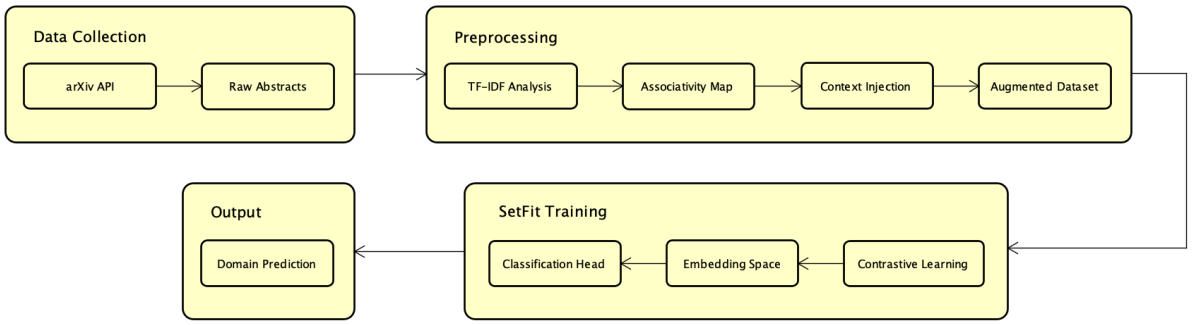

Figure 1. End-to-end system pipeline for scientific abstract classification

3.1. Data Acquisition

The dataset utilized in this project was acquired by running a python script to collect it from the arXiv open-access repository using its public API. arXiv is a very well known and reputable repository for finding literature and research which is a preprint server that hosts over two million scholarly literature in numerous domains. In particular, it contains a sufficient amount of content for bioinformatics, neuroscience, and materials science which matches the project's intent. The python query script collects abstracts from three category codes being 'q-bio.QM,' (Quantitative Methods in Biology, representing Bioinformatics), 'q-bio.NC,' (Neurons and Cognition, representing Neuroscience), and 'cond-mat.mtrl-sci,' (Materials Science within Condensed Matter Physics).

The data collection process resulted in 350 abstracts in bioinformatics, 350 abstracts in materials science, and 337 in neuroscience. Initially, the API responded in XML format which was then parsed to extract the paper titles and abstracts along with preprocessing to remove newline characters and whitespaces. The script also used a delay of 3 seconds between each consecutive API request and was connected to the internet through a Virtual Private Network (VPN) using Proton VPN to follow arXiv's server usage guidelines and avoid running into any issues overloading their server. To prevent any bias, duplicate abstracts were removed, abstracts shorter than 50 characters were filtered out, and the final dataset outputted in a .csv file format had the order randomized to prevent class ordering bias.

3.2. Preprocessing

The preprocessing pipeline used in this project starts with constructing an associativity map that is used to map the relationship between vocabulary terms and scientific domains. This uses the TF-IDF vectorization technique on abstracts grouped by domain label which will identify the top 60 most distinct terms for each field. For terms that receive high TF-IDF scores within a domain but low scores in others signify that it is a domain-specific jargon that is easily distinguishable in comparison to other domains for classification.

For each keyword, the preprocessing system will query WordNet (Miller, 1995) to extract synonyms to help build the associativity map with semantic alternatives that the model can use to better graph the words during training. Thereafter, termed context injection begins with using the associativity map to augment the training samples. When processing an

abstract, the system identifies words that appear in the associativity map and creates context things in the form of keyword-domain-synonym triplets. The hints are then attached to the original abstract csv file that is separated using separator tokens which creates new augmented samples. Ultimately, this creates more training samples by keeping the original and context-injected versions of the abstract that leads to more samples.

3.3. Model selection

I specifically chose SciBERT as the base model for this classification task due to complementary experimentation that was conducted prior as a part of the learning phase. It was revealed that even with context injection and SetFit fine-tuning general task models performed lackluster in comparison to SciBERT and domain-specific alternatives such as Specter 2 (Cohan et al., 2020; Singh et al., 2022), resulted in lackluster results compared to SciBERT. Moreover, SciBERT was pre-trained on a massive corpus of 1.14 million scientific papers across multiple domains as opposed to general web text that BERT is based on. This results in SciBERT having the home advantage of already being familiar with certain terminologies during the training process which lessens the need to second-guess itself.

