Automating Systematic Reviews: Generalisation to New Abstract Screening Tasks with Merged Expert LoRAs

Tomás Bermejo

Master of Science
Artificial Intelligence
School of Informatics
University of Edinburgh
2024

Abstract

Systematic reviews (SRs) of academic literature are a crucial part of many scientific fields including healthcare, but the large time investment they require from experts is a significant barrier to completing them and keeping them up to date. Pre-trained language models (PLMs) could offer a new opportunity to alleviate this burden by automating the abstract screening phase of SRs, in which potentially relevant papers are identified from a large pool of candidate papers. We investigate methods to fine-tune PLMs to carry out this task, which has not yet been studied in the literature, and find that the performance of a model can be greatly improved in this way, even if labelled data from the specific screening task to be carried out is scarce. Specifically, we utilise data from other existing SRs to train 'expert' LoRA adapters, which, when merged, boost the ability of the model to adapt to new screening tasks in a zero- or few-shot manner, doing so better than a multi-task learning approach. Our results suggest there is great promise in using data from many SRs to fine-tune such models, and motivate efforts to create larger multi-SR datasets to develop even stronger abstract screening models.

Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Tomás Bermejo*)

Acknowledgements

This work would not have possible without the help of my supervisors, Andrew Horne and Prof. Mirella Lapata, and I am thankful for their advice and support. I would also like to thank Mattia Opper for his invaluable guidance and encouragement throughout the project, and Marshall Dozier for kindly sharing her knowledge, support and enthusiasm on the topic. Finally, thank you also to EDINA and EPCC, who provided the necessary resources for the project to take place.

Table of Contents

Chapter 1

Introduction

1.1 Systematic Reviews

Systematic reviews (SRs) are reviews of academic literature where a rigorous, predefined process is followed to obtain and synthesise a comprehensive selection of literature that answers a given research question. The methods of an SR are designed to be as systematic and explicit as possible in order to reduce the influence of the authors on the results and thus mitigate the risk of bias and improve the reproducibility of findings. This makes SRs an important part of scientific research across many domains, and in particular in healthcare, where it is crucial to obtain reliable, unbiased data regarding interventions and their outcomes and where individual studies may often provide seemingly conflicting results [15].

However, the methods that make SRs reliable and objective also make them particularly costly and time-consuming, with a typical review involving 3-8 experts in the domain of interest and taking over a year to complete [9]. Within the field of medicine, [6] report that with the yearly number of newly published controlled trials growing incessantly, SRs cannot be created or updated quickly enough for the medical profession to keep up with new information. As such, methods to improve the speed and efficiency of the SR process are urgently needed.

1.2 Abstract screening with large pretrained language models

A key part of the SR pipeline is the *abstract screening* stage, in which authors are required to carry out a keyword search on multiple bibliographic databases and read the title and abstract of every returned result in order to determine whether the paper is likely to match a predefined set of inclusion criteria for the review. Such a search may return 10,000s of records for screening [5], representing hundreds of expert hours spent per review on a largely unfulfilling and repetitive task. As such, the abstract screening task would greatly benefit form automation.

Abstract screening requires deep understanding of often complex scientific language, which could traditionally only be achieved by a human expert. Yet recent years have seen the development and public release of large pretrained language models (PLMs) which show unprecedented skill in general natural language understanding tasks [41, 50, *inter alia*]. Furthermore, parameter-efficient fine-tuning (PEFT) methods make it possible to adapt such models to further boost their performance on a target task by making use of a relatively small labelled dataset. This presents a new opportunity for automation in SRs.

In this work, we seek to fine-tune a PLM using LoRA (a PEFT method, described in Section 3.5) to perform the abstract screening task of SRs, with the aim of establishing whether such a method could be used to automate part of the abstract screening workload of future SRs. A major challenge stems from the fact that every SR is effectively a somewhat different task, with a unique set of inclusion criteria that require identifying particular features in the input paper titles and abstracts. While it is possible to manually annotate a small sample of training data demonstrating the target review criteria, doing so for a large number of papers would be time-consuming and defeat the objective of automation. On the other hand, prior completed SRs are a large source of labelled data from which a model may be able to learn aspects of the general task of abstract screening, improving its performance on the target SR. We therefore explore two questions:

1. How effectively can we fine-tune a PLM for abstract screening on a particular SR when the amount of labelled data from that SR is limited?

2. Can we leverage labelled data from other SRs to improve the performance of the PLM on the target SR?

To tackle the second question above, we draw on recent research in multi-task learning and modular learning (discussed further in the next chapter) which suggests that we can use LoRA to fine-tune small modules (sets of model parameters) corresponding to different tasks and combine these within one model to improve the model's task generalisation abilities. Our vision throughout this work is to develop a system whereby a library of LoRA modules trained on past SRs can be utilised to obtain a model that performs strongly on new SRs, and where further trained LoRA modules can be added iteratively over time.

1.3 Outline and contributions

We begin with an overview of background literature in Chapter 2, covering previous work towards automating SR abstract screening as well as current research on PLM task generalisation and model merging, which we draw upon in this work. Chapter 3 introduces the general methods of the experiments, with 4 subsequently describing their implementation and results in more detail. We interpret and discuss our findings in Chapter 5 and provide some concluding remarks in Chapter 6.

Our results demonstrate that using fine-tuned PLMs is a highly promising strategy for abstract screening automation. We show that even fine-tuning on a small sample of manually labelled examples for an SR brings a notable performance gain that compensates for the time spent annotating the data, and that labelled data from reviews other than the target SR can be used to further improve performance. Additionally, contrary to what is usually seen in the literature, we find that we are able to achieve results equivalent or superior to those of a traditional multi-task learning approach by merging individual expert modules, which opens up the exciting possibility of developing these models in a distributed and iterative way.

Chapter 2

Background

2.1 Natural language processing for systematic reviews

As a result of the great scientific value and time consuming nature of SRs, attempts to automate part of the SR process through natural language processing (NLP) and text mining are plentiful and span the last two decades [39], with the majority of such works focusing on the abstract screening task [49], as we do here. The approach is typically to train a simple machine learning model – most commonly a support vector machine (SVM) [5, 49] – on TF/IDF counts of words or n-grams as features of paper abstracts. Such approaches have proved successful in real world case studies [22] and form the basis of commercial tools [43, 54], but since they only learn to detect surface-level properties of texts that relate to a particular SR, they must be re-trained on labelled data for the specific inclusion criteria and input papers of any target SR one wishes to carry out.

More recent studies [24, 18, 32] have tested the use of PLMs to carry out the abstract screening task. As a result of training on datasets that showcase responses to varied natural language descriptions of tasks, many PLMs are able to perform complex tasks specified in natural language without task-specific tuning [57]. The aforementioned studies exploit this property and find that PLMs offer a promising approach to the automation of abstract screening without the use of review-specific data, but their performance appears to vary considerably between reviews, highlighting the need for a more reliable system. To our knowledge, the present work is the first to employ fine-tuning of a PLM to the abstract screening task. We show this method can reap the benefits of both approaches mentioned, incorporating the general pretrained language abilities of a PLM while offering the option to improve performance using target SR

training data if available.

2.2 Task adaptation by combining adapters

Outside the specific application domain of SRs, our work generally falls under the topic of *task-level generalisation*, which is the challenge of training a model on datasets representing a range of different tasks to then perform inference on a new task for which very little or no labelled training data is available. In our scenario, we can consider each SR to correspond to one task, defined by its particular inclusion criteria and by the domain of the abstracts that must be screened for the review.

