UNIVERSITY OF LONDON
INTERNATIONAL PROGRAMMES

BSc Computer Science and Related Subjects

CM3070 PROJECT
FINAL PROJECT REPORT

Identifying Research Methodologies Used in Research

Author: Tan Hong An Alvin
Student Number : 230655164
Date of Submission: 18 September 2025
Github Link:
https://github.com/CatDogWoofMeow/UOL_FYP.git

Contents

CHAPTER 1: INTRODUCTION

(734/1000 words)

1.1 Project Concept and Motivation

This project explores the use of Natural Language Processing (NLP) models specifically, domain-tuned transformers such as SciBERT, RoBERTa and DistilBERT to automatically classify research articles within the computing discipline. The classification targets key metadata attributes such as discipline, subfield, and methodology. The broader goal is to reduce the manual effort involved in organising and analysing large volumes of academic literature. This is particularly beneficial in research-intensive contexts such as systematic reviews, curriculum development, and meta-research.

The motivation for this project stems from a noticeable gap in existing academic tools: while major digital libraries and repositories (e.g., arXiv, IEEE Xplore, ACM Digital Library) may offer basic categorisation of research articles, they often lack granular and domain-relevant tags. For example, users may not easily distinguish between for example, Design Science / System Design versus Theoretical / Conceptual, or track subfields such as Machine Learning versus Software Engineering. This lack of fine-grained classification limits the ability to analyse patterns in research methodology, evaluate domain trends, or support pedagogy. By leveraging domain-adapted models like SciBERT, this project aims to automate the identification of such nuanced classifications and make them scalable across large corpora of computing literature.

1.2 Research Aims and Questions

The main aim of this project is to develop a machine learning pipeline that can classify research articles into their respective discipline, subfield, and methodology using textual features extracted from PDF documents.

This leads to the following research questions:

1. Can SciBERT, a domain-specific NLP model, effectively classify computing papers by methodology, subfield, and discipline based on text excerpts such as titles, abstracts, and introductions?
2. What are the best practices for constructing and curating a high-quality dataset of academic papers for supervised classification tasks in the computing domain?
3. What kinds of methodological taxonomies are most suitable for categorizing research within computer science and information systems?

These research questions will guide the development of a robust and context-aware classification system for computing literature. By exploring the effectiveness of domain-specific models like SciBERT, RoBERTa or DistilBERT establishing reliable data annotation practices, and grounding the work in appropriate methodological taxonomies, the project aims to contribute both a practical tool and a methodological framework. These outcomes not only support improved metadata tagging and research discovery but also provide foundational insights for future studies in automated academic content classification.

1.3 Objectives and Deliverables

The objectives of this project are aligned with its overarching aim to automate the classification of computing research articles based on discipline, subfield, and methodology. First, the project seeks to collect and pre-process a dataset of research papers within the computing domain. This involves extracting relevant metadata and textual content from scholarly documents to create a labelled dataset suitable for supervised learning.

Next, the project aims to define and apply a taxonomy appropriate for classifying methodologies in computing and information systems. This taxonomy provides the foundation for meaningful categorisation and will be informed by established research frameworks.

With the dataset and taxonomy in place, the project proceeds to develop a classification model using SciBERT, a domain-specific transformer-based model trained on scientific

literature. The implementation will be presented in a Google Colab and Jupyter Notebook containing the full model training pipeline and code.

Following development, the model's accuracy and reliability will be evaluated using standard classification metrics such as accuracy, precision, recall, and F1-score, but priority for macro F1 and per class recall (micro and macro for multi-label model). An evaluation report will document the performance and highlight insights from error analysis.

Finally, the project reflects on its broader implications and possible extensions. This includes discussion of how automated methodology classification can support meta-research, educational tools, or systematic reviews. The final report, accompanied by an appendix of labelled examples, serves as the comprehensive deliverable that encapsulates both the research and development components of this project.

1.4 Project Template Used

This project is based on **Project Idea 1** from the CM3060 Natural Language Programming module:
"Identifying research methodologies that are used in research in the computing disciplines."

The goal is to analyse published research in computing fields such as Computer Science, Information Systems, and Information Technology and automatically identify:

- The **discipline** (e.g., Computer Science),
- The **subfield** (e.g., Machine Learning),
- And the **methodology** used (e.g., Design Science / System Design, Theoretical / Conceptual, Mixed Methods, Qualitative).

This aligns with the broader purpose of making research more transparent by automating the extraction of methodological structure from academic literature.

CHAPTER 2: LITERATURE REVIEW

(1780/2500 words)

2.1 Problem Space and Demand for Automation

The growing scale of scholarly publishing has turned literature screening and tagging into a bottleneck for educators, researchers and evidence-synthesis teams. Quantitative studies from the medical evidence community illustrate the magnitude of the problem. Using PROSPERO registry records, Borah et al. (2017) estimate that a “typical” systematic review takes a mean of 67.3 weeks from protocol registration to publication, with a substantial share of effort concentrated in title/abstract screening and metadata management rather than deep synthesis. More recently, Scott et al. (2023) discusses the “two-week systematic review” methodology and, in doing so, underscore why the traditional pipeline (median of about 66 weeks) is untenable without automation. Although these studies sit outside computing, their lesson generalises: manual screening and fine-grained tagging do not scale.

Digital libraries partially help. Topic taxonomies (e.g., ACM CCS) provide stable anchors for discipline and subfield, but most repositories do not encode methodological stance the difference between, say, Design Science / System Design and Theoretical / Conceptual in a way that downstream users can query or analyse at scale. This gap motivates learning methods that infer discipline, subfield, and methodology directly from text, so that fine-grained labels can be produced where repositories expose only titles/abstracts or raw PDFs.

2.2 What is being Classified from Citation Intent to Methodology

Two strands of research bracket the classification of scholarly text that matters for this project. The first is citation-intent modelling: learning the rhetorical function of citations within a paper. Cohan et al. (2019) propose “structural scaffolds” a multitask approach that exploits section cues and local context and demonstrate large gains over prior work.

The key insight is that rhetorical signals embedded in scientific prose (hedging, contrastive markers, section structure) are learnable and help when labels are not purely topical.

The second strand targets methodology/paradigm labels at the document level. Schmidt et al. (2024) frame automated methodology classification for Information Systems (IS) as a design-science artefact aimed at reducing manual screening costs. Their system classifies whole papers into paradigms/methods and shows that supervised models can deliver practical utility within a discipline.

Read together, the strands suggest a path: treat methodology as a rhetorical signal (Cohan et al., 2019) but pursue explicit methodology labels with end-user value (Schmidt et al., 2024). The present project sits at this intersection, but with two differences. First, it focuses on abstract-centric inputs to stay compatible with how repositories expose text. Second, it targets portability beyond a single discipline by aligning labels with computing/IS practice while avoiding whole-document inputs that are expensive to parse.

2.3 Domain-Pretrained vs General Transformers

Representation drives performance on scientific prose. SciBERT (Beltagy, Lo and Cohan, 2019) is pretrained on 1.14 million full-text scientific articles with a domain-specific vocabulary; across scientific NLP tasks it consistently outperforms general BERT. The advantage is not surprising: domain pretraining captures register, discourse and terminology characteristic of scholarly writing the very cues that often carry methodological meaning. In contrast, RoBERTa (Liu et al., 2019) showcases robust general-domain pretraining (longer training, more data, dynamic masking) and sets strong baselines across many tasks, while DistilBERT (Sanh et al., 2019) provides a distilled, faster model with modest average performance degradation.

For this project, the comparative expectation is clear. When labels are topic-anchored and frequent, general models or even lexical baselines (TF-IDF + Logistic Regression) can be competitive. When labels depend on rhetorical style and sparsely expressed method cues as with Mixed Methods and Qualitative SciBERT is more likely to surface minority

signal without bespoke feature engineering. Hence SciBERT serves as the primary backbone, with RoBERTa and DistilBERT as informative comparators for robustness and cost-efficiency.

2.4 Dataset, Annotation, and Taxonomy Design

The usefulness of any classifier is bounded both by the labels it predicts and by where it reads from in the document. Prior work illustrates the trade-offs. SciCite advances citation-intent classification through careful annotation and by showing that structural context matters, yet it carries no labels for methodology, discipline, or subfield (Cohan et al., 2019). By contrast, Schmidt et al. (2024) construct an IS-specific corpus for methodology/paradigm classification at whole-paper granularity; this increases conceptual coverage but raises annotation cost and reduces portability across venues. My approach blends these perspectives. I standardise abstract-centric extraction from PDFs, normalise the text, and build a corpus labelled at three levels, discipline, subfield, and methodology so the model can learn rhetorical cues while still producing outputs that have immediate value for meta-research and pedagogy. Choosing abstracts as the main signal is pragmatic: repositories expose them reliably, inference remains lightweight, and I avoid the noise introduced by tables, figures, and layout artefacts. It also acknowledges the practical limits of full-text processing at scale, where section detection, reference parsing, and formatting errors can degrade supervision quality and confound evaluation.

A classifier is only as useful as the taxonomy it predicts. For methodology, I adopt a theory-aware scheme grounded in Gregor et al. (2006), who distinguish analysis, explanation, prediction, explanation-and-prediction, and design/action as modes of theory in IS. These distinctions map naturally to a tractable four-way operationalisation suitable for computing/IS abstracts: Design Science / System Design (DS/SD) aligns with design/action; Theoretical / Conceptual (TC) aligns with explanation/theory; and Mixed Methods (MM) and Qualitative reflect widely used empirical stances in CS/IS venues. For discipline and subfield, I rely on the ACM Computing Classification System as a robust, hierarchical structure that downstream users already recognise. Together, these choices couple interpretability for readers (labels that reflect scholarly practice) with

predictability for models (label sets that are coherent, learnable, and portable). In effect, the corpus design ties an abstract-first reading strategy to a theory-informed yet operational taxonomy, striking a balance between conceptual fidelity and the practical constraints of scalable annotation and deployment.

2.5 Multi-Label Learning and Evaluation

Scholarly tagging is naturally multi-label: a paper belongs to one discipline and subfield and may carry multiple method cues. The predominant architectural choice is a single encoder with a sigmoid output per label and BCEWithLogitsLoss as the objective. Evaluation needs to report both micro-F1 (pooling decisions, thus reflecting prevalence) and macro-F1 (equal weight per label) so that minority labels remain visible. As Gibaja et al. (2015) note in their tutorial, threshold selection is a practical determinant of performance: fixed thresholds (e.g., 0.5) are simple but rarely optimal across labels; per-label thresholds guided by validation PR curves typically improve macro-F1 without retraining. In deployment, calibration (e.g., temperature scaling) further improves the utility of confidence scores for end users, especially when outputs populate confidence tables in an interface.

BERTMeSH (You et al., 2020) shows that large-scale multi-label tagging on scholarly text is tractable when a robust encoder is coupled with an appropriate head. Although MeSH indexing is a different task, its success with full-text inputs and a multi-label objective provides a technical precedent for predicting many labels including fine-grained subfields within one model.

2.6 Data Ingestion Abstract-Centric Pipelines and PDF Structure

Working directly from PDFs introduces failure points that can propagate into labels and undermine training. Systems such as CERMINE (Tkaczyk et al., 2015) extract titles,

authors, references, and sections reliably enough to power digital-library pipelines, yet they also expose the fragility of full-text workflows: section detection, multi-column layouts, and figure or table boundaries remain common sources of error. I therefore constrain ingestion to abstracts (with titles and, where available, short introductions). This choice trades some context for markedly greater reliability and portability across corpora, keeps inputs aligned with repository metadata, and reduces layout noise that would otherwise contaminate supervision and evaluation.

The methodological label space is long-tailed Design Science / System Design is common, while Qualitative is rare so the learning problem is as much about imbalance as it is about representation. The literature suggests two principal levers. Reweighting the loss by inverse frequency amplifies gradients for rare classes; it is simple and often effective, but if applied naively can over-emphasise outliers and erode majority performance. Resampling, for example via weighted random sampling with a tempered exponent, alters batch composition so the model encounters more minority examples while limiting degeneracy in mini-batches.

In this project I apply these levers conservatively. Class weights and tempered sampling are used in targeted runs to raise Mixed Methods and Qualitative recall, while warm-up and early stopping stabilise optimisation when the batch distribution is perturbed. Throughout, macro-F1 remains the primary selection metric to protect minority performance and to ensure that any gains on rare labels are not achieved by silently collapsing the majority class. The net effect is a pipeline that privileges reliable abstract-level signals and pairs them with measured imbalance remedies, yielding improvements that are reproducible and robust rather than brittle.

2.7 What the Literature Implies for Design

The reviewed work justifies three design decisions that permeate the system.

First, domain-pretrained encoders over lexical or general models. SciBERT (Beltagy, Lo and Cohan, 2019) is the most defensible base for methodology detection from abstracts because it encodes the rhetorical texture of scientific writing better than general encoders. RoBERTa (Liu et al., 2019) is a useful comparator for breadth and robustness, and DistilBERT (Sanh et al., 2019) probes the cost–accuracy frontier; however, both lack in-domain pretraining, and prior evidence suggests their minority-class performance will lag unless the learning signal is explicitly rebalanced.