Further comparative studies have shown that domain-specific models have consistently outperformed their larger general task models on domain-relevant tasks. Although alternative scientific language models exist such as BioBERT for biomedical text and PubMedBERT for clinical literature, SciBERT's dataset spanned across a broader moat for various scientific fields which suit the context of multi-domain classification. SciBERT also follows the architecture behind the BERT-base configuration with approximately 110 million parameters, 12 transformer layers, 768-dimensional hidden representations, and 12 attention heads which provide enough capacity for handling complex semantic understanding.

3.4. Fine-tuning Architecture

The fine-tuning architecture used in this project follows the SetFit framework's training process which is tailor-made for few-shot learning scenarios such as this. The following diagram illustrates the SetFit training process:

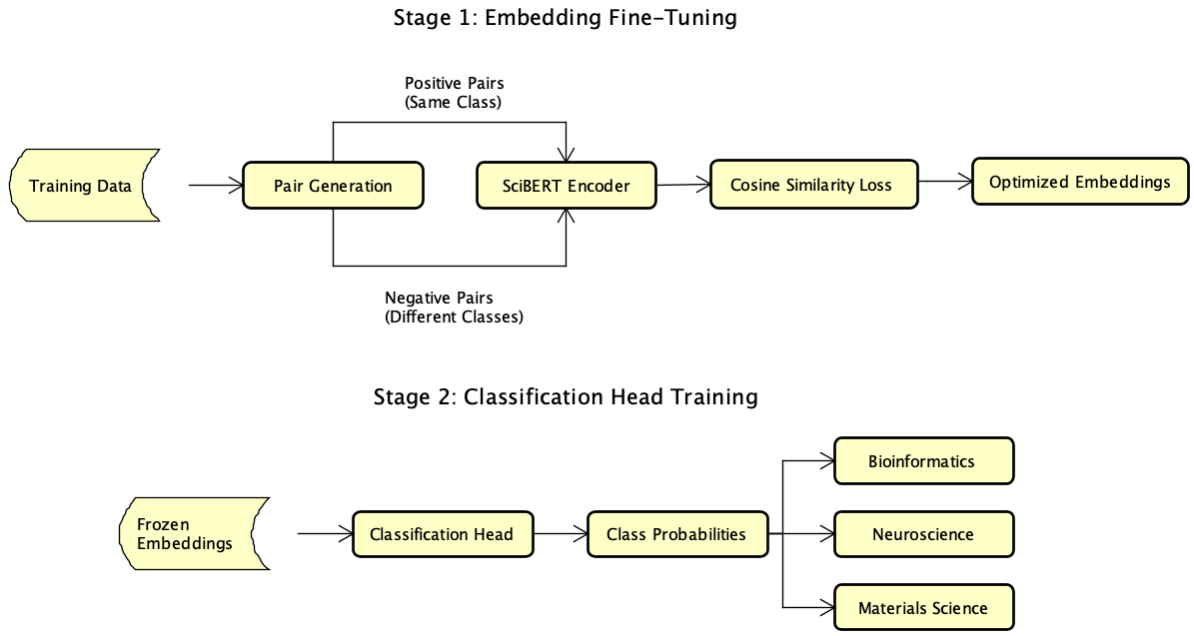

Figure 2. SetFit architecture flowchart

Figure 2 shows a two-stage SetFit training architecture flowchart that shows contrastive learning followed by classification head training. In the first stage, the pre-trained SciBERt model goes through contrastive fine-tuning using the positive and negative pairs of sentences generated from the training data during pair generation. In each training example, positive pairs are created by combining texts from the same class while negative pairs combine texts from different classes. We then train the model to minimize the cosine similarity loss which attempts to maximize the cosine similarity for positive pairs and minimize for negative pairs. In the end, this will generate the optimized embeddings to be used in the next phase. Moreover, the contrastive learning stage transforms the embedding space from its original pre-trained configuration into a task-specific one.

The second stage is about training a lightweight classification head on top of the frozen embeddings. The classification head maps the embeddings to class probabilities which allow us to get the final prediction, in this case, it would tell us which of the three chosen domains a certain abstract belongs to.