A straightforward yet strong approach to task-level generalisation is to simply fine-tune a model on data from all source tasks at once, minimising the sum of the respective task losses [61]. We refer to this approach simply as 'MTL' throughout this work, although many more sophisticated variants of multi-task learning exist [14]. MTL encourages the model to learn shared representations for features that are common to all the tasks, which can improve the model's performance and/or data efficiency on related downstream tasks [14]. However, while learning from source tasks can sometimes improve performance on the target task, the opposite is also possible if the tasks induce conflicting gradients on the model parameters [55]. Under the described MTL paradigm, there is no way to adaptively select the source tasks that are likely to lead to a performance improvement to the target task, unless we are prepared to repeat the entire fine-tuning procedure for every new target task. Furthermore, since sequential fine-tuning may lead to catastrophic forgetting [37, 46], training must be done using all source datasets at once, limiting our ability to incrementally improve a model as more data is acquired.

These considerations lead us to consider the *modular learning* [47] paradigm, where separate groups of parameters (i.e., modules) are used to learn specific tasks or skills, making it possible to avoid interference between tasks by separating them into different parts of the model. This is a rich and broad area of study, with design decisions to be made concerning the architecture of modules, the way they are combined within a model and how they are trained. While a large number of designs exist in the literature relating to these areas, the right choices in a given setting are likely to be heavily application dependent and extensive comparative studies are lacking [47]. In the present work, we narrow our focus to the general approach of training task-specific (i.e. review-specific) parameter-efficient LoRA [27] modules and dynamically combining them within a

base model at inference time. Such an approach is relatively simple to apply, not requiring any application-specific architecture, and opens the door to easily adjusting the use of different modules depending on the target review. It also results in a system whereby new modules can gradually be trained and added to the system over time (known as continual learning [7]) as labelled data for new reviews becomes available, and where training could in theory be carried out in a distributed way, with different parties training different experts using different datasets. We interchangeably refer to these review-specific LoRA modules as 'adapters' or as 'experts' in this work, given their conceptual similarity to expert modules in a Mixture-of-Experts [11] framework.

Under the described paradigm of expert module selection, two key decisions are (1) how to combine the parameters of different modules and (2) how to select which modules to combine to process a particular test item. The following subsections introduce some key existing literature relevant to each of these concerns.

2.2.1 Combining parameter-efficient modules

Given two equally structured modules, the simplest approach to combining them is to take a parameter-wise average to produce a single module. [62] show that this simple approach can be applied to LoRA modules in order to combine their respective skills, defining the merged module parameters as $\theta_{\text{merge}} = \lambda\theta_1 + (1-\lambda)\theta_2$, where θ_1 and θ_2 are the parameters of the modules to be merged and $\lambda \in [0,1]$ is a hyperparameter controlling the relative influence of each module. They apply this method (AriMerge) to merge two modules trained on different NLP tasks, showing that the resulting merged module performs well in both tasks. Through our experiments, we are aiming to take this a step further and test whether several modules merged in this way will also generalise better to completely unseen tasks, as a result of the composition of generally useful skills.

While parameter-wise averaging often works well in practice, it is not fully understood from a theoretical perspective. Several alternative theoretically-grounded methods to merging modules or entire models have been proposed, often giving stronger results than AriMerge in various settings [36, 30, 16]. In our experiments, we focus on TIES-Merging [60] since this has been found to perform better than alternative methods specifically in the setting of combining models trained on different tasks to improve generalisation to new tasks within NLP. TIES-Merging is a method to merge

any number of LoRA modules[1], consisting of three steps known as 'trim', 'elect' and 'disjoint merge'. In the 'trim' step, each module is pruned down to $k\%$ of its weights, by setting all parameters that fall within the bottom $(100-k)\%$ of parameters in the module in terms of magnitude to 0. Then, the 'elect' involves calculating the sign of the sum of values over the module for each parameter. That is, representing the trimmed parameters of module i as a vector θ_i, a 'sign vector' for the n module to be merged is calculated as $\text{sgn}(\Sigma_{i=1}^{n}\theta_i)$, where the 'sgn' function applies elementwise. Finally the merged parameters are obtained by taking an average over the module for each parameter only including those parameters that have not been trimmed and whose sign agrees with the sign in the corresponding position of the sign vector from the previous step.

TIES-Merging is motivated by the fact that merging models creates a risk of destructive interference between parameters, which increases as the number of merged models increases. Removing parameters that are redundant or conflicting in their sign should help to reduce this risk. This is a promising approach to try in our setting, where experts for different reviews may have conflicting parameters.

2.2.2 Selecting experts to maximise transfer

We refer to an improvement on a target task caused by training on a separate task as 'positive transfer', and use the term 'transferability' to refer to the degree of positive transfer that can be obtained from a particular combination of training and evaluation tasks. The amount of positive transfer that can be obtained from one task to another naturally depends on properties of the tasks, such as the degree to which they depend on overlapping skills [64]. We aim to investigate how such properties can be identified between our tasks, so that the most useful experts can be combined for inference on a particular target review task. One existing approach, AdapterSoup [13], is to make use of sentence embeddings of input text from each of the training datasets and the target dataset. LoRA experts trained on the the individual training datasets are selected if the average embeddings of those datasets have a cosine similarity with those of the target dataset that is above some threshold value. In [13], the authors combine the selected adapters via simple parameter averaging, but as discussed above, this can result in interference when merging several adapters; we consider whether TIES-Merging

[1]TIES-Merging is formally defined in terms of merging *task vectors*, which are vectors of parameter differences between a particular task and a parameter initialisation that is shared between tasks. We simplify things by referring to vectors of LoRA module weights, which are an example of such vectors.

could yield better results. Furthermore, AdapterSoup is specifically targeted towards *domain* rather than *task* adaptation, i.e., the model is only trained and evaluated on language modeling, with each dataset representing a different style of language. We are interested in seeing whether such an approach can also prove beneficial in our setting, where we are performing classification tasks over datasets relating to somewhat different academic domains.

An alternative approach for expert selection is to focus on the expert modules themselves, rather than the task data. Prior work [53, 1] has found that vector embeddings for tasks can be calculated from model parameters, and their similarity can be used as a proxy for transferability between them. In the context of PEFT, [63] found that the simply taking all the parameters of a PEFT module such as LoRA as a vector embedding can serve this purpose. Unfortunately, obtaining such an embedding for a target task requires some labelled data with which to train a PEFT module. We will investigate whether fine-tuning on the very limited data set available in the few-shot adaptation setting is enough to create useful task embeddings.

Chapter 3

Methods and data

3.1 Problem definition

In order to answer the research questions posed in 1.2, we carry out a series of experiments where the goal is to build a binary classification model that correctly predicts the inclusion decisions made by human experts for a collection of paper titles and abstracts at the abstract screening stage of an SR. We consider 'include' as the positive class of this classification problem, and 'exclude' as the negative class. We refer to the review on which the model is evaluated as the 'target SR'. A model may be trained on expert-labelled data corresponding to a small proportion of all the screened papers for the target SR, and/or on labelled data from other SRs.

For our experiments investigating task-level generalisation using data from other SRs, we define two settings of interest: the zero-shot setting, where no labelled data from the target SR is used, and the few-shot setting, where we allow access to 50 labelled items from the target review of which a minimum of 5 must belong to the positive class.[1] In practical terms, such a dataset would be created by an expert manually labelling abstracts returned from a SR search query until the minimum numbers of total items and positive items are met, discarding any negative items found beyond the 45th. This represents a tradeoff between ensuring that the set carries some useful information on both classes while allowing for an expert to manually curate the set in under an hour[2].