Second, abstract-centric, multi-label inference with explicit thresholds. Full-text multi-label systems (You et al., 2020) show what is possible with rich inputs, but abstract-centric pipelines are more portable and robust to PDF idiosyncrasies (Tkaczyk et al., 2015). Given the sparsity of positive labels per instance, macro-F1 and per-label recall must complement accuracy. Following Gibaja et al. (2015), per-label thresholds should be treated as tunable design parameters rather than fixed at 0.5; calibration improves downstream confidence displays.

Third, tempered imbalance controls under compute constraints. Resource-limited settings (e.g., Colab GPUs) favour lighter controls that do not require extensive sweeps. In practice, tempered sampling plus short warm-up offers a good balance: it raises MM/Qualitative recall without catastrophically eroding DS/SD, and it respects time/credit limits highlighted by the broader evidence-synthesis literature (Borah et al., 2017; Scott et al., 2023) on why automation is needed in the first place.

Overall gap. Citation-intent studies prove that rhetorical roles are learnable (Cohan et al., 2019); discipline-specific prototypes show real utility (Schmidt et al., 2024); domain-pretrained encoders outperform general ones on scientific text (Beltagy, Lo and Cohan, 2019); multi-label tagging is feasible at scale (You et al., 2020); and PDF metadata extraction is mature but imperfect (Tkaczyk et al., 2015). What remains underexplored is a portable, abstract-centric, methodology-aware classifier tailored to computing/IS that (i) uses a domain-pretrained encoder, (ii) treats thresholds/calibration as first-class design parameters, and (iii) employs tempered imbalance controls that respect realistic compute budgets. This project is designed to fill that gap.

CHAPTER 3: PROJECT DESIGN

(1984/2000 words)

3.1 Domain and Users

This project is situated within the domain of academic research analysis and classification, intersecting Natural Language Processing (NLP) and computing education. The system is designed to benefit multiple user groups engaged in scholarly communication and knowledge organisation. First, it serves computing students who are learning about different research methodologies and need to search for example papers classified by type. Educators and academic supervisors can also benefit by using the system to guide students through structured literature reviews or to compile curated reading lists based on methodological focus. Additionally, academic librarians and repository curators may leverage the system to enhance metadata tagging within institutional or public digital archives. Lastly, researchers in Computer Science and Information Systems, particularly those conducting systematic reviews, meta-analyses, or disciplinary mapping, can use the tool to streamline the identification and categorisation of relevant publications. By improving the granularity and automation of methodological classification, the project directly contributes to more efficient, structured, and meaningful access to academic content.

3.2 Justification of Design Choices

Two constraints shape the design: (1) PDFs are messy; (2) GPU time is limited. I therefore make three deliberate choices.

Abstract-centric inputs. I read primarily from titles and abstracts (short introductions if available). Abstracts are reliably exposed by repositories and contain much of the rhetorical signal that indicates methodology. This avoids the brittle failure modes of full-text pipelines (multi-column layout, figure/table bleed, section mis-detection) while keeping inference fast.

Theory-aware but operational taxonomies. For methodology I use a four-way scheme Design Science / System Design (DS/SD), Theoretical / Conceptual (TC), Mixed Methods (MM), Qualitative aligned to IS theory but pragmatic for learning on abstracts. Discipline and subfield follow the ACM Computing Classification hierarchy so outputs match labels that end-users already recognise.

Domain-pretrained encoders with pragmatic tuning. I adopt SciBERT as the reference backbone after establishing (in Chapter 5) that it offers the best macro-F1 and the only non-zero Qualitative recall under matched conditions. Extended training interventions (class-weighted loss, a tempered WeightedRandomSampler with α=0.5 and caps, short warm-up, early stopping) are used judiciously to lift minorities without collapsing the majority.

3.3 Project Structure

The system couples a compact ingestion pipeline with a reproducible training loop and explicit feedback channels to data curation and deployment.

Data flow and components. Users (students, lecturers, librarians, researchers) upload PDFs to the ingestion layer, which extracts the abstract (plus title and short introduction where available) and normalises text into two views: a with-stopwords view for transformer models and a no-stopwords view for TF-IDF baselines. The multi-label SciBERT service encodes the abstract and produces probabilities for discipline, subfield, and methodology; a default 0.5 decision threshold yields labels that are presented with confidences in the results UI. Misclassifications and low-confidence cases are logged and routed to a curation sprint, where ambiguous items are manually annotated and, if necessary, the taxonomy is versioned before new data enter the training set. Persistent resources are maintained as data stores: the corpus of PDFs, the labelled CSV, serialized model artifacts (model.pt, mlb.pkl, classes.npy), and metrics logs. A separate training pipeline consumes these stores and writes updated artifacts and metrics.

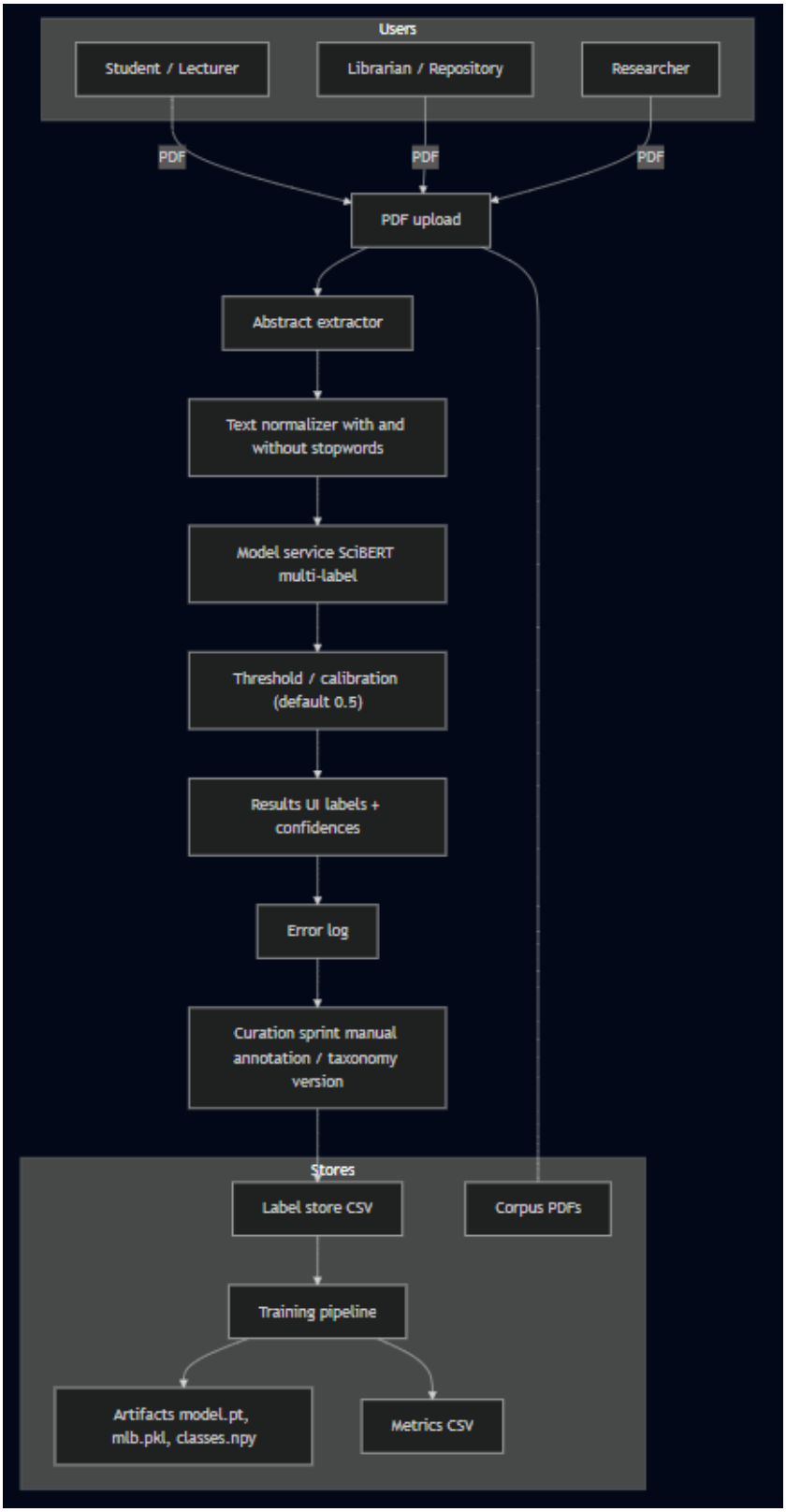

Figure 1. Data Flow

Learning loop and decision gate. Model development proceeds on a fixed, stratified 80/20 split (seed=42) so backbone comparisons are attributable to representation rather than partition variance. Training is controlled through a concise RunConfig that toggles sequence length (256/512), class-weighted loss, a tempered WeightedRandomSampler ($\alpha = 0.5$ with caps), short linear warm-up, gradient clipping, and early stopping by validation loss. After each run, validation computes macro-F1 and per-class recall. A decision gate accepts a configuration only if it (i) improves macro-F1 and (ii) increases minority-label recall while keeping DS/SD within a small tolerance ($\approx \leq 5$ percentage points drop relative to the baseline at the same sequence length). If the gate fails, the loop either adjusts training knobs (weights, sampler, warm-up, patience) or triggers a data action targeted growth for Mixed Methods/Qualitative or a taxonomy check. Accepted runs are checkpointed and logged to artifacts/metrics for reproducibility.

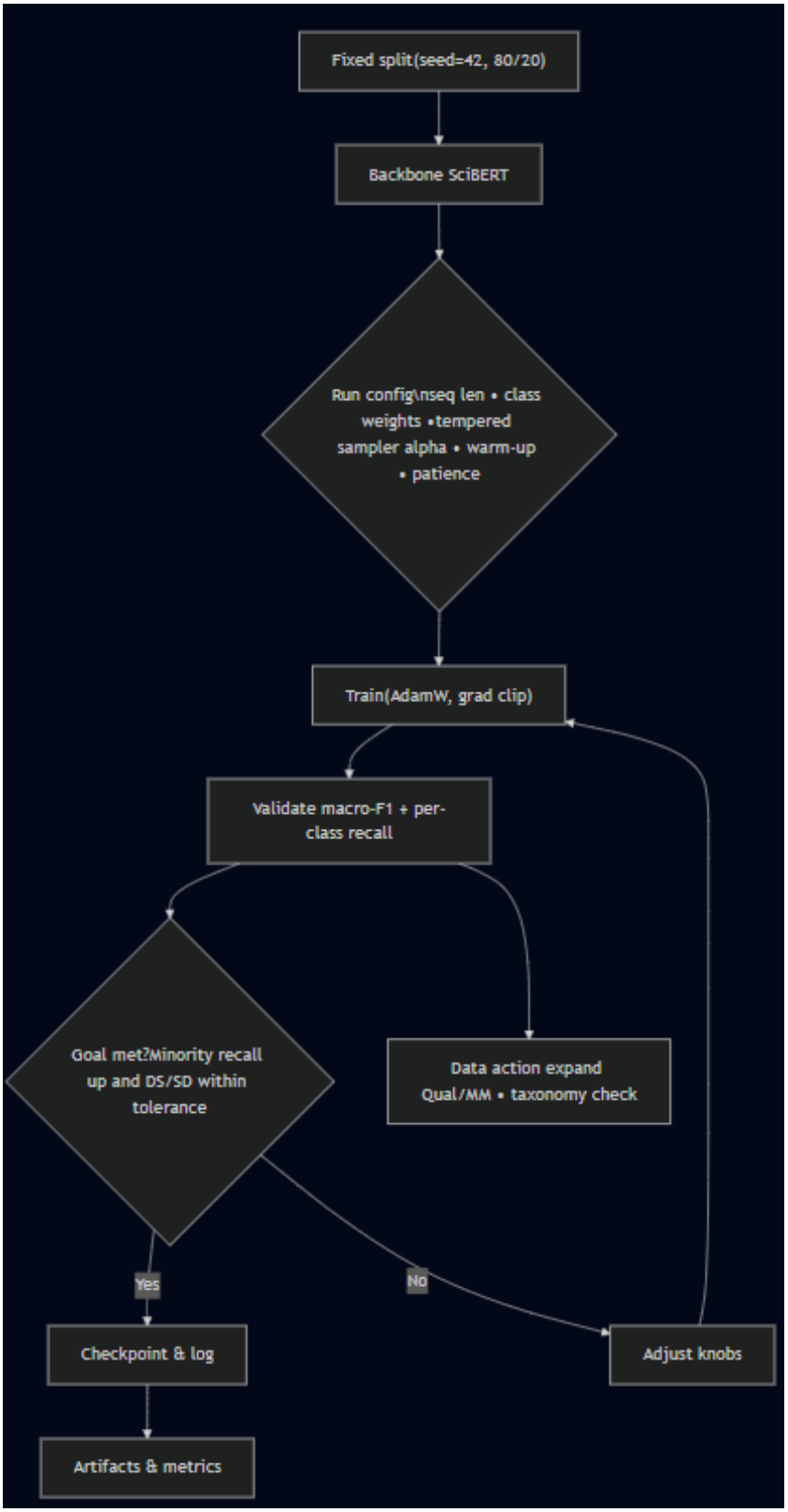

Figure 2. Training/ Evaluation Feedback Loop

Interfaces and reproducibility. Each stage exposes a simple contract: ingestion outputs normalised text; modelling consumes tensors and emits probabilities; the decision layer applies thresholds; evaluation writes metrics in a consistent schema; interpretability (LIME) surfaces token-level cues that inform the next curation sprint. All experiments use fixed seeds, a frozen split for deep models, and versioned label mappings so that results can be replicated and compared across milestones. The deployment path remains deliberately light-weight: a Colab notebook loads artifacts, accepts a PDF, and returns labels with a grouped confidence table, ensuring that the same design can be exercised by students, instructors, and repository staff without special infrastructure.