4. Experiments

4.1. Implementation of k-fold cross validation

The project requirements detailed the necessity for implementing a k-fold cross validation procedure. To fulfill that criteria, this project utilizes a 5-fold cross-validation method, this is done to achieve a more robust model by minimizing bias. The following diagram visualizes how the dataset is separated into 5 folds:

Iteration	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Output Accuracy
Iteration 1	**VALIDATION**	Training	Training	Training	Training	Accuracy_1
Iteration 2	Training	**VALIDATION**	Training	Training	Training	Accuracy_2
Iteration 3	Training	Training	**VALIDATION**	Training	Training	Accuracy_3
Iteration 4	Training	Training	Training	**VALIDATION**	Training	Accuracy_4
Iteration 5	Training	Training	Training	Training	**VALIDATION**	Accuracy_5
Final Result						**Mean ± Std Dev**

Figure 3. 5-fold cross validation procedure

Figure 3 shows what a 5-fold cross validation procedure looks like in tabular format. This method is done to make sure that every sample is validated exactly once. Initially, the entire dataset is divided into 5 equal sized sections or folds, each fold is used once as validation and for the rest of the folds in that specific iteration, they will be used for training. This kind of method reduces variance and improves how robust the system is.

The cross validation uses random shuffling with a fixed seed of 42 before splitting to make sure that the class distributions are preserved across the folds. In each fold, a fresh model is initialized from the pre-trained SciBERT weights, trained on the training set specifically for that fold and evaluated in the validation set. We can also calculate performance metrics of the trained model which includes metrics such as accuracy (amount

of total correct predictions out of all samples), precision (ratio of true positive predictions to the total predicted positives), recall (ratio of true positives to all actual positives) , and F1 score (harmonic mean of precision and recall) which is calculated in each fold. The final result that is displayed in the terminal is the mean and standard deviation across all folds.

4.2. Context Injection

The context injection technique used in this project is a novel data augmentation technique that I came up with after being inspired by I would use an online thesaurus to understand difficult words I have never heard of when reading novels. This method takes advantage of the fact that scientific writing inherently consists of an abundance of domain-specific vocabulary that strongly correlates to that field which makes it a very important pivot point, hence if we could improve the efficiency in this process, theoretically we can enhance the data. The following diagram illustrates how context injection takes place on an abstract:

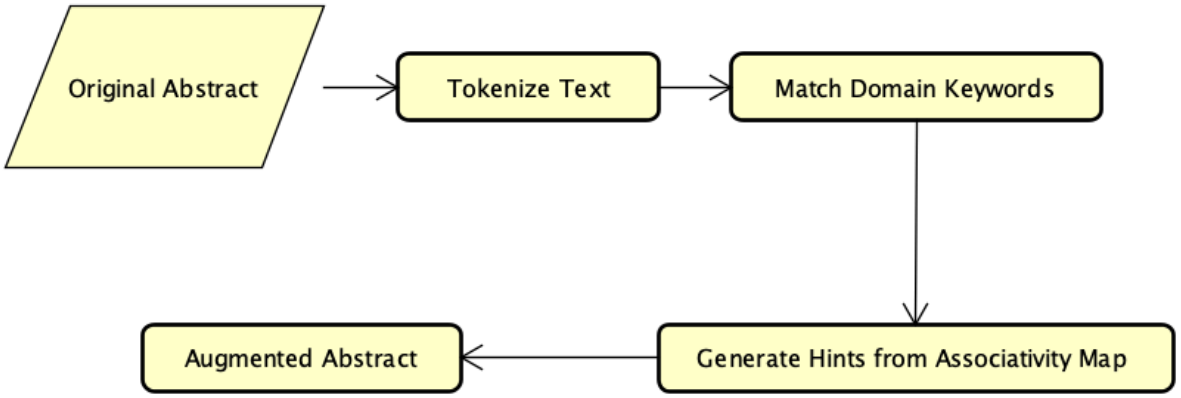

Figure 4. Context injection flowchart

Figure 4 indicates how domain-specific hints are attached to the original abstracts to generate augmented abstracts. Initially, the original abstract is taken as input and the context injection system tokenizes the text and identifies words that appear in the pre-computed associativity map that was built prior during the preprocessing phase. For each matched keyword, the system retrieves its corresponding domain label and synonyms from the associativity map which creates a hint string that encodes this information. However, to avoid overaugmenting the data, only three hints are attached to each abstract. Using a separator token and a hint marker, we are able to append the hints to the original abstract while preserving the initial dataset.

4.3. Training Configuration

The training configuration starts with setting the batch size to 16 samples due to hardware limitations and although larger batch sizes could potentially improve training mechanics, it is too time-consuming for consumer GPUs. The contrastive learning was set to have 3 iterations for pair generation which means that we create 3 sets of positive and negative pairs from the training data. We used 1 epoch set per iteration as contrastive learning

converges very fast and further additions to the epoch count could risk overfitting. The objective of the training is cosine similarity loss which adjusts the angular dynamics between embeddings that form the basis for classification. A fixed random seed of 42 is also used throughout data shuffling, model initialization, and dropout sampling to increase reproducibility during different runs.