[1] In theory, we would not want the number of negative class items to be smaller than 5 either, but in practice this is not an issue since all of our datasets are dominated by negative items.

[2] [33] estimate an expert's screening rate at 60-120 abstracts per hour.

3.2 Datasets

We were provided with access to nine datasets, each corresponding to one SR carried out by University of Edinburgh researchers, extracted from Covidence[3], an online SR management platform. Each dataset contains the titles and abstracts of all papers that were returned from the original keyword search and screened for the review, along with the decision that was made at the abstract screening stage ('include' or 'exclude'), and the title and inclusion criteria of the review itself. We refer to each review and its corresponding dataset by a given letter, from A to I. Throughout the experiments, we often use the term 'review' to refer to a particular dataset and its associated classification task, where this is clear from context.

We received the inclusion criteria for the reviews in a variety of formats, such as tables of criteria included in the published reviews, flowcharts used by the authors in the screening process, or simply textual descriptions. We therefore standardised these by converting them into a passage of text per review that could be fed to the model, structuring this in sections corresponding to kinds of criteria such as the type of study, topic, or scope. An example can be found within the example prompt in Appendix C. The reviews corresponding to our datasets broadly relate to public health but vary substantially between them in subject matter. The datasets also span a wide range in terms of their sizes (from 643 to 6976 total papers) and in their proportions of papers marked for inclusion (from 1.88% to 37.0%). Dataset H contains only 26 positive class items, which we do not consider enough to form representative training and test splits, so we only use this dataset for the purpose of multi-task model validation. We hold out dataset I from all of our experiments until the final evaluation of our model, so it can be used as a fair test set.

Any items from the datasets where the title or abstract fields are empty are removed. Items with abstracts in a language other than English are also removed, since our base model is pretrained with a large majority of English text [2]. Non-English texts are identified using the fasttext [31] text classification Python module. Abstracts are truncated to a maximum of 8000 characters to avoid very long prompts to the model, which could go beyond the model's maximum context size or exhaust available GPU memory. A description of each dataset can be found in Appendix A.

[3] https://www.covidence.org/

3.3 Models and hardware

One would expect performance in our experiments to increase with the size of the base PLM, yet the time constraints and hardware available for the project (a mixture of NVIDIA A100 40/80 GB GPU units in a shared cluster) place a limit on the size of model that can feasibly be fine-tuned in these experiments. Furthermore, by using models that can be run and fine-tuned on resources typically available to academic researchers, our results are more likely to be useful and reproducible by teams working on SRs.

To obtain a good tradeoff between performance and computational cost, we run all of our experiments (except where explicitly stated otherwise) using Meta Llama 3 8B Instruct [3] and use the 4-bit NormalFloat data type and Double Quantisation as described in [19] while fine-tuning to further reduce the model memory footprint. This allows fine-tuning with a typical batch size on 40 GB of GPU memory while obtaining reasonable performance.

We obtain pretrained model weights from the Hugging Face Hub[4] and implement model training and evaluation using the Transformers [58] and Pytorch [4] Python libraries.

In order to obtain a baseline to which we can compare our results, we also use the much simpler model from [22], an SVM classifier which can be trained on a typical personal machine in seconds. We use the implementation provided by the authors.

3.4 Classification task

In order to leverage the general natural language understanding and instruction-following capabilities of the base PLM, we frame the abstract screening task in a text-to-text format where the task is fully described in each input to the model. To obtain a prediction, the model is passed a prompt made up of the following elements in the specified order (see Appendix C for an example of the prompt):

- General instructions describing the task.
- The title and inclusion criteria of the SR.
- The title and abstract of the research paper to consider.

[4] https://huggingface.co/

- Final question encouraging the model to generate ‘Yes’ or ‘No’ in response.

To obtain baseline results, we also carry out experiments using in-context learning and a base model without fine-tuning. In these experiments, we add a number of examples to the prompt consisting of a title and abstract and the correct response (see Section 4.1).

A probability distribution over the two classes is obtained by taking the model’s logits corresponding to the ‘Yes’ and ‘No’ tokens at the next token position and passing these two values through the softmax functions, as is typical in ‘prompt-based learning’ [34]. These probabilities are converted to discrete predictions by comparing them to a threshold value, which can be adjusted to bias the model towards a particular class (see 3.6 Evaluation). In theory, there is a risk that the model could give high probability to responses that indicate a positive or negative response using different token patterns (e.g., ‘yes‘ without capitalisation, or ‘Yes’ preceded by a newline token) so in our tests we also compute the softmax over the entire vocabulary and record the sum of probabilities assigned to the ‘Yes’ and ‘No’ tokens, to verify that the model is conforming to the expected output format.

3.5 Fine-tuning

Given the aforementioned constraints on computational resources and the limited size of some of the available datasets, as well as our focus on modular approaches, using a parameter-efficient fine-tuning (PEFT) technique is the natural choice. Compared to full fine-tuning, PEFT methods greatly reduce the cost and memory use of fine-tuning by only tuning a small fraction of the model’s parameters, while often achieving achieving superior performance in low-resource settings [12] and better generalisation [21].

We opt for LoRA (Low-Rank Adaptation) [27] as our chosen PEFT method, since it is widely used, easily implemented through open source Hugging Face libraries and obtains strong performance relative to other PEFT methods that train a similar or smaller number of parameters [59]. LoRA modifies a given model parameter matrix, $W_0 \in \mathbb{R}^{d\times k}$, with the addition of a low-rank learnable matrix represented as BA, where $B \in \mathbb{R}^{d\times r}$, $A \in \mathbb{R}^{r\times k}$ and the rank r is a hyperparameter which should be much smaller than the original matrix dimensions d and k. The resulting adapted weight matrix is given by

$$W = W_0 + \alpha BA$$

where $\alpha \in \mathbb{R}^+$ is a hyperparameter controlling the influence of the adapted parameters relative to the base parameters. B is initialised as a matrix of zeroes, yielding the original weight matrix before training begins. We initialise A with values from a Kaiming uniform distribution [26], as per the official code accompanying [27].

Following [27], we apply LoRA to the query and value matrices of the self-attention block in each Transformer [52] layer of our model, using different A and B matrices for each target matrix. During fine tuning, only the A and B parameters are adjusted, while the rest of the model is kept frozen. For the Llama 3 8B model, this results in training 3.4 million parameters, or 0.075% of the full model.

We train the model by minimizing the cross entropy loss between the model's probabilities for the single next token after the given prompt and the target labels. Note that unlike at inference time, where only the logits for the 'Yes' or 'No' tokens are used to obtain a two-point distribution, the training loss is calculated with respect to probabilities over the full vocabulary, so that the model is discouraged from producing invalid tokens that do not correspond to either of the classes.

3.6 Evaluation

Evaluation of our models serves two primary purposes: 1) to compare the quality of different models and hence establish the most effective techniques to solve our tasks, and 2) to provide interested parties with a clear picture of the potential effectiveness of the developed models in a real use case. The former requires the definition of a single established metric on which models can be systematically compared, while the latter calls for a range of measures such that decision makers can weigh up and consider the many facets of the tool's effectiveness.

In the context of an SR, excluding a paper that meets the review inclusion criteria at the abstract screening stage is a serious issue, potentially leading to incomplete and biased review results. False positives are less problematic, since irrelevant papers that pass the abstract screening stage can be removed by human experts at later stages of the process, but if the number of such errors is large, the tool will be of little value. As such, the Cochrane Handbook for SRs specifies that the aim of a search for papers is to balance precision and recall, with recall being most critical [33], but the exact trade-off between the two metrics is a subjective decision to be made on a case-by-case basis.