3.4 Key Technologies and Methods

The system is implemented in Python using PyTorch and Hugging Face Transformers for modelling, pdfplumber for robust abstract extraction from PDFs, spaCy and NLTK for normalisation, and scikit-learn for metrics and multi-label encoding. Ingestion is intentionally abstract-centric: the pipeline extracts the abstract (and title or short introduction when present) and produces two normalised views one with stopwords for transformer models, which preserves discourse cues learned during pretraining, and one without stopwords for TF-IDF baselines. Discipline and subfield are assigned via deterministic mappings; methodology follows a four-way scheme (DS/SD, TC, MM, Qual). To guarantee label order at training and inference, I persist the encoder state and label layout as mlb.pkl and classes.npy.

Modelling proceeds along three tracks. First, a transparent TF-IDF + Logistic Regression baseline establishes a variance-aware reference that is cheap to reproduce. Second, Model-1 transformer baselines at 512 tokens place SciBERT, RoBERTa, and DistilBERT under an identical recipe so architectural differences are not confounded by token budget. Third, the deployment model is a multi-label SciBERT with a linear head and BCEWithLogitsLoss, trained with AdamW, gradient clipping, and a short linear warm-up. Extended sweeps run at 256 tokens and toggle class-weighted loss and a tempered

WeightedRandomSampler ($\alpha = 0.5$ with caps) to reshape the learning signal efficiently under Colab constraints. All deep models reuse a frozen 80/20 stratified split (seed = 42) for comparability; runs write a single, consistent metrics row to CSV and save artifacts (model.pt, mlb.pkl, classes.npy) for reproducibility.

Interpretability is designed into the workflow. I use LIME to surface locally influential tokens on validation abstracts; these explanations are not treated as ground truth, but they inform data curation and guard against spurious shortcuts. Delivery is a Colab-friendly notebook that loads the serialized artifacts, performs abstract extraction and normalisation, and reports discipline, subfield, and methodology with a grouped confidence table. The default decision threshold is 0.5; per-label thresholding and calibration are reserved for later work without changing the model architecture.

Alternatives considered. Small ensembles (e.g., SciBERT + RoBERTa voting) could smooth idiosyncratic errors but double inference cost and complicate reproducibility in Colab. Text augmentation (synonym swaps, back-translation) risks distorting formal academic prose with uncertain gains on abstracts. Self-training/weak labelling is promising for growing minority classes but requires calibrated probabilities to avoid confirmation bias; I defer it until calibration is in place. Keeping the stack lean preserves maintainability and leaves room for these extensions without architectural changes.

3.5 Gantt Chart

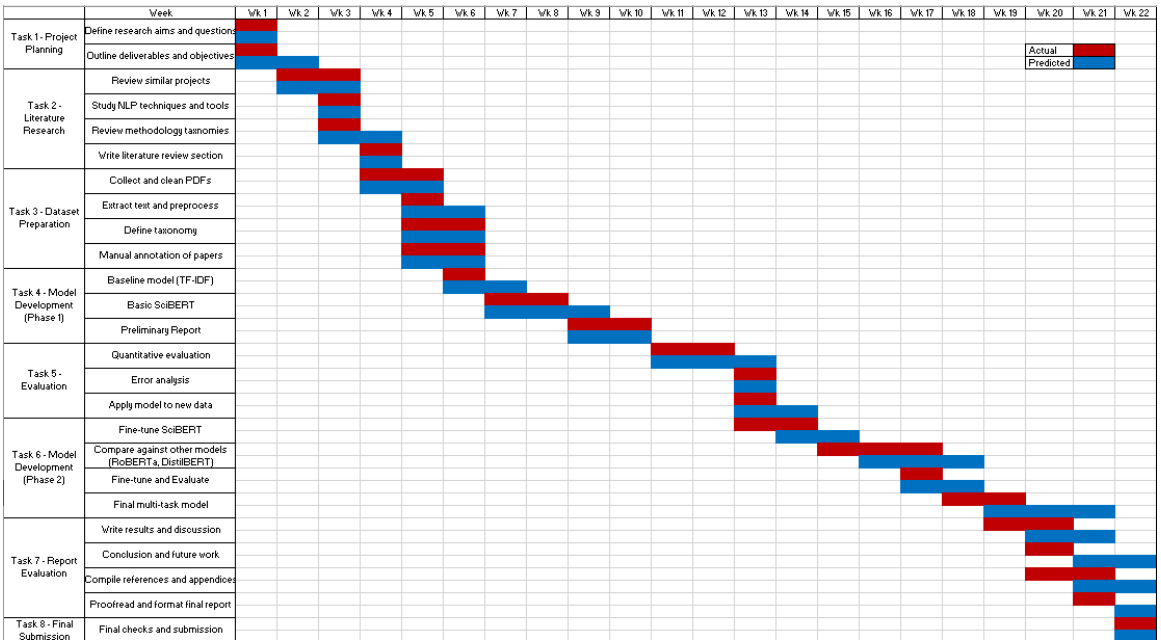

Figure 3. Gantt Chart

The Gantt chart contrasts the planned (blue) and actual (red) timelines across nine tasks from inception to submission. Rather than a simple sequence, the schedule is organised around a critical path Dataset Preparation → Model Development (Phase 1) → Evaluation → Model Development (Phase 2) → Report Integration with short, intentional overlaps to absorb variance from GPU availability and data curation. Tasks off the critical path (literature review, interpretability sprints, and figure production) are used to backfill idle time so that delays in training do not stall overall progress.

Tasks 1–2 (Project Planning and Literature Research) function as de-risking stages: they lock the objectives, taxonomy sources, and extraction approach while seeding the label design. Their overlap with early data collection is deliberate; freezing a first taxonomy while papers are being gathered shortens the feedback loop between design decisions and practical constraints. Task 3 (Dataset Preparation, Weeks 4–6) is a gating milestone: success is defined by a high-quality labelled CSV and reliable abstract extraction. This gate ensures that downstream model comparisons are attributable to representation rather than data noise.

Task 4 (Model Development, Phase 1) delivers the architectural decision under matched conditions. By running TF-IDF and 512-token transformer baselines on the same 80/20 stratified split, I select SciBERT on macro-F1 and minority-label recall. The report checkpoint at Week 10 is not merely administrative; it records the backbone decision and fixes the split so that subsequent results are comparable.

Task 5 (Evaluation, Weeks 11–14) and Task 6 (Model Development, Phase 2) are tightly coupled: analysis of per-class recall informs which training levers to activate (class weights, tempered sampling, warm-up) in the extended runs at 256 tokens. Here the schedule shows the only material variance: a ≈1-week slip driven by Colab GPU pre-emptions and quota limits. The mitigation shorter context, early stopping, and batching families contained the delay within the planned slack and preserved the critical path. Acceptance for Phase 2 is "macro-F1 improves while DS/SD recall remains within a small tolerance," which prevents minority gains from masking majority collapse.

Tasks 7 and 8 (Reporting, Integration, Submission) run in parallel with late experiments. Because artifacts and metrics are logged deterministically, figure regeneration is automated and does not depend on GPU access, allowing reporting to proceed even when training is queued. Overall, the close alignment of blue and red bars in the first half and the contained variance in Phase 2 indicate a schedule that is both feasible and evidence-driven: each milestone has a measurable acceptance criterion, and deviations are explained by resource constraints with clear corrective actions.

3.6 Testing and Evaluating Plan

Evaluation is integrated with the design so that each architectural choice has a corresponding measurement. I begin at the input boundary: abstract extraction is spot-checked on a stratified subset, and failures are logged by type (missing abstract, garbling, heading mis-detected). This establishes a realistic upper bound for an abstract-only system and prevents over-claiming downstream.

For transformer backbones, I adopt a fixed, stratified 80/20 hold-out (seed = 42). With deep models the binding constraint is compute rather than variance, so a frozen split makes backbone comparisons fair and reproducible while leaving enough validation mass to examine per-class recall on minority labels. By contrast, TF-IDF baselines are cheap to run, so I also report 5-fold cross-validation to provide variance-aware references.

Backbone selection is conducted under a matched 512-token recipe so that differences can be attributed to representation rather than token budget or optimiser quirks. The primary selector is macro-F1, which gives equal weight to all classes; per-class recall is inspected to ensure that gains are not achieved by silently collapsing performance on the majority (DS/SD). Extended experiments at 256 tokens then probe training-signal interventions class-weighted loss, tempered sampling, short warm-up, early stopping by comparing each configuration to its family baseline (C1/D1/E1). I apply an explicit do-no-harm gate: a run is accepted only if macro-F1 improves and DS/SD recall remains within a small tolerance (about five percentage points) of the corresponding baseline; otherwise, I either adjust the knobs or trigger a data action (targeted growth for Mixed Methods/Qualitative or a taxonomy check).

The multi-label system is evaluated with metrics aligned to its usage: element-wise accuracy, subset (exact-match) accuracy, and micro/macro-F1, reported both overall and by family (discipline, subfield, methodology). Where feasible, I derive per-label precision–recall curves to recommend label-specific thresholds typically lower for Qualitative and sometimes higher for DS/SD and I prepare for temperature scaling so confidences shown in the PDF tool are better calibrated. Interpretability remains part of the evaluation loop: after each milestone, LIME runs on validation abstracts to verify that predictions rely on plausible cues; the resulting token summaries and error patterns seed the next annotation sprint. Throughout, metrics and artifacts are written to disk with consistent schemas and tied to a taxonomy version and seed, producing an auditable trail that supports replication and figure generation in later chapters.

CHAPTER 4: IMPLEMENTATION

(1713/2500 words)

This chapter details the system I implemented to classify research papers by discipline, subfield, and methodology. The implementation proceeds in stages: (i) traditional baselines (TF-IDF + Logistic Regression), (ii) transformer baselines (SciBERT, RoBERTa, DistilBERT) under a matched training recipe, (iii) variants of transformer baselines that is modified for training enhancements, (iv) a multi-label SciBERT that predicts all three label families jointly, and (v) an end-to-end PDF pipeline that loads a trained model and produces human-readable predictions. All models are trained with reproducible seeds, the same train/validation split per task, and consistent logging for later comparison.

4.1 Data, Labels and Preprocessing

All experiments start from a single CSV uploaded in Colab. Rows with missing text_excerpt are dropped. Two normalised text views are created with spaCy (tokenisation + lemmatisation) and NLTK (stopwords):

- processed_with_stopwords: lowercased, punctuation stripped, lemmatised, stopwords kept. This preserves function words and discourse markers that transformer pretraining can usefully exploit.
- processed_no_stopwords: identical normalisation but stopwords removed. This suits sparse lexical models whose weighting (TF-IDF) already accounts for term frequency.

A small exploratory notebook done in Jupyter Notebook measured token lengths after WordPiece/BPE tokenisation. The distribution shows that 256 tokens comfortably cover most abstracts, while 512 eliminates truncation even for long ones. I therefore use 512 for the “Model 1” transformer baselines (to isolate architecture effects) and 256 for the extended variants (to study optimisation choices efficiently).

```
Count: 5582
count    5582.000000
mean      166.563776
std        64.506522
min        11.000000
50%       166.000000
75%       213.000000
90%       254.900000
95%       275.000000
99%       303.000000
max       596.000000
dtype: float64

% > 256 tokens: 9.48%
% > 512 tokens: 0.02%
```

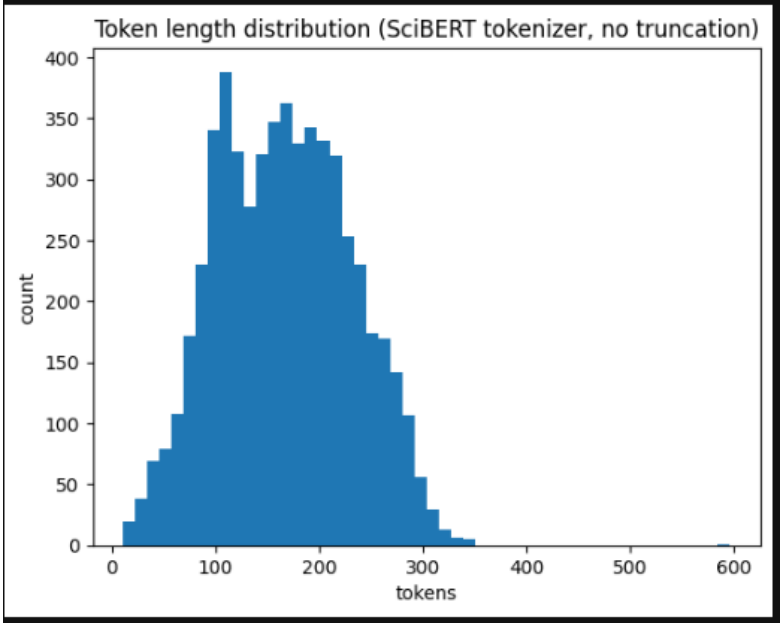

Figure 4. Token length distribution analysis

	processed_no_stopwords	processed_with_stopwords
0	report magnitude flash detect approximately mi...	we report on a magnitude flash detect for appr...
1	discuss feasibility gamma ray burst grb tev ga...	we discuss feasibility of gamma ray burst grb ...
2	dimension infiniterange kawasaki spin exchange...	in all dimension infiniterange kawasaki spin e...
3	technique develop explore complete space inter...	technique be develop for explore the complete ...
4	set general structure analysis frustrationfree...	we set up a general structure for the analysis...