5. Results

5.1. Performance Metrics

Our fine-tuned model was able to achieve over 98% accuracy after the 5-fold cross validation and showed little to no variance in performance between the domains. Notably, as I had expected, neuroscience and bioinformatics would have some overlaps which would cause challenges to the model which can be inferred from the appended visualization aids.

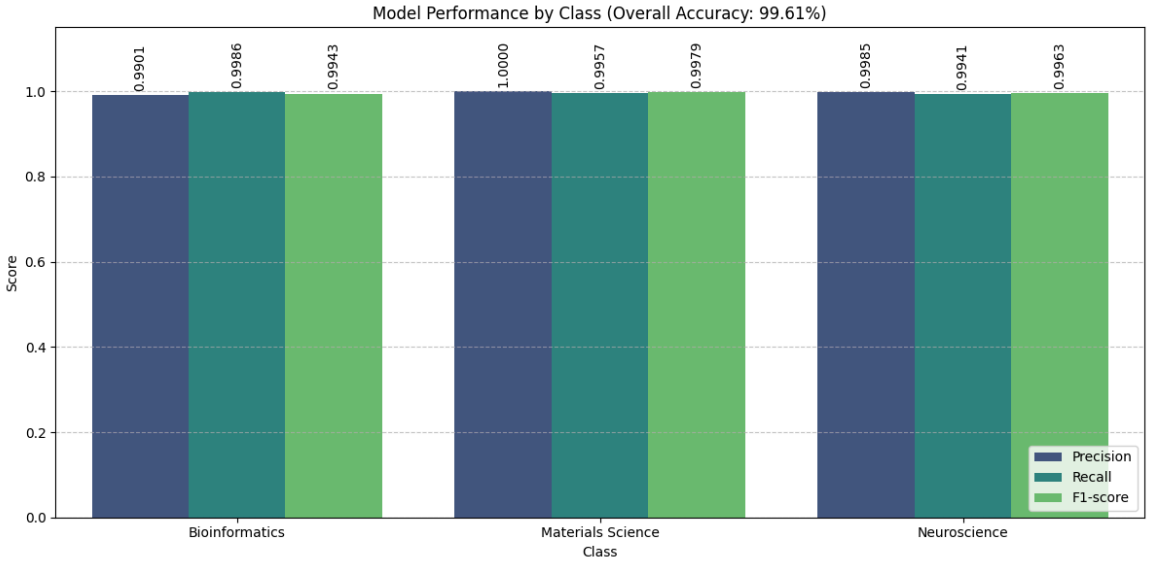

Figure 5. Classification performance metrics

5.2. Confusion Matrix

The confusion matrix generated from the cross-validation predictions showed the tendencies of the model when classifying. The confusion matrix results did reaffirm that there are still some challenges faced in classifying bioinformatics and neuroscience as they are both slightly overlapping at times and can be easy to be mistaken. Positively, they are both symmetrical in misidentification so the model has a balanced understanding of both domains.

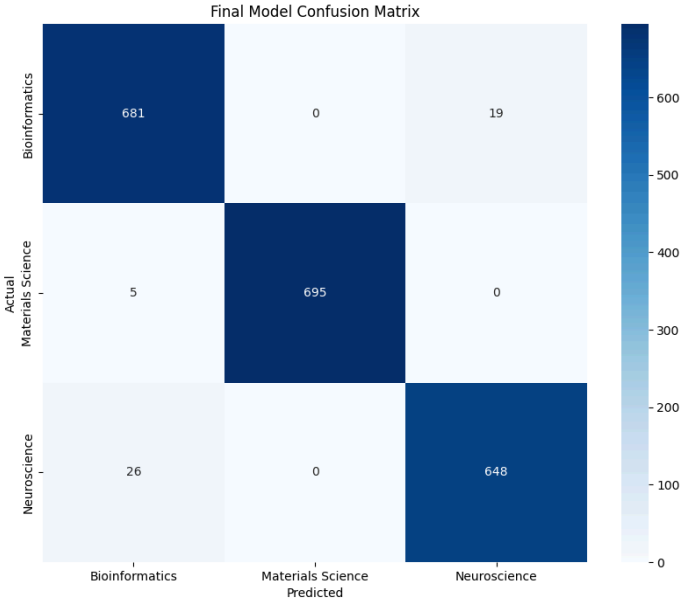

Figure 6. Confusion matrix for all domains

5.3. Embedding Space

The t-SNE visualization of the embedding space showed strong evidence that the contrastive learning approach was able to create strong correlations and representations of domains for the scientific abstracts. All clusters showed clear separation with neuroscience and bioinformatics sharing occasional misclassifications.