Since our models produce continuous-valued outputs, we can easily adjust the threshold value above which the positive class is predicted, in order to strike a desired

balance between precision and recall. Therefore, to illustrate the real-world potential value of our final model, we report on its precision at the threshold that achieves at least 95% recall, which is a reasonable target since it is estimated to be the level of agreement between two independent human screeners [5].

However, for the purpose of developing the model and choosing between alternative methods and hyperparameters, a single metric to optimise is necessary. Rather than arbitrarily choosing a minimum recall value or classification threshold, we measure the average precision score, which approximates the area under the precision versus recall curve of the model with a weighted average of precisions over different thresholds. Specifically, we consider a list of all possible threshold values that would result in distinct partitions into the two classes of all the predicted outputs, in ascending order. Then, the average precision is the sum, over all the thresholds n, of the precision achieved at the threshold (P_n) multiplied by the increase in recall relative to the previous threshold ($R_n - R_{n-1}$):

$$AP = \sum_n (R_n - R_{n-1}) P_n \tag{3.1}$$

We use the scikit-learn [45] implementation of this metric.

As noted in 3.2, the positive class rate varies dramatically between different review datasets, and we observe in our experiments that the probabilities produced by the models are often poorly calibrated. We therefore expect that in a real use case, the classification threshold would be manually adjusted for a given target review, based on a number of initial examples. Measuring average precision allows us to measure the models' predictive skill independently of its degree of calibration. While this could also be achieved by measuring the area under the receiver operating characteristic curve (AUROC) [38], this metric is based on specificity, i.e. accuracy over all negative class items, and therefore typically gives artificially high values for most models on problems that have a very high proportion of negative class items, such as the ones considered here. On the other hand, average precision has the desirable property that if the search criteria providing the papers for a review dataset was made far too wide and a number of obviously irrelevant papers were added to the dataset which the model easily identified as false, this would not affect the score.

3.7 Plan of experiments

We will begin our experiments (4.1) by testing the capabilities of PLMs on our task without any fine-tuning, as this will serve as a baseline for out work. We will then look to fine-tune a model on a particular SR (4.2) and investigate the tradeoff between the amount of training data used and the performance benefit, allowing us to answer question (1) posed in the Introduction (1.2), as well as serving as a point of reference for subsequent work on task generalisation. Following this, we will tackle question (2), concerning the use of past SR data to generalise to a new SR, by breaking this challenge down into several steps. Firstly, we will train a LoRA adapter on each of our individual review datasets and examine their performance when evaluated on other datasets 4.3. We will then make use of these adapters for an investigation of the performance of different merging methods to combine adapter abilities 4.4, followed by a comparison of several possible methods to predict adapters' cross-task abilities. Finally, we will combine the findings from previous experiments to develop a full approach to selecting and merging individually-trained expert LoRA adapters for use on a target SR, comparing this to a strong baseline of multi-task learning.

Chapter 4

Experiments

4.1 Zero- and few-shot prompting

Recently developed PLMs are able to perform complex tasks without any task-specific fine-tuning, by following instructions given in natural language in their prompt (zero-shot prompting), possibly with the addition of examples of the task with corresponding expected responses (few-shot learning) [10]. Hence, we may expect reasonable performance from our models on the abstract screening without performing and any fine-tuning. This represents a performance baseline which fine-tuned models must significantly surpass in order to justify the additional costs of fine-tuning and obtaining labelled training data. We also make use of zero-shot tests to inform the prompt design for the rest of the experiments. We compare the Meta Llama 3 8B Instruct model to OpenAI's GPT-4o model, a private model accessible through OpenAI's API [40]. While the GPT-4 family of models are much larger and more powerful than Llama 3 8B [41, 44], it is worth making this comparison since there may be situations where using the OpenAI API in a zero- or few-shot setting is more practical and/or cost effective than fine-tuning a smaller, self-hosted model.

Through these experiments, we also test a number of different approaches to formatting prompts and in-context examples in order to determine an effective format to use throughout the rest of the experiments, which is shown in Appendix C. The zero-shot results for each of the models on each review dataset (using the most effective prompt format for each model) are shown in Table 4.1. As expected, GPT-4o achieves the higher average precision score, showing a greater skill in separating the two classes over the full range of classification thresholds, but surprisingly, Llama is superior in terms of its F_1 score when evaluated at the default threshold of 0.5, or if the threshold is

Model	F_1	Average precision	Precision @ 95%
GPT-4o	0.631	**0.847**	0.333
Llama 3 8B	**0.729**	0.814	**0.536**

Table 4.1: Performance of different models when evaluated with zero-shot prompting on dataset A. 'Precision @ 95%' represents the best precision that can be obtained while setting the classification threshold low enough to obtain a recall of at least 0.95.

set high enough to achieve a recall of 0.95.

Although the use of in-context examples has been shown to improve PLM performance on a wide variety of tasks and particularly for very large models [10], we find that it leads to worse results for the Llama model, and only a slight improvement for GPT-4o, as shown in Figure 4.1. For these experiments, in-context examples are chosen manually from the training split of the data to ensure representation of both class labels and different types of research paper.

A possible explanation for these results is that examples are long (each containing the abstract of a research paper) and contain relatively little information from which to learn the task, since each label is simply a 'Yes' or 'No' answer without an explanation of which parts of the abstract were relevant (or not) to the review criteria. Extracting useful information from such examples requires complex reasoning which the smaller Llama model may not be capable of, and it is likely that any potential benefit from the examples is cancelled out by the effect of crucial semantic information from the review criteria and test abstract parts of the prompt being 'drowned out' by the examples. We find that repeating the review criteria in a summarised form after the examples somewhat improves results, supporting the hypothesis that crucial information about the review is being diluted, but this still does not result in an improvement over zero-shot prompting for Llama. While it may be possible to obtain better results by improving the helpfulness of the examples, e.g. through more sophisticated methods of selection, or with chain-of-thought reasoning [56], this lies outside the scope of this project.

4.2 Fine-tuning on the target review

Having established a training-free baseline, we next turn to fine-tuning the Llama 3 model with LoRA using data from the target review dataset, which we refer to as the