Figure 5. Difference between processed and none processed stopwords

4.2 Baseline Architecture: TF-IDF and Logistic Regression

As a transparent reference point, I trained a sparse lexical baseline using TF-IDF features and a multinomial Logistic Regression classifier. Text was vectorised with TfidfVectorizer(max_features=3000) over the processed_with_stopwords field. The 3k cap keeps the model tractable on a modest dataset while still covering domain-specific terms that tend to drive lexical baselines. I kept stopwords for this baseline to align the feature space with the transformer experiments (which also ingest the with_stopwords view) and because preliminary checks showed that removing them did not materially change the ranking of models in this task.

Training followed two regimes. First, an 80/20 stratified hold-out split established a simple, repeatable benchmark on a fixed validation slice. Second, a stratified 5-fold cross-validation pass tightened the point estimate by averaging performance across multiple partitions of the data. In both settings I used LogisticRegression(max_iter=1000); the higher iteration limit merely guarantees convergence on the high-dimensional sparse matrix and does not otherwise change the objective.

To make later comparisons frictionless, the methodology run emits exactly the metrics used throughout this chapter, validation accuracy, macro-F1, and per-class recall, using classification_report. A small helper, save_methodology_csv(...), writes a single-row CSV with a fixed schema.

```
[Methodology - TFIDF Basic]
Accuracy: 0.756490599820949
                              precision    recall  f1-score   support

Design Science / System Design       0.77      0.93      0.84       721
                 Mixed Methods       0.80      0.04      0.08        90
                   Qualitative       0.00      0.00      0.00        11
      Theoretical / Conceptual       0.72      0.57      0.63       295

                      accuracy                           0.76      1117
                     macro avg       0.57      0.39      0.39      1117
                  weighted avg       0.75      0.76      0.72      1117

Methodology metrics saved to tfidf_basic_methodology_metrics.csv

[Discipline - TFIDF Basic]
Accuracy: 0.4816472694717995
                                        precision    recall  f1-score   support

                             Astrophysics       0.86      0.61      0.71        41
                         Computer Science       0.27      0.34      0.30       238
                                Economics       0.67      0.83      0.74       267
Electrical Engineering and Systems Science       0.00      0.00      0.00        28
                              Mathematics       0.45      0.60      0.51       209
                                  Physics       0.34      0.08      0.14       119
                    Quantitative Biology       0.55      0.39      0.46        99
                              Statistics       0.44      0.31      0.36       116

                                accuracy                           0.48      1117
                               macro avg       0.45      0.40      0.40      1117
                            weighted avg       0.46      0.48      0.46      1117
```

Figure 6. TF-IDF Basic report

```
[Subfield - TFIDF Basic]
Accuracy: 0.5201432408236347
                                precision    recall  f1-score   support

                Algebraic Geometry       0.51      0.77      0.61       193
Computer Science and Game Theory       0.00      0.00      0.00        42
              Galaxy Astrophysics       0.75      0.50      0.60        36
               General Economics       0.63      0.65      0.64       370
      Image and Video Processing       0.00      0.00      0.00        18
                Machine Learning       0.40      0.55      0.46       275
   Neural and Cognitive Modeling       0.56      0.21      0.30       107
                 Quantum Physics       0.14      0.01      0.02        76

                        accuracy                           0.52      1117
                       macro avg       0.37      0.34      0.33      1117
                    weighted avg       0.48      0.52      0.48      1117
```

Figure 6.1. TF-IDF Basic report

```
[Methodology - TFIDF K-Fold]
Cross-validated accuracy (mean): 0.7586887881325748
                              precision    recall  f1-score   support

Design Science / System Design       0.77      0.93      0.84      3601
                Mixed Methods       0.80      0.02      0.03       450
                  Qualitative       0.00      0.00      0.00        54
      Theoretical / Conceptual       0.71      0.59      0.65      1477

                     accuracy                           0.76      5582
                    macro avg       0.57      0.39      0.38      5582
                 weighted avg       0.75      0.76      0.72      5582

Methodology metrics saved to tfidf_kfold_methodology_metrics.csv

[Discipline - TFIDF K-Fold]
Cross-validated accuracy (mean): 0.4704376482064413
                                         precision    recall  f1-score   support

                             Astrophysics       0.79      0.55      0.65       209
                         Computer Science       0.30      0.35      0.32      1259
                                Economics       0.64      0.86      0.73      1263
Electrical Engineering and Systems Science       0.20      0.01      0.01       168
                              Mathematics       0.44      0.56      0.49      1101
                                  Physics       0.19      0.04      0.07       580
                     Quantitative Biology       0.55      0.42      0.48       456
                               Statistics       0.41      0.29      0.34       546

                                 accuracy                           0.47      5582
                                macro avg       0.44      0.38      0.39      5582
                             weighted avg       0.44      0.47      0.44      5582


[Subfield - TFIDF K-Fold]
Cross-validated accuracy (mean): 0.5231065674505764
                               precision    recall  f1-score   support

                Algebraic Geometry       0.52      0.79      0.63      1005
Computer Science and Game Theory       0.00      0.00      0.00       273
              Galaxy Astrophysics       0.73      0.49      0.59       189
               General Economics       0.61      0.73      0.66      1714
      Image and Video Processing       0.00      0.00      0.00       106
                Machine Learning       0.42      0.48      0.45      1424
   Neural and Cognitive Modeling       0.44      0.16      0.24       486
                Quantum Physics       0.18      0.02      0.04       385

                       accuracy                           0.52      5582
                      macro avg       0.36      0.34      0.33      5582
                   weighted avg       0.46      0.52      0.48      5582
```

Figure 7. TF-IDF KFold report

	run	val_acc	macro_f1	recall_Design_Science_/_System_Design	recall_Mixed_Methods	recall_Qualitative	recall_Theoretical_/_Conceptual
0	TFIDF_basic	0.756491	0.390160	0.934813	0.044444	0.0	0.566102
1	TFIDF_kfold	0.758689	0.381373	0.931408	0.017778	0.0	0.591063

Figure 8. TF-IDF models comparisons

4.3 Transformer Baseline

To make the backbone comparison about representation, I fine-tuned three encoders: SciBERT (allenai/scibert_scivocab_uncased), RoBERTa (roberta-base), and DistilBERT (distilbert-base-uncased) under a single, matched recipe. Every model uses the same 80/20 stratified split (seed = 42) and the same text view (processed_with_stopwords) so that function words and discourse markers remain available to the encoder; this mirror how BERT-family models were pretrained and avoids silently handicapping one model with a different input distribution. I set the sequence length to 512 tokens for these baselines to remove truncation as a confound in the architectural comparison long abstracts and multi-clause methodological statements are common in scientific prose, and shorter windows can mix representation differences with context differences. Optimisation follows the stable, widely used configuration for BERT-like models: AdamW with a 2e-5 learning rate, batch size 4, and gradient clipping at 1.0. Training runs for three epochs, and I keep the checkpoint with the best validation accuracy; accuracy is a low-variance selector on a small validation split, while the analysis still relies on macro-F1 and per-class recall, which are computed and logged for each run.

I implemented a compact PyTorch loop rather than relying on the high-level Trainer. This gives precise control over the loss surface, early-stopping/selection criteria, and later makes the transition to multi-label straightforward (where the objective and decision rule differ). It was also more memory-stable in Colab. In practice the code maps the Hugging Face tokenizer with padding="max_length" and truncation=True, sets the tensor columns (input_ids, attention_mask, and label) on a datasets object, and iterates the usual forward/backward/step cycle with clipping. After each baseline run, a single-row CSV is written using the exact schema established in the TF-IDF stage run, val_acc, macro_f1, and the four methodology recalls so later comparison notebooks can concatenate results across very different models without column wrangling.

```
Epoch 01 | train_loss: 0.7739 | val_loss: 0.6848 | val_acc: 0.7905
Epoch 02 | train_loss: 0.6327 | val_loss: 0.8851 | val_acc: 0.7995
Epoch 03 | train_loss: 0.5445 | val_loss: 0.7035 | val_acc: 0.8201
```

Figure 9. SciBERT Model 1

```
Validation accuracy: 0.8200537153088631
                              precision    recall  f1-score   support

Design Science / System Design       0.84      0.95      0.89       685
                 Mixed Methods       0.63      0.35      0.45        94
                   Qualitative       0.75      0.17      0.27        18
       Theoretical / Conceptual      0.80      0.72      0.76       320

                      accuracy                           0.82      1117
                     macro avg       0.76      0.55      0.59      1117
                  weighted avg       0.81      0.82      0.81      1117
```

Figure 9.1. SciBERT Model 1

```
Epoch 01 | train_loss: 0.8051 | val_loss: 0.8655 | val_acc: 0.7529
Epoch 02 | train_loss: 0.7374 | val_loss: 0.8184 | val_acc: 0.7932
Epoch 03 | train_loss: 0.6601 | val_loss: 0.7511 | val_acc: 0.8057
```

Figure 10. RoBERTa Model 1

```
Validation accuracy: 0.8057296329453895
                              precision    recall  f1-score   support

Design Science / System Design       0.86      0.90      0.88       685
                 Mixed Methods       0.49      0.48      0.48        94
                   Qualitative       0.00      0.00      0.00        18
       Theoretical / Conceptual      0.78      0.75      0.77       320

                      accuracy                           0.81      1117
                     macro avg       0.53      0.53      0.53      1117
                  weighted avg       0.79      0.81      0.80      1117
```

Figure 10.1. RoBERTa Model 1

```
Epoch 01 | train_loss: 0.7041 | val_loss: 0.7227 | val_acc: 0.7878
Epoch 02 | train_loss: 0.5476 | val_loss: 0.6577 | val_acc: 0.8201
Epoch 03 | train_loss: 0.4154 | val_loss: 0.6752 | val_acc: 0.8290
```

Figure 11. DistilBERT Model 1

```
Validation accuracy: 0.8290062667860341
                              precision    recall  f1-score   support

Design Science / System Design       0.92      0.88      0.90       685
                 Mixed Methods       0.56      0.56      0.56        94
                   Qualitative       0.00      0.00      0.00        18
      Theoretical / Conceptual       0.74      0.85      0.79       320

                      accuracy                           0.83      1117
                     macro avg       0.55      0.57      0.56      1117
                  weighted avg       0.82      0.83      0.82      1117
```

Figure 11.1. DistilBERT Model 1

4.4 Model Analysis and Interpretability with LIME

To understand why the baseline SciBERT makes particular methodology decisions and to guide subsequent engineering, I used LIME for text classification as a black-box explainer. LIME perturbs an input (masking or deleting tokens), queries the model on these variants, and fits a simple local surrogate that highlights which tokens most influenced the prediction around that instance.

Procedure. I loaded the best SciBERT Model 1 checkpoint in a Jupyter environment (Colab CPU struggled with the perturbation volume) and ran LIME on validation abstracts. For each case, I extracted the top-10 influential tokens (by absolute weight), then aggregated across many explanations per methodology label in two ways: (i) frequency, how often a token appears in the top-10 for that label; and (ii) cumulative importance, sum of its LIME weights. This produced per-label vocabularies that reflect what the model actually uses.

Design impact. These observations supported two concrete choices already reflected in the pipeline:

- Sequence length at 256 for extended runs most influential tokens reside well within the abstract; the token-length study plus LIME's locality pattern justified shortening without information loss.
- Where to target rebalancing. Since Qualitative signal is distributed and less keyword-anchored, I focused on training-signal changes (class weights, tempered

oversampling, warm-up) rather than chasing features; for Mixed Methods, mild rebalancing + stable scheduling was sufficient.

Auxiliary outputs. I also used the aggregated LIME vocabularies to prototype a lightweight rule-based heuristic (presence counts of top tokens per label) and to seed arXiv harvesting queries when expanding data. These helpers were not part of the final scoring pipeline but accelerated experimentation.

Limitations. LIME is local and perturbation-based; it does not capture cross-token interactions or long-range dependencies. I treated it as diagnostic evidence to guide engineering, not as a definitive explanation of model internals.