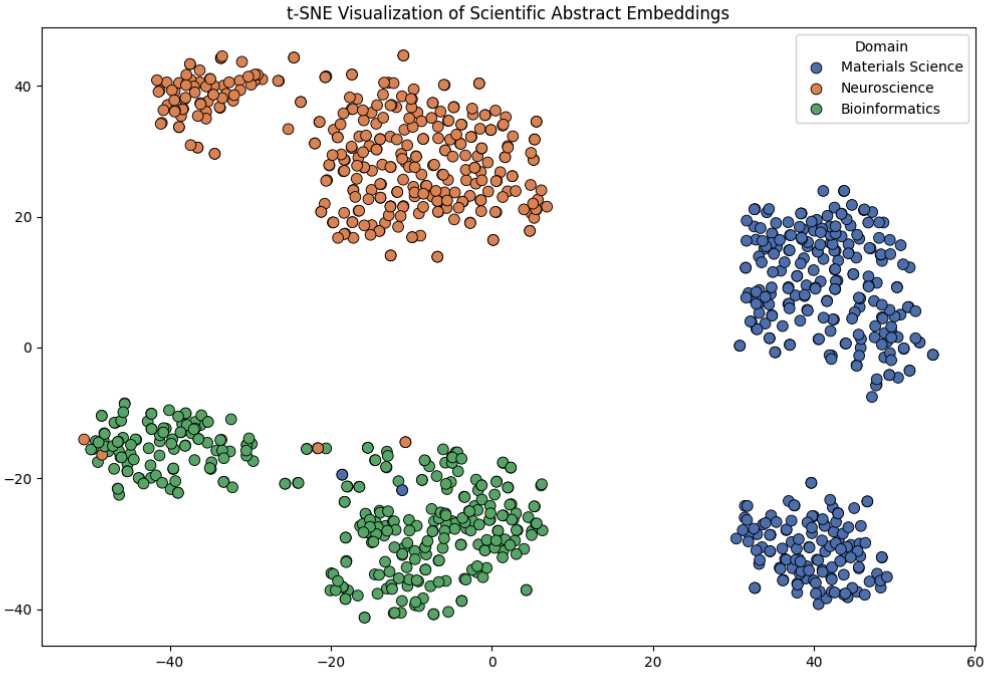

Figure 7. t-SNE visualization of embedding space

5.4. Model Comparisons

To further validate the impact of the SetFit approach with SciBERT, comparative experiments were conducted prior to benchmark models and performance.

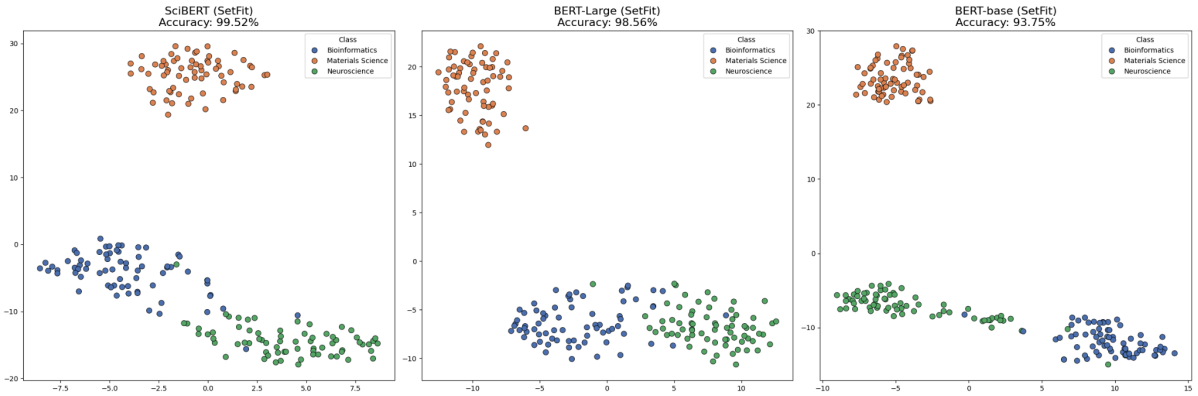

Figure 8. Comparison of different base models using SetFit framework

Figure 8 showed SciBERT's overwhelming dominance in SetFit fine-tuning compared to larger general task models such as BERT-Large and BERT-base which shows how relevant domain-specificity is for scientific text classification.