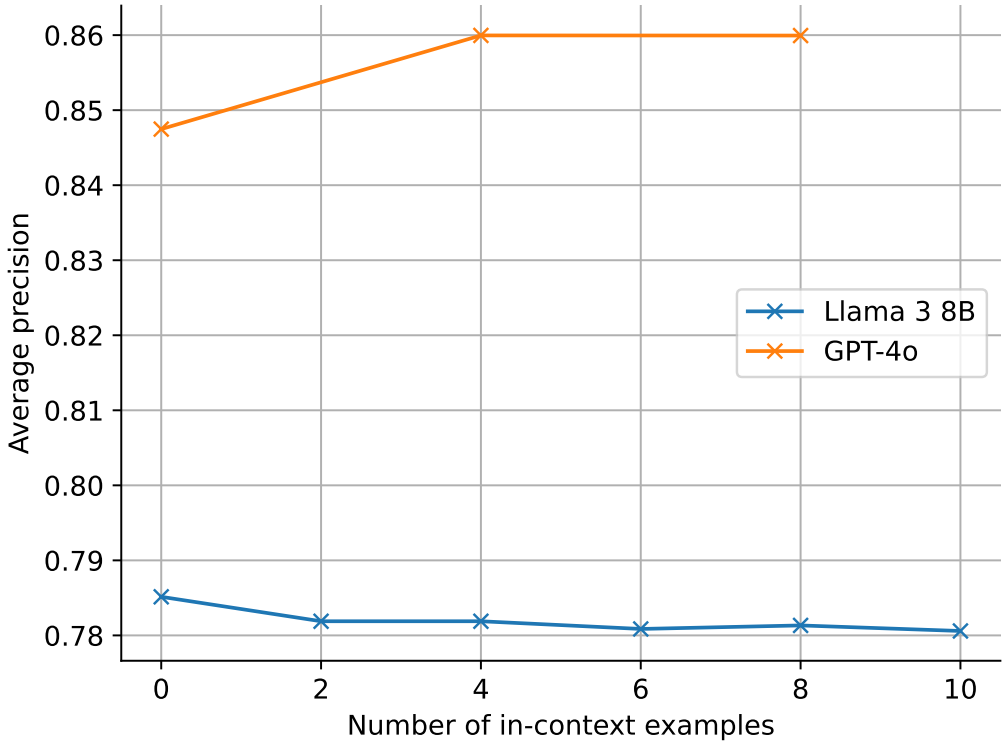

Figure 4.1: GPT-4o performance improves with inclusion of in-context examples, while Llama is hindered by the examples. Results for > 0 shots include a repeated summary of the review criteria after the examples in the model prompt, which somewhat improves results. We ran fewer evaluations on GPT-4o due to the per-token cost of inference via its API.

'in-review' setting. This would of course not be an feasible strategy in a real scenario where the target review has a very narrow scope, since there may not be enough existing papers that match the criteria to build an effective training set. Nevertheless, some SRs can select thousands of papers at the abstract screening phase (e.g. review A of our datasets); for such reviews, it may be an effective strategy to manually label a few hundred papers and use these to fine-tune a model that can identify the rest. In order to establish the tradeoff between training data size and model performance, the experiments in this section evaluate models fine-tuned on subsets of varying sizes of the training data for the target review. The results are also compared with the baseline SVM model of [22] trained and evaluated on the same data.

We randomly partition dataset A into training, validation and test splits according to a 75:15:15 ratio. This is the only dataset used for these experiments, since other datasets are not large enough to create such a partition with a sufficient number of items per split. When creating subsets of the training dataset of particular sizes, we employ two different approaches: 1) sampling a fixed number of positive and negative class examples, to obtain a 50:50 balance between the classes; 2) sampling over the whole dataset at once, to obtain a similar class balance to the underlying dataset (which, for almost any SR, will be a greater proportion of negative items [17]). While varying the distribution of classes risks creating a bias in the model towards the over-represented class, this can be compensated for when selecting a classification threshold at test time, and should not affect the average precision score we are interested in. On the other hand, we theorise that a positive item, as an example of all the inclusion criteria that the model is required to identify, is likely to be a richer source of task information than a negative one, so balanced sets could prove to be more beneficial.

Throughout the LoRA experiments, we use the AdamW optimizer [35] and a linearly decaying learning rate, tuning its inital value as a hyperparameter. Additionally, we multiply the learning rate of the B matrix by 16, as suggested by [25]. Optimal values of hyperparameters may vary between different dataset sizes, yet we do not have the time and resources to do a comprehensive hyperparameter sweep for each size. We therefore select hyperparameters as follows. Informed by online documentation and reports of similar experiments, we begin with a base configuration of a LoRA rank of 4, 0.05 LoRA dropout probability, and LoRA α value of 16, and vary each of these individually within a range around these values for three representative dataset sizes (100, 600, 2000), running three experiments with different random seeds for each configuration and evaluating on the validation set. We find that for most hyperparameters, there is a

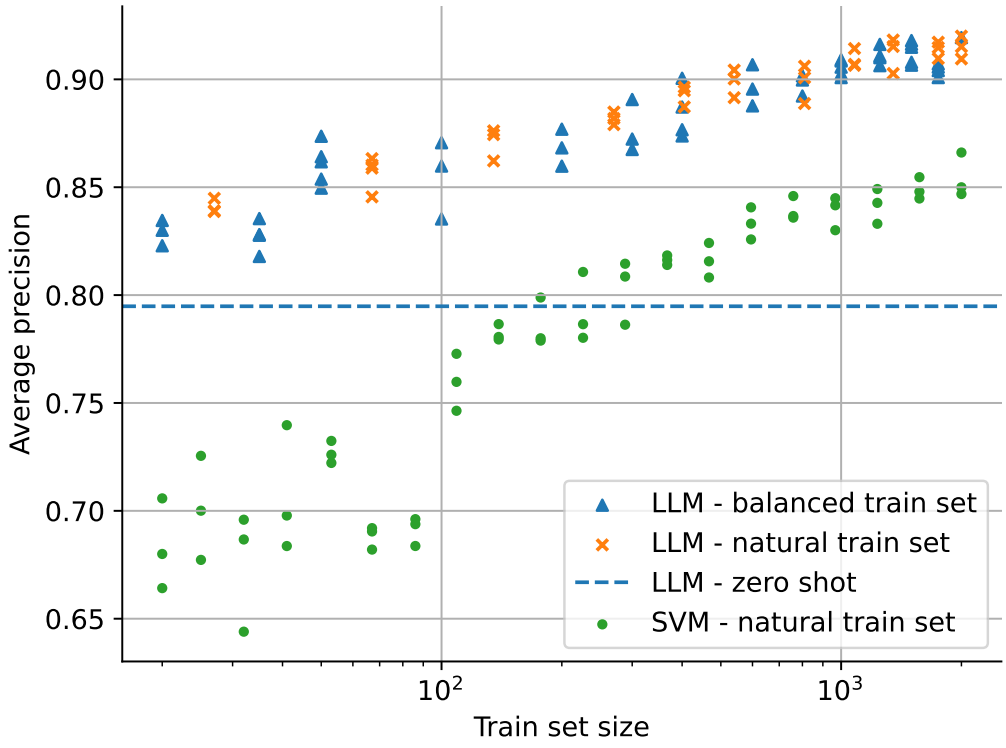

Figure 4.2: Validation set performance of models trained on differently-sized subsets of the training data for the target review. A 'natural' training set has the same class ratio as the underlying dataset, while a 'balanced' one has an equal ratio of positive and negative items. The training data is re-sampled for each experiment.

setting which performs better than or on par with all other settings at every dataset size, so use these values throughout. The exceptions are the LoRA rank and the number of training epochs, for which we carry out further evaluations on the validation set at finer dataset size intervals to establish the correct settings for each size. Finally, we train the model with the selected hyperparameters at each dataset size (listed in Appendix B) and evaluate on the test split, obtaining the results shown in Figure 4.2.

The results show that while a larger training set leads to better results, even a dataset of as few as 20 items can be used to obtain a sizeable performance boost through fine-tuning. Furthermore, balancing the classes in the datasets appears to make little difference to the outcome, which is encouraging since manually creating a balanced labelled dataset would require an expert to evaluate a greater number of abstracts. However, it should be noted that this dataset has a relatively high positive class rate of 37%, and some over-sampling of the positive class could still be required for datasets

that contain an extreme class imbalance. The figure also shows a clear superiority of the fine-tuned PLM approach relative to the baseline SVM method across all tested dataset sizes, as well as the SVM requiring more than 100 training examples to match the zero-shot performance of Llama 3.

4.3 Cross-review generalisation

The above results show that when targeting a medium or large SR, training on a manually labelled subset of the full set of candidate papers is a promising approach. However, this is unlikely to be effective for reviews with fewer than 100 relevant papers, and it has the added drawback that fine-tuning is required for every new SR that one wishes to carry out. Therefore, we now turn to investigate the 'cross-review' setting, where we attempt to improve a model's performance on a target review using training data from different reviews.