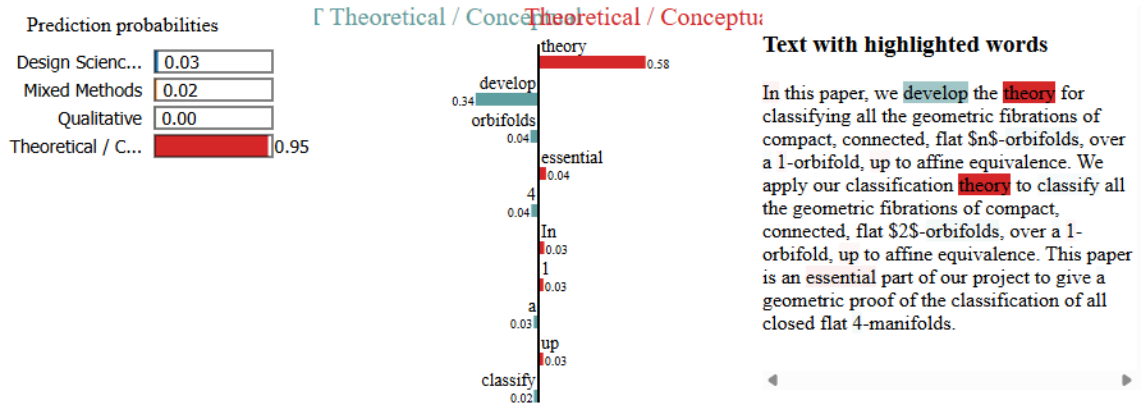

Figure 12. LIME tool top 10 tokens

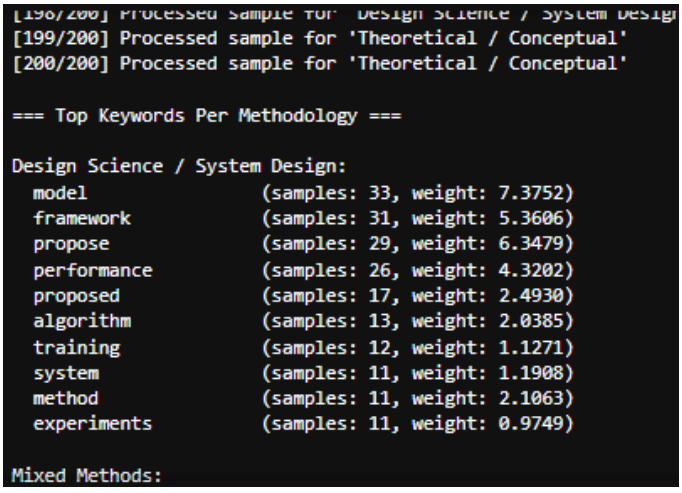

Figure 13. Cumulative weightage of tokens

4.5 Training Enhancements for Extended Models

Having fixed the backbone comparison, I then vary the optimisation bias at 256 tokens to study whether the training signal can be shaped to improve harder labels without destabilising the easy ones. The implementation centralises these toggles in a small experiment harness:

(a) Class-weighted loss.
CrossEntropyLoss (weight=…) uses weights proportional to N / (K · count_k) (renormalised to mean 1). This increases the gradient contribution of rarer classes without changing the architecture. In code, weights are computed from label counts on the training set and passed once to the loss constructor.

(b) WeightedRandomSampler with tempered oversampling.
For runs that alter what the model sees each epoch, a WeightedRandomSampler draws examples i.i.d. with probability ∝ count(label)^{-α} up to a small cap. The α exponent tempers the effect (α=0.5 oversamples minorities without making batches degenerate); the cap prevents a handful of rare examples from monopolising training. This sampler plugs directly into the DataLoader, leaving the model code unchanged.

(c) Label smoothing.
A small smoothing (e.g., 0.05 where supported) reduces over-confident updates by spreading a fraction of probability mass to non-targets. It is a lightweight regulariser that often pairs well with shorter context windows.

(d) Linear warm-up.
get_linear_schedule_with_warmup increases the learning rate gradually over the first 10% of steps. This avoids large, noisy updates while the classifier head is randomly initialised, particularly helpful when also altering batch composition.

(e) Early stopping by validation loss.
While the baselines use val-accuracy for checkpoint selection, the extended runs stop on

validation loss with patience. This suits schedules that temporarily trade accuracy for better calibration.

All of the above are exposed through a small RunConfig dataclass and a single run_once(cfg) function that: tokenises to cfg.max_len, builds loaders (with or without sampler), constructs the loss (with or without weights/smoothing), sets up warm-up if requested, trains with clipping, and records metrics. The three SciBERT runs (C1–C3), RoBERTa runs (D1–D3), and DistilBERT runs (E1–E3) are short, named configurations that flip exactly these switches.

This unified harness is the backbone of the comparative figures later on and makes the experiments repeatable in Colab despite fluctuating GPU quotas.

```
Map: 100% 4465/4465 [00:05<00:00, 832.88 examples/s]
Map: 100% 1117/1117 [00:01<00:00, 804.20 examples/s]
Some weights of BertForSequenceClassification were not initialized from the model checkpoint at allenai/scibert_scivocab_uncased and are newly initialized: ['classifier.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
[methodology | C1_baseline_len256] Epoch 01 | train_loss: 0.8500 | val_loss: 0.8427 | val_acc: 0.7672 | patience 0/2
[methodology | C1_baseline_len256] Epoch 02 | train_loss: 0.7877 | val_loss: 0.9658 | val_acc: 0.7816 | patience 0/2
[methodology | C1_baseline_len256] Epoch 03 | train_loss: 0.7272 | val_loss: 0.7590 | val_acc: 0.7959 | patience 1/2
[methodology | C1_baseline_len256] Epoch 04 | train_loss: 0.6747 | val_loss: 0.7916 | val_acc: 0.7959 | patience 0/2
[methodology | C1_baseline_len256] Epoch 05 | train_loss: 0.6213 | val_loss: 0.8990 | val_acc: 0.7842 | patience 1/2
[methodology | C1_baseline_len256] Early stopping.
Map: 100% 4465/4465 [00:02<00:00, 1683.07 examples/s]
Map: 100% 1117/1117 [00:00<00:00, 1672.88 examples/s]
Some weights of BertForSequenceClassification were not initialized from the model checkpoint at allenai/scibert_scivocab_uncased and are newly initialized: ['classifier.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
[methodology | C2_lossweights_len256] Epoch 01 | train_loss: 2.0755 | val_loss: 1.8565 | val_acc: 0.4557 | patience 0/2
[methodology | C2_lossweights_len256] Epoch 02 | train_loss: 1.8999 | val_loss: 2.0195 | val_acc: 0.2068 | patience 0/2
[methodology | C2_lossweights_len256] Epoch 03 | train_loss: 1.7851 | val_loss: 1.9005 | val_acc: 0.5577 | patience 1/2
[methodology | C2_lossweights_len256] Early stopping.
Map: 100% 4465/4465 [00:02<00:00, 1678.64 examples/s]
Map: 100% 1117/1117 [00:01<00:00, 1027.47 examples/s]
Some weights of BertForSequenceClassification were not initialized from the model checkpoint at allenai/scibert_scivocab_uncased and are newly initialized: ['classifier.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
[methodology | C3_combo_len256] Epoch 01 | train_loss: 1.6337 | val_loss: 1.9740 | val_acc: 0.2032 | patience 0/2
[methodology | C3_combo_len256] Epoch 02 | train_loss: 1.2531 | val_loss: 1.8831 | val_acc: 0.7278 | patience 0/2
[methodology | C3_combo_len256] Epoch 03 | train_loss: 1.0664 | val_loss: 1.9965 | val_acc: 0.7046 | patience 0/2
[methodology | C3_combo_len256] Epoch 04 | train_loss: 1.0050 | val_loss: 1.9554 | val_acc: 0.2954 | patience 1/2
[methodology | C3_combo_len256] Early stopping.
```

	run	max_len	lr	warmup	sampler	alpha	loss_weights	val_acc	macro_f1	val_loss	recall_Design_Science_-_System_Design	recall_Mixed_Methods	recall_Qualitative	recall_Theoretical_-_Conceptual
0	C3_combo_len256	256	0.000015	0.1	True	0.5	True	0.727842	0.554317	1.8831	0.773723	0.574468	0.444444	0.690625
1	C1_baseline_len256	256	0.000020	0.0	False	0.0	False	0.795882	0.517314	0.7590	0.908029	0.468085	0.000000	0.696875
2	C2_lossweights_len256	256	0.000020	0.0	False	0.0	True	0.455685	0.417329	1.8565	0.341606	0.457447	0.500000	0.696875

```
Saved summary to: sciBERT_methodology_3run_summary_len256.csv
```

Figure 14. SciBERT extended models

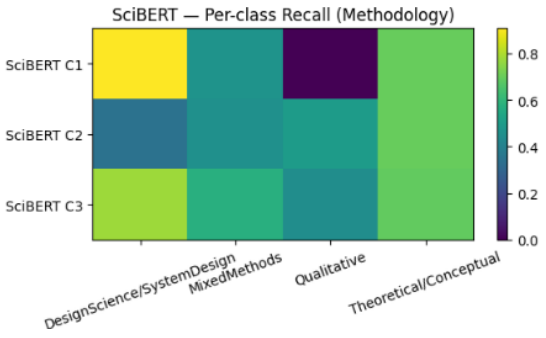

Figure 15. SciBert extended model heatmap

4.6 Multi-Label Inference with SciBERT

The final component replaces the single SoftMax head with a multi-label head and exposes a small PDF-inference loop suitable for Colab:

Architecture. SciBERT remains the encoder; the classifier is a linear layer with one logit per label across the union of discipline, subfield, and methodology classes. The objective is BCEWithLogitsLoss, which treats each label decision independently after a shared representation. Predictions are the elementwise sigmoid(logits) > threshold (0.5 in this implementation; per-class thresholds are easy to add later).

Label encoding. A MultiLabelBinarizer produces a stable label order (saved to classes.npy) and a pickled encoder (mlb.pkl) so inference uses the exact same layout as training.

Training loop. The loop mirrors the single-label version (seed control, tokenise processed_with_stopwords, batch size 4, gradient clipping). A short linear warm-up stabilises the first steps; early stopping monitors validation loss and restores the best state. Colab GPU is used when available; CPU remains viable for a handful of documents.

PDF inference. pdfplumber extracts the abstract by scanning for an “Abstract” heading and falling back to the first N characters if needed. The script then prints the three predicted families (discipline, subfield, methodology) and a grouped confidence table sorted by probability, which makes the model’s behaviour inspectable without a GUI.

This pipeline is intentionally minimal but production-oriented: artefacts (mlb.pkl, classes.npy, multi_label_scibert.pt) are uploaded explicitly; missing assets trigger an upload prompt; and the forward pass tolerates token_type_ids (None for SciBERT/RoBERTa) to avoid runtime surprises.

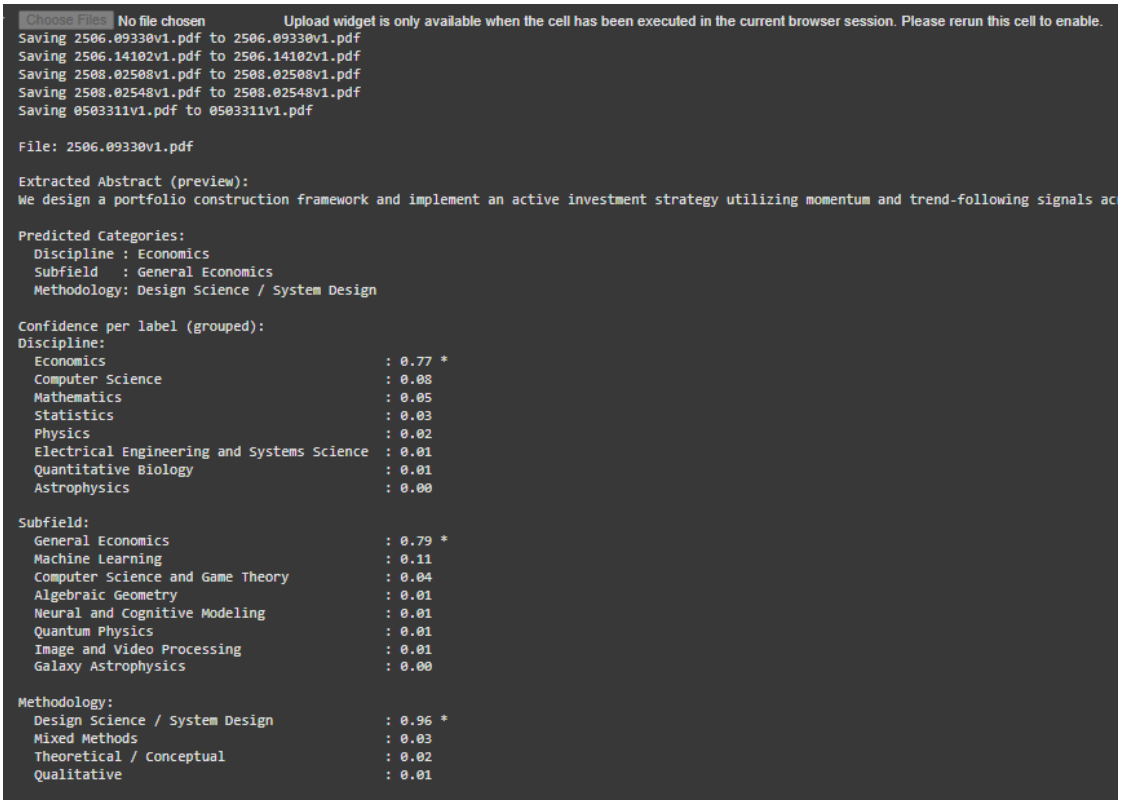

Figure 16. Full pipeline output

CHAPTER 5: EVALUATION

(2480/2500 words)

5.1 Evaluation Objectives

This chapter evaluates how well my system classifies research papers by methodology, and by extension how design choices affect performance on difficult, minority labels. I pursue four questions. First, do contextual encoders materially outperform a strong lexical baseline on the methodology task? Second, when training conditions are held constant, which backbone: SciBERT, RoBERTa, or DistilBERT offers the best balance across classes? Third, can training-signal interventions at a shorter context window shift recall toward minority labels (Mixed Methods and especially Qualitative) without collapsing majority performance (Design Science / System Design and Theoretical / Conceptual)? Finally, when the task is extended to multi-label prediction of discipline, subfield, and methodology jointly, do the single-label patterns persist?