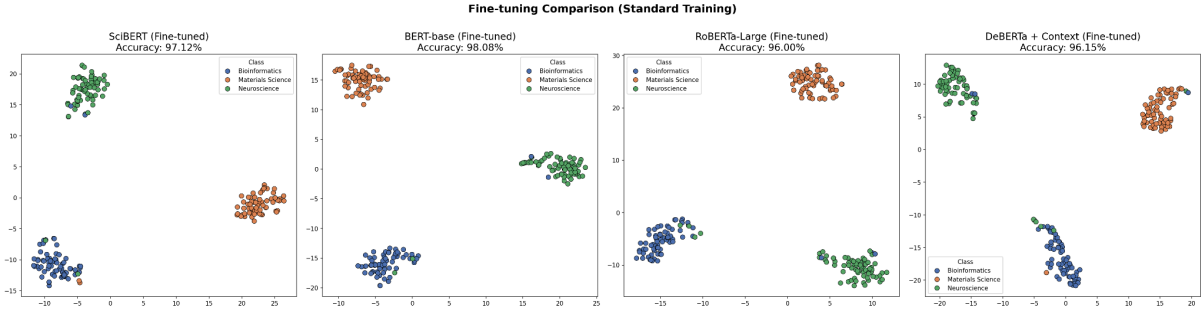

Figure 9. Comparison of different base models using standard fine-tuning

Figure 9 shows a second batch of comparison for model performances on standard fine-tuning. Fascinatingly, large models such as BERT-base, RoBERTa-Large were able to outperform SciBERT in classification despite the domain advantage. It can be inferred from Figure 8 and 9 that contrastive learning provided SciBERT the ability to form tighter cluster separations, allowing it to correlate words to domains better than BERT-base and BERT-large.

6. Conclusion

6.1. Summary

The project was able to successfully showcase the efficacy of the SetFit framework with contrastive learning. Through using SciBERT as the base model, implementing a novel context injection method, and contrastive fine-tuning, I was able to produce a scientific

abstract classification model of really high accuracy. The implementation incorporated 5-fold cross validation and two independent training runs to reduce the likelihood of error or bias. Context injection augmentation proved to be effective in extending the vocabulary window of a model, especially for models such as BERT-base or BERT-large. Overall, the project validates that SetFit is a valid solution to achieving competitive scientific text classification model accuracy with limited computational resources and data as opposed to traditional fine-tuning approaches.

6.2. Future Works

Future works could look into expanding the domain selection to include medicine, chemistry, physics, and computer science or other fields that may overlap one another as a challenge to solve for the future. Further expansion or investigation on advanced augmentation techniques may also provide a way to bridge the bottleneck experienced by researchers and scholars. Additionally, model compression techniques including knowledge distillation and quantization remain interesting pathways to allow deployment in circumstances with resource limitations.

7. References

Beltagy, I., Lo, K., & Cohan, A. (2019). SciBERT: A Pretrained Language Model for Scientific Text. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 3615-3620.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 4171-4186.

Miller, G. A. (1995). WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11), 39-41.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830.

Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 3982-3992.

Salton, G., & Buckley, C. (1988). Term-weighting Approaches in Automatic Text Retrieval. *Information Processing & Management*, 24(5), 513-523.

Tunstall, L., Reimers, N., Jo, U. E. S., Bates, L., Korat, D., Wasserblat, M., & Pereg, O. (2022). Efficient Few-Shot Learning Without Prompts. *arXiv preprint arXiv:2209.11055*.

Van der Maaten, L., & Hinton, G. (2008). Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(11), 2579-2605.

Singh, A., D'Arcy, M., Cohan, A., Downey, D., & Feldman, S. (2022). SPECTER 2.0: Generating high-quality document-level embedding of scientific papers.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is All You Need. *Advances in Neural Information Processing Systems 30 (NeurIPS)*, 5998-6008.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., & Rush, A. M. (2020). Transformers: State-of-the-Art Natural Language Processing. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 38-45.