Intuitively, one may expect the degree to which learning can be transferred between two review tasks to depend on properties of the two reviews, such as the similarity between the review domains or between their criteria. An understanding of such factors would be beneficial in guiding efforts to create effective datasets for SR abstract screening, as well as to adaptively select which training data should be used in order to generalise well to a particular target review. To this end, we train a LoRA adapter on each SR dataset and evaluate each adapter on each dataset. Given that we have 8 datasets excluding the held out test set, and one of these (H) is only used for evaluation due to its small size, this gives $8 \times 7 = 56$ combinations of training and target reviews. Since our primary focus is on inherent properties of particular SRs and their inclusion criteria rather than specifics of the datasets, all training datasets are balanced in terms of their class ratio within this section.

Each adapter is trained using the hyperparameters identified in section 4.2 for the dataset size that most closely matches the training dataset size. Since most of the datasets have 100–300 positive class items each, a 30% validation split would contain only 30–90 positive examples, which we consider insufficient to obtain reliable performance metrics. Therefore, we use the entire target dataset when evaluating an adapter that has been trained on a different review. For the in-review case (i.e. when the training and evaluation reviews match), we use 5-fold cross-validation so that evaluation is still over the full set of examples. To keep comparisons between the in-review and cross-review cases fair, adapters are always trained on the same amount of data that would be used

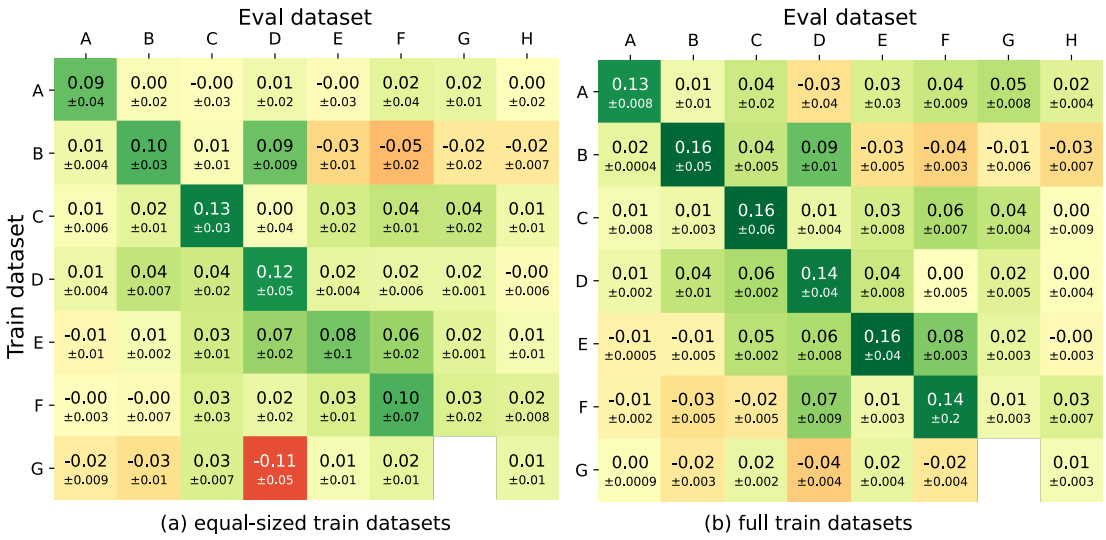

Figure 4.3: Heatmaps showing the improvement in average precision score over zero-shot performance for all tested combinations of training and evaluation datasets. For each combination, fine-tuning was performed three times from different random initialisations; larger numbers in each cell show the mean result and small, lower numbers give the standard deviation.

if applying cross-validation. We make an exception to this approach for the smallest training dataset (review G) which we believe is too small to take representative training and validation splits; we therefore use the full set for training and do not run an in-review experiment for it.

As seen in the previous section for the in-review setting, increasing the size of the training dataset generally produces better results. We expect this trend to also apply in the cross-review setting, particularly since larger datasets may contain a greater diversity of examples, favouring more general learning. So that we can separate the effects of dataset size from the effects of other properties of the review tasks, we train two versions of each adapter: one trained on the training split described above, which varies in size between datasets, and one trained on a truncated set of 196 items. We pick this size as it is the largest that we can obtain across all training datasets.

The results for the two training set types are shown in Figure 4.3. For each combination of training and evaluation reviews, we report on the difference between the average precision score obtained after fine-tuning and that obtained for zero-shot evaluation of the base model on the target review. We refer to this measure as ΔAP henceforth. We find that while more training data does help, dataset size by no means determines the amount of positive transfer that is gained from a particular review. Review A's balanced

training set is more than five times larger than that of any other review, and more than ten times larger than the truncated training sets. While the adapter trained on the whole of the review A training set does perform better than the one trained on the truncated dataset (the average ΔAP over target reviews is 0.023 versus 0.007), the adapters trained on reviews C, D and E achieve comparable performance despite their much smaller dataset sizes.

On the whole, cross-review training is generally helpful, giving a positive ΔAP value for 36 of the 49 tested review combinations with truncated datasets. Yet the specific combination of reviews involved can make a large difference to the result, with the mean ΔAP values per review pair varying from -0.11 to $+0.09$. There is no obvious pattern to these differences; for example, training on review B does not transfer well to review H, despite both reviews targetting empirical studies on human adaptation measures to natural hazards, and instead transfers much better to review D which is about point-of-care medical tests.

4.4 Multi-review fine-tuning

We now consider methods to incorporate learning from several training reviews at once to improve performance on a target review. As covered in Chapter 2, combining training from multiple tasks, through either model merging or MTL, can be an effective way to improve a model's generalisation to new tasks. Motivated by the results from the last section, which showed that one can obtain substantial positive transfer to a target review if the training review is chosen wisely, a possible approach to our task generalisation problem is to train a library of adapters, each corresponding to a review for which we have training data, and, at test time, to adapt the base model with the merged set of adapters that are likely to be most effective on the target review. This suggested approach relies on two assumptions:

1. That we can predict the ΔAP value of a merged adapter on a particular review from the aggregate ΔAP values of its constituent adapters on that review.

2. That we can predict the ΔAP of an adapter trained on one review and evaluated on another, based on properties of those reviews.

This section investigates the first assumption, which is essentially a question of how successfully we can merge adapters while retaining and composing their individual

skills. We compare two popular methods of adapter merging, AriMerge [62] and TIES [60] (described in Chapter 2). We also compare the proposed adapter-merging approach to the alternative of MTL fine-tuning across all datasets, while assuming that the second assumption can be fulfilled (we examine this in the following section). MTL has the disadvantage that it does not allow adaptive selection of training data at test time depending on the target review, but it more directly encourages the model to learn representations that apply generally across reviews due to the use of a mixed task optimisation objective, hence usually performing better than a merged set of adapters trained on the same tasks [60]. It is therefore an open question whether our task generalisation problem will benefit more from MTL fine-tuning on all available review datasets or from a merge of well-selected review-specific adapters.

4.4.1 Evaluation of adapter merging methods

In order to evaluate how well adapter skills are compounded through merging, we make use of the previously trained review-specific adapters. Taking one adapter fine-tuned on each review, we create a merged adapter for every possible combination of three of these adapters, and evaluate each merged adapter on two reviews (B and D). For each target review, we do not evaluate combinations including the adapter trained on that review, so this effectively gives $\binom{6}{3} = 20$ adapter combinations to evaluate on each review. For each combination, we test merges using TIES with varying values of k, as well as with AriMerge. While [62] only defines AriMerge in terms of combinations of two adapters, we make the natural extension to three adapters by weighting each set of parameters by $\frac{1}{3}$ and summing parameter-wise. Figure 4.4 shows, for each adapter evaluation, the performance of the merged adapter against the average performance of the constituent adapters.