All comparisons are made on the same 80/20 stratified split (seed = 42). For the "Model 1" baselines I fix the sequence length at 512 tokens to eliminate truncation as a confound in the backbone comparison; for the extended experiments I use 256 tokens to enable broader sweeps within Colab constraints. This separation keeps architectural conclusions clean while still letting me explore optimisation choices efficiently. I treat this as a testing-on-data evaluation (no user study) for a supervised text classification project.

5.2 Metrics and Rationale

I optimise and select models with the goal of balanced performance across methodology labels, not just headline accuracy. Accordingly, macro-F1 is my primary metric. Macro-F1 gives each label equal weight, which exposes failure on small or diffuse classes such as Qualitative that would otherwise be masked by majority-class dominance. I also report validation accuracy because it is a stable selector on a small validation set and helps sanity-check runs; however, accuracy is never the deciding score. To understand what changes when I alter a backbone or the training signal, I examine per-class recall for the four methodology labels: Design Science / System Design (DS/SD), Mixed Methods

(MM), Qualitative, and Theoretical / Conceptual (TC). Recall is the right lens for my objective: if the system cannot retrieve minority cases, it fails the use-case even if precision is high.

In the multi-label setting (discipline, subfield, methodology together), I report both macro-F1 (treating each label equally) and micro-F1 (pooling decisions across labels and samples, thus reflecting prevalence). I also cite exact-match (subset) accuracy to show end-to-end utility, but I interpret it carefully: because most labels are "off" for any given paper, exact match can look high even when minority labels underperform. All multi-label metrics are computed at a fixed 0.5 sigmoid threshold. This keeps the analysis internally consistent; where relevant I note how per-label thresholds or calibration could further improve the precision–recall trade-off.

5.3 Baselines vs Transformer (SciBERT)

I begin with a transparent lexical baseline TF-IDF features (3 000 max features) with multinomial Logistic Regression and compare it to the first transformer I introduce, SciBERT (Model 1), trained under the matched recipe at 512 tokens. The two TF-IDF variants behave similarly: the hold-out run achieves 0.756 accuracy with 0.390 macro-F1, while the 5-fold estimate lands at 0.759 / 0.381. Per-class recalls show why macro-F1 is low despite respectable accuracy: DS/SD recall is very high (at about 0.93), TC is moderate (at about 0.56–0.59), MM collapses (at about 0.04 and at about 0.02 respectively), and Qualitative is 0 in both settings. In short, a sparse lexical model can find majority DS/SD articles and a good share of TC, but it does not capture the diffuse cues that signal MM or Qualitative methodology in scientific prose.

Under identical data and split, SciBERT (Model 1) lifts performance across the board: 0.820 accuracy with 0.594 macro-F1, a +0.20 absolute macro-F1 gain over TF-IDF. The per-class picture explains the improvement. DS/SD stays high (0.946 recall), TC rises (0.725), MM jumps from about 0.02–0.04 to 0.351, and critically Qualitative is no longer a blind spot (0.167 recall). That last point matters disproportionately: the minority labels drive macro-F1, and they reflect the project's intent to classify methodological stance, not

just domain lexicon. The change from zero to non-zero Qualitative recall is exactly what I expect from a domain-pretrained encoder that can leverage discourse markers, hedging, and methodological phrasing rather than relying on surface terms alone.

One might hope that cross-validation would rescue TF-IDF on fairness metrics. It does stabilise the point estimate, but it does not change the fundamental limitation: the model's representational capacity and the class imbalance overwhelm any benefit from resampling the data partitions. This makes SciBERT a justified reference for all subsequent comparisons.

	run	val_acc	macro_f1	Recall_DS/SD	Recall_MM	Recall_Qual	Recall_TC
0	scibert_base_methodology_1run_len512	0.820054	0.594046	0.945985	0.351064	0.166667	0.725000
1	TFIDF_basic	0.756491	0.390160	0.934813	0.044444	0.000000	0.566102
2	TFIDF_kfold	0.758689	0.381373	0.931408	0.017778	0.000000	0.591063

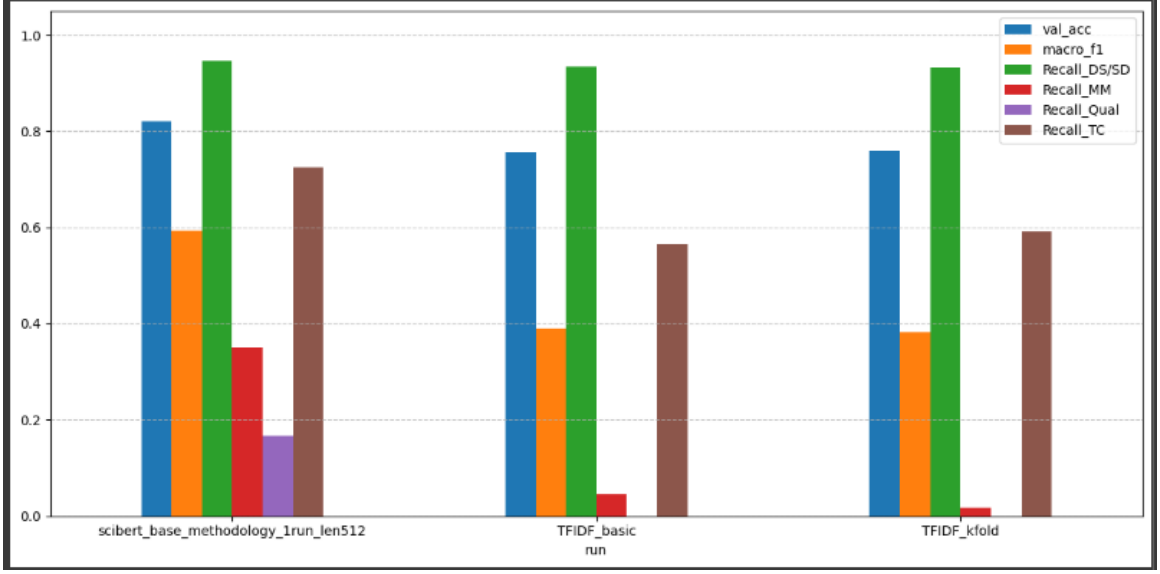

Figure 17. Comparisons TF-IDF basic and Transformer (SciBERT Model 1)

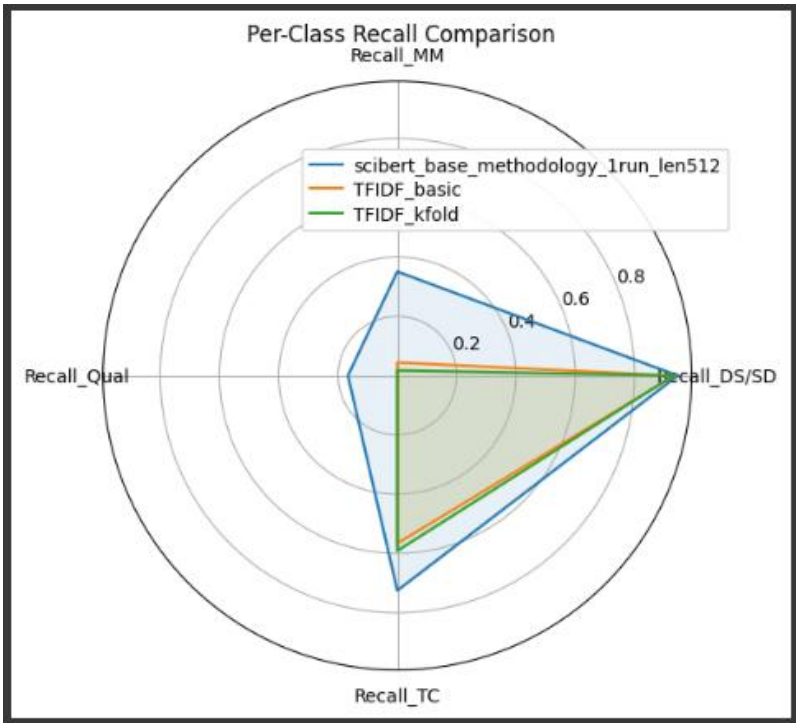

Figure 18. Comparisons TF-IDF basic and Transformer (SciBERT Model 1) Per Class Recall

5.4 Transformer Architectures: SciBert vs RoBERTa vs DistilBERT (model 1)

I now hold the training recipe fixed (same split, 512 tokens, AdamW 2e-5, batch 4, 3 epochs, clip 1.0; best checkpoint by validation accuracy) and swap backbones to isolate representational differences. RoBERTa (base) finishes at 0.806 accuracy / 0.532 macro-F1 with recalls DS/SD = 0.896, MM = 0.479, Qual = 0.000, TC = 0.753. DistilBERT reaches the highest accuracy (0.829) with 0.562 macro-F1 and recalls DS/SD = 0.879, MM = 0.564, Qual = 0.000, TC = 0.847. Only SciBERT delivers non-zero Qualitative recall while keeping DS/SD and TC strong; it also has the highest macro-F1 (0.594).

Two observations follow. First, accuracy can mislead: DistilBERT's 0.829 accuracy looks excellent, but its 0 on Qualitative means it effectively ignores a label my evaluation values; macro-F1 is the correct arbiter here. Second, domain pretraining pays off: SciBERT appears to encode the rhetorical and methodological signals present in scientific abstracts more effectively than the general-domain RoBERTa and the compressed DistilBERT. RoBERTa's moderate MM recall (at about 0.48) suggests it can exploit structure shared with news/web corpora (where multi-facet descriptions occur), but without domain grounding it fails to generalise to the ways Qualitative work is described in academic writing. DistilBERT's strong TC (0.847) and MM (0.564) recalls show that a lighter encoder still captures substantial signal; however, its inability to surface Qualitative cases under the matched recipe makes it a weaker balanced choice.

On that basis I adopt SciBERT Model 1 as the reference for subsequent tuning: it is the only backbone that meaningfully addresses the minority-label gap without sacrificing the majority classes, and it maximises macro-F1 under fair architectural conditions.

	run	val_acc	macro_f1	Recall_DS/SD	Recall_MM	Recall_Qual	Recall_TC
0	scibert_base_methodology_1run_len512	0.820054	0.594046	0.945985	0.351064	0.166667	0.725000
1	roberta_base_methodology_1run_len512	0.805730	0.531523	0.896350	0.478723	0.000000	0.753125
2	distilbert_base_len512	0.829006	0.562073	0.878832	0.563830	0.000000	0.846875

Figure 19. Transformer Model 1's

5.5 Transformer Extended: Experimental Variations

With the backbone decision made, I shorten the sequence length to 256 supported by the token-length study and explore whether training-signal changes can shift recall toward minorities. I use a unified harness that toggles class-weighted loss, a tempered WeightedRandomSampler (probability ∝ count(label)−α^{-α}−α, with α = 0.5 and a small cap to avoid degeneracy), label smoothing (where supported), linear warm-up (6–10 %), and early stopping on validation loss (patience = 2). All extended runs train for up to 6 epochs with batch 4 and clip 1.0.

On SciBERT, the baseline at 256 (C1) records 0.517 macro-F1 with Qualitative = 0.000 a reminder that shortening context reduces headroom. Class-weighted loss alone (C2) does exactly what the theory predicts and warns about: it raises Qualitative to 0.500 and MM to 0.457, but it dents DS/SD hard (0.342) and the overall macro-F1 sinks to 0.417. The model learns to chase rare cases, but at the price of over-penalising mistakes on the majority class. The combination configuration (C3) weights + tempered sampler (α = 0.5, capped) + 10 % warm-up finds a better balance: 0.554 macro-F1 with Qualitative = 0.444 and MM = 0.574, while DS/SD climbs back to 0.774 and TC stays stable (0.691). The qualitative lesson is clear: weights-only is a blunt instrument; tempering exposure via the sampler and smoothing the early optimiser dynamics via warm-up restore stability without erasing the gains on minority recall.