The results broadly confirm the assumption that a merged adapter's performance correlates strongly with the performance of its constituents, but shows an interesting difference between the different merging methods. When using TIES, merged adapter performance very closely matches the mean performance of constituents, while AriMerge appears to show a stronger compounding effect, giving an adapter that performs better than the average when this average is positive and worse when it is negative. Out of the adapter combinations where all adapters have a positive ΔAP, the merged adapter performs better than any of the constituents on 60% of occasions when merging with AriMerge, compared to only 15% when using TIES and $k = 0.2$ (the recommended

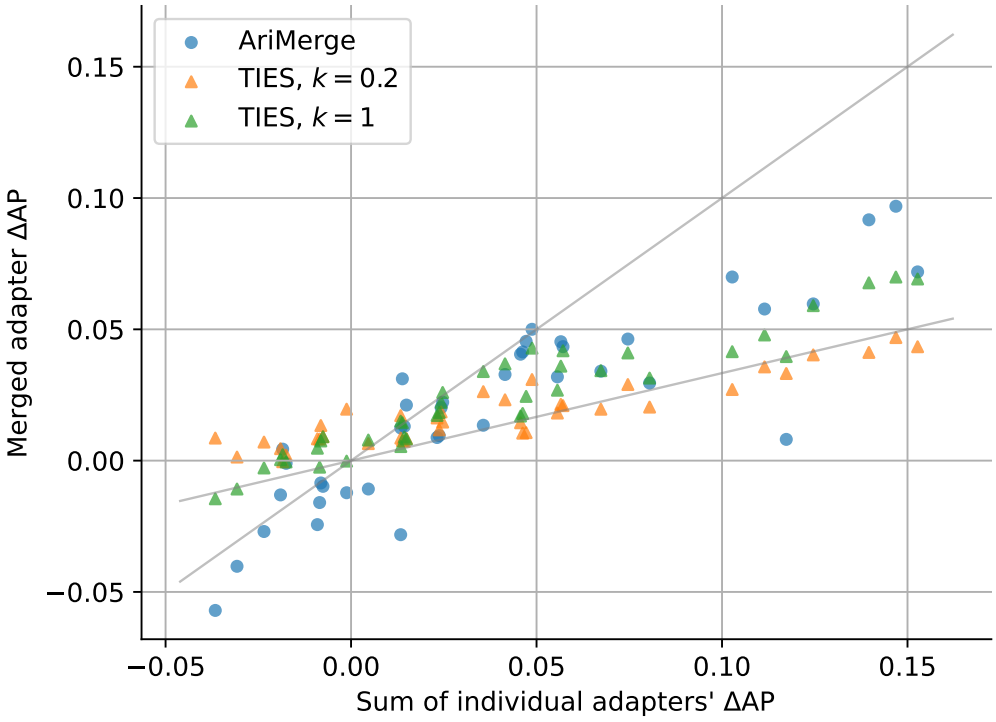

Figure 4.4: Comparison of the individual and merged performance of trios of adapters. Performance is measured as the change in average precision score that results from adding the adapter to the base model (ΔAP). Grey lines mark where the merged score is equal to the sum of the individual scores ($\Delta AP_{merged} = \Delta AP_{sum}$) and to the mean of the individual scores ($\Delta AP_{merged} = \Delta AP_{mean}$). TIES with $k = 0.5$ was also tested, giving very similar results to $k = 1$

value from [60]) or 35% with $k = 1$ (i.e., no pruning of adapter weights).

These findings align well with the fact that AriMerge has been shown to aggregate the individual skills of constituent adapters [62]. Given that each of our trained adapters must learn to identify a specific set of features corresponding to the inclusion criteria of a particular review, one can expect that a merged adapter that aggregates the skills of several adapters trained on different reviews will be capable of identifying the union of all these features, allowing it to perform better on a new review than any of the original adapters alone. On the other hand, the primary aim of TIES is to avoid interference between adapters by pruning parameter values that are in conflict with the majority and averaging, effectively leading to an adapter that captures the shared properties of the individual adapters, rather than their individual strengths. Furthermore, using a lower k

value results in a greater proportion of parameters being trimmed to 0, which naturally results in merged ΔAP values closer to 0. Due to the fact that the tasks in our setting are qualitatively very similar, it is likely that there is very little interference between trained adapters to begin with, explaining why applying TIES brings little benefit.

4.4.2 Comparing merged adapters to MTL

The findings above indicate that using AriMerge to combine adapters that give strong performance individually on a target review will produce a merged adapter that very likely performs better than their average, and likely performs better than any of those individual constituents too. We note that the former, weaker result may still be useful without the latter due to the fact any prediction of an adapter's target review performance will include some degree of error. Merging a number of probably-strong adapters allows us to reduce the variance of resulting merged adapter performance, relative to that of a single adapter. However, as noted previously, whether such an approach will perform better than MTL using all available review tasks is an open question, which we now investigate. Using the reviews A–G, we hold out one review and use the rest as training data to test a number of alternative methods. As in Section 4.3 we use equally-sized training sets with an even class balance for all reviews, thus eliminating the influence of dataset size and class balance from our results.

The first of the tested methods is the **MTL** approach of fine-tuning a single LoRA adapter on the combined dataset of all training reviews. We ensure that each training batch consists of exactly one example from each review, thus making the optimised loss function a constant mixture of the task-specific losses and keeping learning stable.

We compare this to an '**Oracle Selection**' approach, where we use the target review evaluation scores of adapters trained on each of the training reviews individually to select the best adapters, and use either the top adapter individually, a merge of the top 3 adapters, or a merge of all adapters with a positive ΔAP (with merges carried out using AriMerge). While the evaluation performance of each adapter would not be available to us in a real test scenario, this represents an upper bound on the performance we may expect to get if we can devise a good strategy for adapter selection. If this method does not produce good results, we should not try to identify such a strategy and opt for an MTL approach instead.

We note that using Oracle Selection may result in picking adapters that perform best purely as a result of a 'lucky' initialisation that is particularly well suited to the target

dataset, rather than as a result of inherent properties of the training and review tasks. We therefore also consider an '**Informed Selection**' approach, which is similar to the Oracle approach, but adapters are ranked based on the performance of *other* trained adapters (i.e., starting with a different random initialisation) trained and evaluated on the same pair of reviews. If this approach proves effective, this would indicate that it is possible to make a good selection of adapters based only on knowledge of the tasks on which they were trained.

Target review	Baseline	MTL	Oracle Selection			Informed Selection
			Top 1	Top 3	All positive	Top 3
A	0.819	0.814	0.835	0.837	**0.839**	0.835
B	0.572	0.592	0.617	0.623	**0.624**	0.612
C	0.550	0.626	0.602	0.627	**0.630**	0.604
D	0.580	0.649	0.666	**0.670**	0.664	0.673
E	0.565	0.590	**0.604**	0.601	0.580	0.604
F	0.283	**0.350**	0.348	0.346	0.339	0.338
G	0.613	0.663	0.659	**0.666**	0.663	0.664
Average	0.569	0.612	0.619	**0.624**	0.620	0.619

Table 4.2: Average precision scores obtained from fine-tuning models on a range of reviews and evaluating on an unseen target review, using the different methods described in the text. The baseline column shows the zero-shot performance of the base model. Each reported value is the mean result of three different training runs using different random adapter initialisations.