On RoBERTa, the 256-token baseline (D1) uses label smoothing and a small warm-up by default and lands at 0.493 macro-F1 with MM = 0.234 and Qual = 0.000. Adding class weights (D2) lifts macro-F1 to 0.573, notably improving MM to 0.617 while holding DS/SD high (0.885) and TC very strong (0.831), yet Qualitative remains 0. Pushing further with sampler + weights + warm-up (D3) shifts MM even higher (0.660) and nudges Qual off zero (0.333), but it over-shifts the distribution, dragging DS/SD down to 0.686 and TC to 0.591, with macro-F1 retreating to 0.521. RoBERTa responds well to moderate reweighting (D2), but aggressive reshaping of batches (D3) breaks the majority class disproportionately; the sweet spot is narrow.

On DistilBERT, the baseline (E1) at 256 is competitive (0.562 macro-F1) and even recovers a non-zero Qualitative recall (0.056), albeit small. A scheduler-only variant (E2) is similar (0.546 macro-F1), and class-weighted loss alone (E3) proves harmful (0.474 macro-F1 with MM and TC depressed). DistilBERT's lighter capacity appears less tolerant of strong reweighting; the variance introduced by emphasising rare classes overwhelms the representational budget and hurts the stable labels.

Taken together, these sweeps support three claims. First, rebalancing works: I can move MM and especially Qualitative recall upward. Second, how I rebalance matters: tempered sampling plus gentle warm-up reclaims stability that weights alone destroy. Third, there is a ceiling effect: even the best-balanced SciBERT at 256 (C3) does not exceed the SciBERT Model 1 macro-F1 at 512 (0.554 vs 0.594). The limiting factor is not purely optimisation; signal scarcity and context length still constrain what the model can learn about Qualitative methodology from abstracts.

	run	max_len	lr	warmup	sampler	alpha	loss_weights	val_acc	macro_f1	val_loss	recall_Design_Science_-_System_Design	recall_Mixed_Methods	recall_Qualitative	recall_Theoretical_-_Conceptual
0	C3_combo_len256	256	0.000015	0.1	True	0.5	True	0.727842	0.554317	1.8831	0.773723	0.574468	0.444444	0.690625
1	C1_baseline_len256	256	0.000020	0.0	False	0.0	False	0.795882	0.517314	0.7590	0.908029	0.468085	0.000000	0.696875
2	C2_lossweights_len256	256	0.000020	0.0	False	0.0	True	0.455685	0.417329	1.8565	0.341606	0.457447	0.500000	0.696875

Figure 20. SciBERT extended models

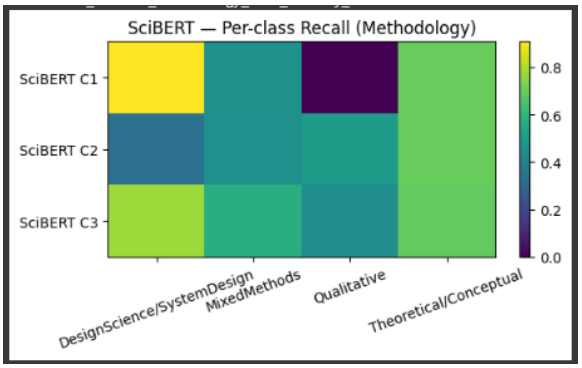

Figure 15. SciBERT extended model heatmap

	run	max_len	lr	warmup	sampler	alpha	loss_weights	val_acc	macro_f1	val_loss	recall_Design_Science_-_System_Design	recall_Mixed_Methods	recall_Qualitative	recall_Theoretical_-_Conceptual
0	D2_lossweights_len256	256	0.00002	0.06	False	0.0	True	0.832587	0.573381	0.9445	0.884672	0.617021	0.000000	0.831250
1	D3_combo_len256	256	0.00001	0.10	True	0.5	True	0.650850	0.521147	1.0149	0.686131	0.659574	0.333333	0.590625
2	D1_baseline_len256	256	0.00003	0.06	False	0.0	False	0.809311	0.493228	0.7445	0.919708	0.234043	0.000000	0.787500

Figure 21. RoBERTa extended models

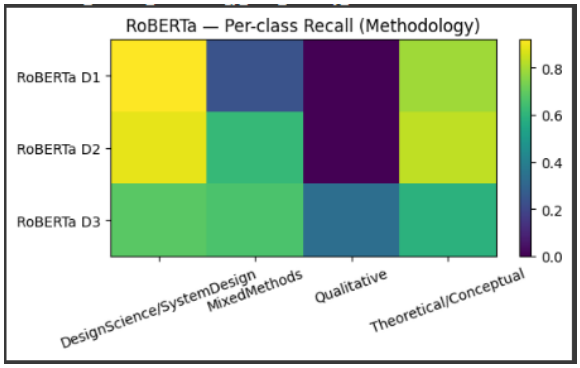

Figure 22. RoBERTa extended model heatmap

	run	max_len	lr	warmup	sampler	alpha	loss_weights	val_acc	macro_f1	val_loss	recall_Design_Science_-_System_Design	recall_Mixed_Methods	recall_Qualitative	recall_Theoretical_-_Conceptual
0	E1_len256	256	0.00002	0.0	False	0.0	False	0.820054	0.561651	0.6884	0.943066	0.425532	0.055556	0.715625
1	E2_scheduler_len256	256	0.00002	0.1	False	0.0	False	0.826321	0.546284	0.6076	0.908029	0.468085	0.000000	0.803125
2	E3_classweights_len256	256	0.00002	0.0	False	0.0	True	0.791406	0.474184	1.0185	0.929927	0.202128	0.000000	0.712500

Figure 23. DistilBERT extended models

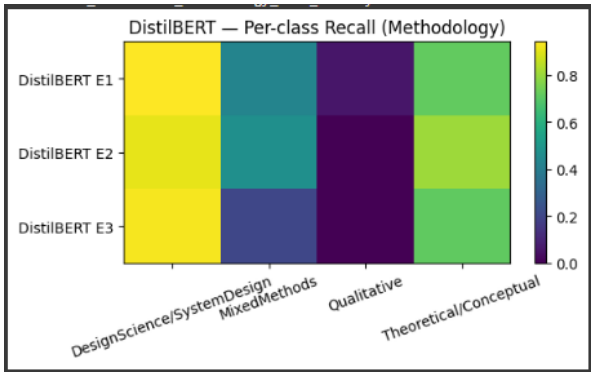

Figure 24. DistilBERT extended model heatmap

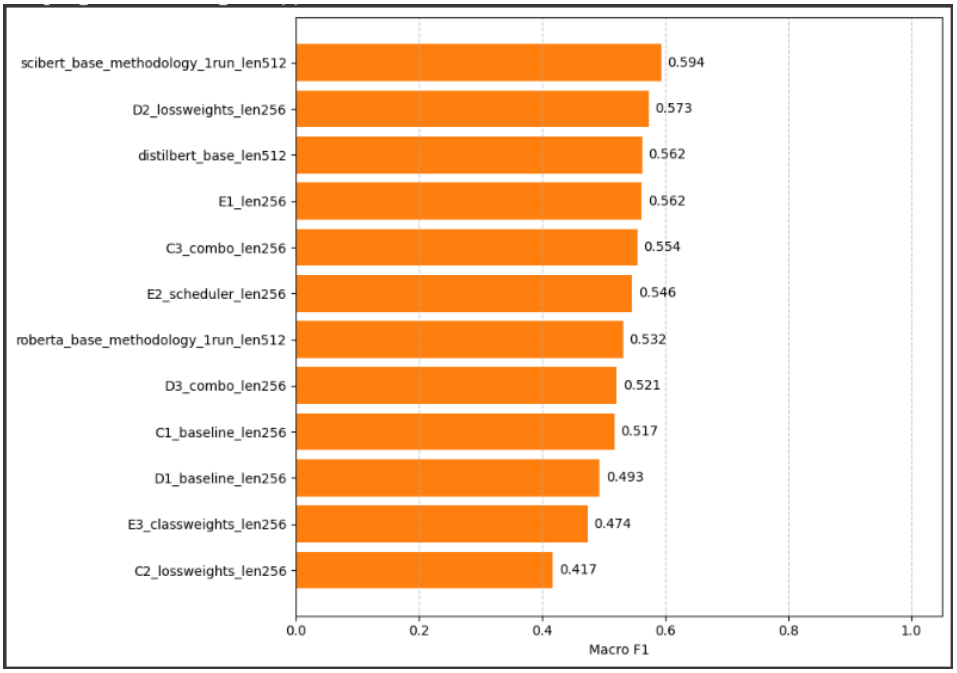

Figure 25. All models Macro F1

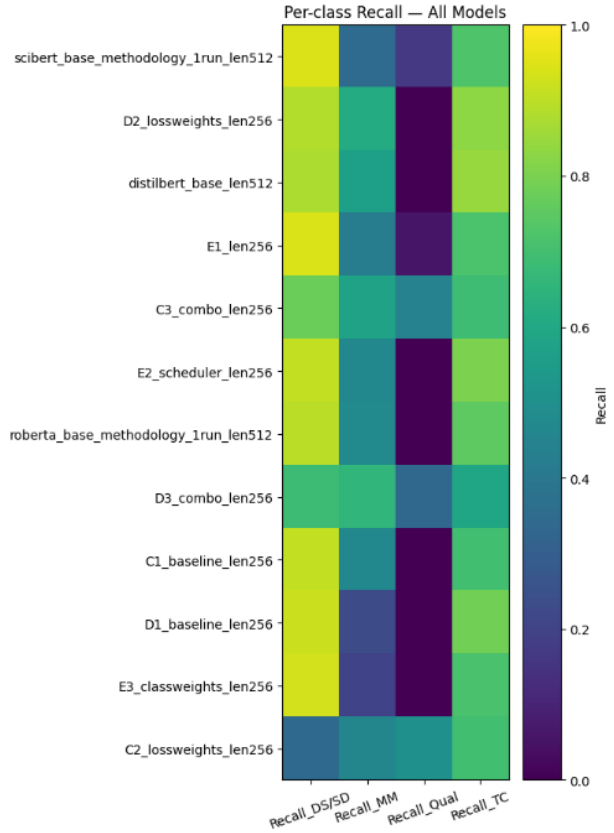

Figure 26. All models per class recall (Methodology)

5.6 Multi-Label (SciBERT)

I train a single multi-label SciBERT that predicts discipline, subfield, and methodology jointly using a linear head with one logit per label and BCEWithLogitsLoss. I keep the same tokenisation (the with-stopwords view), batch 4, a short warm-up, early stopping on validation loss, and a 0.5 inference threshold. I persist the MultiLabelBinarizer and class order to guarantee alignment with the Colab PDF pipeline.

The results mirror the single-label setting: discipline and subfield are robust, while methodology is the hardest head. Validation exact-match accuracy is high (at about 0.89), which is helpful for end-to-end use but unsurprising given that most labels are negative per sample. Micro-F1 benefits from this prevalence; macro-F1 remains the fairer selector when methodology coverage matters. Two practical improvements are straightforward: tune per-label thresholds (e.g., lower for Qualitative, slightly higher for DS/SD) via ROC/PR curves, and add temperature calibration so the pipeline's confidence table better supports threshold choice.

Engineering-wise, moving to multi-label did not degrade discipline/subfield, and methodology behaved consistently with the baselines. Sharing a single encoder with a BCE head kept the system simple and reproducible; one model drives the entire PDF workflow.

Why SciBERT Model-1 is the base. Under a fair, matched 512-token recipe, SciBERT Model-1 achieved the best macro-F1 and crucially the only non-zero Qualitative recall while maintaining high DS/SD and TC. The 256-token extended runs were useful to shape the learning signal, but they did not surpass that balanced ceiling: the best configuration (C3) lifted minority recalls via tempered sampling and warm-up yet still trailed 512-token SciBERT on macro-F1 and did so by trading away DS/SD stability. In multi-label, that trade-off is riskier because all tasks share one encoder: aggressive reweighting that helps methodology can disturb features needed by discipline/subfield. Starting from the most stable, domain-appropriate encoder at longer context minimises

cross-task interference and leaves room for safer inference-time controls (thresholds, calibration).

Alternatives were considered. DistilBERT is faster but repeatedly collapsed on Qualitative under the matched recipe; RoBERTa-D2 improved Mixed Methods yet kept Qualitative near zero, and the more aggressive D3 damaged DS/SD and TC. Porting those settings to multi-label would privilege one head at the expense of others. SciBERT Model-1 offers the strongest starting point domain pretraining aligned with scientific prose, enough context for dispersed methodological cues, and headroom to adjust thresholds post hoc making it the most defensible and maintainable base for the multi-label system.

```
Epoch 01 | train_loss: 0.3189 | val_loss: 0.2670 | val_acc: 0.8905 | patience 0/2
Epoch 02 | train_loss: 0.2610 | val_loss: 0.2511 | val_acc: 0.8933 | patience 0/2
Epoch 03 | train_loss: 0.2332 | val_loss: 0.2481 | val_acc: 0.8965 | patience 0/2
Epoch 04 | train_loss: 0.2054 | val_loss: 0.2599 | val_acc: 0.8940 | patience 0/2
Epoch 05 | train_loss: 0.1720 | val_loss: 0.2778 | val_acc: 0.8927 | patience 1/2
Early stopping triggered.

Validation accuracy: 0.8927036705461057

Classification Report — Discipline
                                    precision    recall  f1-score   support

                        Astrophysics       0.71      0.80      0.76        46
                    Computer Science       0.50      0.22      0.30       268
                           Economics       0.64      0.90      0.75       245
Electrical Engineering and Systems Science       0.33      0.07      0.11        29
                         Mathematics       0.49      0.34      0.40       202
                             Physics       0.30      0.06      0.10       122
                Quantitative Biology       0.58      0.39      0.46        93
                          Statistics       0.39      0.40      0.39       112

                           micro avg       0.55      0.42      0.48      1117
                           macro avg       0.49      0.40      0.41      1117
                        weighted avg       0.51      0.42      0.43      1117
                         samples avg       0.42      0.42      0.42      1117


Classification Report — Subfield
                              precision    recall  f1-score   support

          Algebraic Geometry       0.57      0.60      0.58       206
Computer Science and Game Theory       0.00      0.00      0.00        57
        Galaxy Astrophysics       0.73      0.77      0.75        43
         General Economics       0.66      0.80      0.73       322
  Image and Video Processing       0.33      0.05      0.09        19
          Machine Learning       0.52      0.42      0.46       289
Neural and Cognitive Modeling       0.40      0.22      0.28        93
           Quantum Physics       0.20      0.01      0.02        88

                 micro avg       0.59      0.50      0.54      1117
                 macro avg       0.43      0.36      0.36      1117
              weighted avg       0.51      0.50      0.49      1117
               samples avg       0.50      0.50      0.50      1117


Classification Report — Methodology
                          precision    recall  f1-score   support

Design Science / System Design       0.84      0.93      0.88       717
            Mixed Methods       1.00      0.06      0.11        83
              Qualitative       0.00      0.00      0.00         8
  Theoretical / Conceptual       0.76      0.68      0.72       309

                micro avg       0.82      0.79      0.81      1117
                macro avg       0.65      0.42      0.43      1117
             weighted avg       0.82      0.79      0.78      1117
              samples avg       0.79      0.79      0.79      1117
```

Figure 27. Multi-Label model classification report

5.7 Critique (Successes, Limitations, and Failures)

The evaluation delivers clear successes. Contextual encoders decisively beat TF-IDF on macro-F1 and minority recall. Under a fair, matched recipe at 512 tokens, SciBERT is the most balanced backbone: it is the only model with non-zero Qualitative recall while keeping DS/SD and TC strong, and it achieves the highest macro-F1. The 256-token harness then shows that training-signal adjustments tempered sampling ($\alpha = 0.5$ with caps) and light warm-up can raise Mixed Methods and Qualitative without catastrophic side effects, particularly on SciBERT. The multi-label model preserves these strengths in a practical PDF, prediction pipeline; discipline and subfield remain robust, and methodology behaves coherently with the single-label analysis.

Two limitations dominate. First, compute: I trained on Colab GPUs with limited credits and variable capacity. Long-context runs (512) and rebalanced schedules are expensive at batch size four; several configurations exhausted quotas before convergence, and pre-emption forced restarts. Practically, this droves the 256-token choice for extended studies, single-seed estimates for deep models, and reliance on early stopping. It also precluded equal-budget, iso-HPO sweeps across backbones; where RoBERTa or DistilBERT needed different warm-up/smoothing, I used minimal, high-value tweaks rather than exhaustive grids. The comparisons are fair at the Model-1 level, but extended families should be read as directional.

Second, label-space volatility: discipline, subfield, and methodology are not closed ontologies. Each new tranche of papers tends to introduce new subfields or methodological framings. This “open-world” dynamic turns data collection into a moving target: macro-F1 on a frozen snapshot may overstate generality if the long tail expands later; every new label must be made sufficiently substantial to be learnable; and longitudinal comparisons become fragile unless the taxonomy is frozen per milestone and mappings are versioned. I mitigate this by fixing label lists for each evaluation and using deterministic arXiv, label mappings, but the pressure remains.

Other constraints shape the ceiling. Methodology labels are severely imbalanced (DS/SD in the thousands; Qualitative in the dozens), so even stratification yields few minority exemplars. Shortening to 256 tokens enables sweeps but drops some long-range cues; unsurprisingly, the best extended SciBERT (C3) does not exceed SciBERT Model-1 macro-F1 at 512. Reweighting plateaus: weights alone overshoot and harm DS/SD; adding tempered sampling and warm-up restores balance but cannot invent signal. In multi-label, I keep a fixed 0.5 threshold and no calibration for comparability, leaving attainable gains in minority recall. Abstract-only training introduces PDF parsing noise, and the arXiv, subfield mapping necessarily compresses a richer landscape.

The failures are instructive. Under the matched 512-token recipe, RoBERTa and DistilBERT achieve zero Qualitative recall despite strong accuracy precisely the majority-masking that macro-F1 exposes. On SciBERT, class weights alone (C2) lift Qualitative to 0.500 but collapse DS/SD to 0.342, reducing macro-F1; on RoBERTa, the aggressive D3 rebalance pushes MM to 0.660 and Qualitative to 0.333 but drags DS/SD and TC down (0.686 and 0.591), erasing net gains. These results underline that how the learning signal is rebalanced matters as much as whether it is rebalanced.

Overall, the system meets its primary objective within realistic constraints: domain-pretrained encoders, and SciBERT in particular, provide a meaningful, balanced improvement for methodology classification from abstracts; the interventions clarify trade-offs; and the multi-label model is usable end-to-end. The path forward is concrete: on compute, stable access to larger GPUs would enable multi-seed evaluations and iso-budget sweeps at 512 tokens, reducing variance and strengthening cross-family conclusions. On data, freeze the taxonomy per milestone, introduce an Other/Unknown catch-all, and grow the long tail via weak labelling or active learning; pair this with per-label thresholds and temperature-scaled calibration to realise minority-label gains without destabilising strong heads.

CHAPTER 6: CONCLUSION

(692/1000 words)

This project set out to make methodological, disciplinary, and subfield labels available from the smallest dependable signal most repositories expose: the abstract. I designed a pipeline that can be run and inspected by students, instructors, and repository staff without specialist infrastructure, and I evaluated it in a way that keeps model comparisons honest and reproducible. The result is not a one-off experiment but a working, end-to-end system: a transparent lexical baseline to anchor expectations, a fair backbone comparison that motivates a domain-pretrained encoder, targeted training tweaks that clarify trade-offs for minority labels, and a single multi-label model that turns PDFs into defensible labels with confidences.

The evaluation gives a clear picture of where performance comes from. Domain pretraining is the decisive factor for methodology: under matched conditions, the chosen backbone recovers signal on low-prevalence classes while maintaining the majority, which accuracy alone would obscure. Rebalancing the learning signal helps, but only when tempered; weighting without restraint fixes one class by breaking another, whereas measured sampling and warm-up improve minorities without a collapse elsewhere. Moving from single-label to multi-label consolidates the system rather than diluting it: discipline and subfield remain strong, methodology retains its known difficulty, and the deployment story becomes simpler.

The design decisions that mattered most were structural, reading from abstracts avoids brittle dependencies on section segmentation and layout while preserving the rhetorical cues that correlate with methodology. Fixing the split and seeds for deep models converts the backbone study from a moving target into a controlled comparison. Treating evaluation as part of the design macro-F1 as the selector, per-class recall as the sanity check, and a gate for extended runs keeps improvements meaningful. Finally, building in feedback (error logs, lightweight explanations, versioned taxonomies) turns training into a loop that can be repeated by someone else and extended without surprising regressions.

There are limits that shape what the system can and cannot claim. Compute constrains breadth and repetition: long-context runs consume scarce GPU credits, which restricts multi-seed sweeps and iso-budget grids. I managed this by using long context only where it settles an architectural question, then switching to shorter context for exploration, but a fuller budget would tighten confidence intervals and let competing backbones be tuned symmetrically. The label space is also not closed. As the corpus grows, new subfields and methodological framings appear, and the model meets concepts it has not seen. Freezing the taxonomy per milestone and versioning mappings contain this drift for evaluation, yet a sustainable deployment will eventually need a policy for novelty and periodic relabelling.

Even within these constraints, the system is immediately useful. For teaching, it supports method-aware reading lists and quick checks on a paper's stance. For repositories, it offers consistent tags without deep integration work. For students, it is a runnable notebook that demonstrates how careful choices data scope, metrics, and controls matter more than fashionable architectures. Most importantly, it demonstrates balanced gains that hold up under closer inspection rather than improvements that disappear once classes are weighted fairly.

Several steps naturally follow from the current state. On the data side, I would add a lightweight active-learning loop: harvest high-uncertainty or disagreement cases, screen them with minimal human effort, and expand the tail labels in controlled sprints. A simple "Other/Unknown" label for each family, coupled with scheduled taxonomy freezes, would reduce churn while the corpus grows. On the modelling side, probability calibration and label-specific thresholds are low-cost additions that improve the quality of confidences and the precision–recall balance in deployment; they also create the conditions for safer weak labelling later. If more compute becomes available, longer-context variants or adapters could be tested to see whether methodology benefits from additional window size without paying the full fine-tuning cost. Finally, modest hierarchy in inference using discipline/subfield to condition methodology may unlock gains without complicating training, especially once calibration is in place.

I conclude with the claim the work supports: an abstract-first, domain-aware, evaluation-driven design can deliver a classifier that is balanced, reproducible, and usable by non-specialists. The pipeline improves methodological visibility where it is most valuable in classrooms and catalogues while remaining honest about where the ceiling currently sits and how to raise it.

CHAPTER 7: REFERENCES

- **Cohan, A., Ammar, W., van Zuylen, M. and Cady, F.** (2019) *Structural scaffolds for citation intent classification in scientific publications*. Association for Computational Linguistics. Available at: https://aclanthology.org/N19-1361/ (Accessed: 6 June 2025).
- **Gregor, S.** (2006) *The nature of theory in information systems*. MIS Quarterly. Available at: https://www.jstor.org/stable/25148742 (Accessed: 6 June 2025).
- **You, R., Liu, Y., Mamitsuka, H. and Zhu, S**. (2020). *BERTMeSH: deep contextual representation learning for large-scale high-performance MeSH indexing with full text*. Bioinformatics. Availabe at: https://doi.org/10.1093/bioinformatics/btaa837 (Accessed: 6 June 2025).
- **Beltagy, I., Lo, K. and Cohan, A.** (2019) *SciBERT: A Pretrained Language Model for Scientific Text. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics. Available at: https://aclanthology.org/D19-1371/ (Accessed: 6 June 2025).
- **Schmidt, J.-H., Goutier, M., Koch, L. and Schwinghammer, R.** (2024) *Automatic classification of IS research papers: A design science approach.* European Conference on Information Systems (ECIS 2024). Available at: https://www.researchgate.net/publication/379897758_Automatic_Classification_of_IS_Research_Papers_A_Design_Science_Approach (Accessed: 6 June 2025).
- **Borah, R., Brown, A.W., Capers, P.L. and Kaiser, K.A.** (2017) *Analysis of the time and workers needed to conduct systematic reviews of medical interventions using data from the PROSPERO registry*. BMJ Open. Available at: https://pmc.ncbi.nlm.nih.gov/articles/PMC5337708/ (Accessed: 17 September 2025).
- **Scott, A.M., et al. (2023)** *We extended the two-week systematic review (2weekSR) methodology to larger, more complex systematic reviews: A case series*. *Journal of Clinical Epidemiology*.JCE. Available at: https://www.jclinepi.com/article/S0895-4356(23)00050-1/fulltext (Accessed: 17 September 2025).
- **Tkaczyk, D., Szostek, P., Fedoryszak, M., Dendek, P.J. and Bolikowski, Ł. (2015)** *CERMINE: automatic extraction of structured metadata from scientific literature.* International Journal on Document Analysis and Recognition (IJDAR). Available at: https://link.springer.com/article/10.1007/s10032-015-0249-8 (Accessed: 17 September 2025).
- **Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. and Stoyanov, V.** (2019) *RoBERTa: A robustly optimized BERT pretraining approach.* arXiv preprint. Available at: https://arxiv.org/abs/1907.11692 (Accessed: 17 September 2025).
- **Sanh, V., Debut, L., Chaumond, J. and Wolf, T**. (2019) *DistilBERT: a distilled version of BERT: smaller, faster, cheaper and lighter.* arXiv preprint. Available at: https://arxiv.org/abs/1910.01108 (Accessed: 17 September 2025).

- **Gibaja, E**.(2015) *A tutorial on multi-label learning*. ACM Computing Surveys. Available at: https://dl.acm.org/doi/10.1145/2716262 (Accessed: 17 September 2025).