The results for the different methods across all reviews are provided in Table 4.2. We firstly observe that all the tested methods substantially improve the base model's performance on an unseen review and that differences between the methods are slight, with the average ΔAP over all reviews ranging from $+0.043$ for MTL to $+0.055$ for oracle selection of the top 3 adapters. Within the strategies that use Oracle Selection, neither is consistently best across reviews, but since the top 3 strategy obtains the highest average score, we use only this method within the Informed Selection approach. Impressively, Informed Selection obtains a better score than MTL for 5 of the 8 target reviews and a better average score. While the difference is very small, the fact that the scores are comparable is in itself encouraging, since merging adapters has the additional

practical advantages outlined previously of very easily allowing iterative and distributed training.

Finally, to confirm that the selection of adapters to merge is important, we also repeat the tests above using the Informed Selection approach to select the *worst* ranked three adapters, finding that the average performance across reviews drops to 0.557, i.e. below zero-shot performance. This is evidence that there exist properties of the review tasks that have a large effect on the cross-review performance of models on those tasks.

4.5 Adapter selection

Throughout the previous section, our experiments made use of evaluation scores of adapters on the target review in order to rank and select adapters. In reality, we would not have access to these scores, since the model would be used to classify papers for a target review for which we do not already have labels. In this section, we consider methods that can be used to predict the performance of an expert adapter on an target review in either a zero-shot or few-shot fashion, with the few-shot setting allowing access to 50 target labelled items from the target review as described in Section 3.1.

4.5.1 Adapter ranking methods

Relevant methods from prior research in identifying tasks for transfer learning were briefly introduced in section 2.2.2; our methods are based on those works.

Firstly, we consider the approach introduced by AdapterSoup [13] of using **domain similarity** between pairs of datasets as a predictor of transferability between them. Specifically, this method consists of taking a sample of sentences from the input texts of each dataset, encoding them using a sentence-level encoder model, and taking the mean of the embeddings to obtain a vector representation of each dataset's domain. The similarity between datasets is then calculated as the cosine similarity between these representations, and can be used to rank adapters corresponding to the training datasets by the datasets' similarity to the target dataset.

To implement this method, we concatenate the sentences of all abstracts within a review dataset (splitting the abstracts into sentences with NLTK [8]), randomly sample 1000 of the sentences and encode them using the Sentence-BERT [48] model. We use the pretrained all-mpnet-base-v2[1] Sentence-BERT model, as it is the same model

[1] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

used by AdapterSoup, and gives strong performance as a sentence encoder while being efficient enough to encode all the sampled sentences within a few seconds on our GPU setup.

We note, however, an important shortcoming of this method of representing our tasks: it only captures features of their respective *domains*, and ignores differences between their respective labelling functions, i.e. their review inclusion criteria. Intuitively, we would expect overlaps between review criteria to be an important factor in determining transferability, since the skill of a model in identifying a particular criterion could be re-used between the tasks. Therefore, we also implement a variation of this method, **criteria similarity**, where we instead use an encoding of the SR inclusion criteria as the representation of each dataset. We experiment with two options to create these encodings: 1) encoding the full criteria of each review using the same model as above, 2) passing the criteria and task instructions through the same base model we use for the abstract screening task (Llama 3 8B) and taking the output at an intermediate layer of the model, averaging over token positions. The motivation for this second alternative is that we would expect it to result in encodings that capture features which are particularly relevant to the the target task and model.

In the few-shot setting, an obvious potential method is to simply evaluate each available adapter on the labelled data, and use this as an estimate of the performance on the full dataset. We refer to this method as **few-shot evaluation**. It has the disadvantage of requiring 50 predictions for every available adapter every time the model is to be used on a new target SR. If the system was to be scaled to include adapters for many more reviews than we use here, this could become an infeasible approach due to the computational cost. Nevertheless, with 7 trained modules this only entails evaluating 350 additional items per target review, which is not unreasonable since our largest review dataset is twice this size.

As mentioned in Section 2.2.2, another approach suggested in past work [63] is to use the vectorised parameters of a PEFT module fine-tuned on a particular task as an embedding of the task and use this to predict transferability. In our case, this means we must fine-tune a new adapter on the few-shot dataset of the target review and find the existing adapters whose parameters have the highest cosine similarity with those of the new adapter. We call this **adapter similarity**. Compared to few-shot evaluation, this method scales much better with the number of trained adapters, since only a cosine similarity calculation between reasonably small LoRA modules is required for each one. Furthermore, while the requirement to fine-tune on each new target dataset may

seem like a heavy additional cost, this is something that would likely be done regardless whenever few-shot data is available in order to improve the performance of the model, as we will explore in the final section of experiments.

To test each of these similarity metrics as methods for ranking adapters, we compute each metric between all of our datasets and compare the resulting values with the ΔAP scores reported in Section 4.3 for the degree of transfer between the corresponding datasets. We note that the previously reported transfer between reviews was not always commutative, i.e. for some pairs of reviews, the ΔAP score was quite different depending on which of the two was the target dataset. This of course limits the extent to which any of the proposed similarity metrics, which are commutative, can successfully predict the transfer.

We compare the similarity and transferability scores by computing:

- **Avg. top 3 hits**: How many of the actual top 3 experts for a given target dataset are also within the top 3 selected by the similarity metric, averaged over target reviews.
- **Avg.** ρ: Spearman's rank correlation coefficient between the rankings induced by the two scores over the training datasets for a given target dataset, averaged over all target datasets
- **Overall** ρ: Spearman's rank correlation coefficient calculated between the rankings over all training/target dataset combinations.

We report this range of scores since each has its own disadvantages. Score a) is easily interpreted, but does not take into account the full ranking, while b) and c) do. Unfortunately, since there are only six experts to choose from (not counting the one trained for the target review), a per-target-review correlation coefficient is unreliable and gives very variable results. This is why we also calculate c), though we note that this risks introducing the target dataset as a confounding variable that creates a spurious correlation.

4.5.2 Results

The results of evaluating the adapter ranking methods described above on our set of adapters is shown in Table 4.3

The results suggest that none of the tested similarity metrics produce particularly effective rankings of the adapters. The domain similarity metric produces the best

Ranking method	Avg. ρ	Overall ρ	Avg. top 3 hits
Domain similarity	**0.148**	**0.274**	**1.75**
Criteria similarity (S-BERT)	-0.035	-0.157	1.63
Criteria similarity (Llama layer 4)	-0.189	-0.304	1.38
Criteria similarity (Llama layer 32)	0.145	0.136	**1.75**
Adapter similarity (fully trained)	0.069	0.033	1.71

Table 4.3: Evaluation of methods to predict the ranking of adapters in terms of their transferability to a target review. A description of the evaluation scores in the columns is given in the text.

scores, but only has a correlation coefficient of $\rho = 0.148$ averaged over target reviews, which typically indicates no correlation or a very weak one.

As previously mentioned, the test adapter similarity method would need to use an adapter for the target task that has been trained only on the few-shot dataset. However, as a preliminary test, we first checked whether this method can accurately predict transferability between the adapters trained on larger review datasets that were used in previous experiments. In fact, we find that transferability from one review to another shows no correlation with the cosine similarity between adapters trained on the two reviews. Investigating further, we note that the cosine similarities between our adapters depend very little on the training dataset: all pairs of adapters have a similarity in the range 0.88–0.90 if they were trained from the same random initialisation, and below 0.01 if the random initialisations were different. A possible explanation of this finding is that despite the low rank of the LoRA matrices, the adapters are still greatly overparametrised for their task, and as such only a small fraction of the module parameters are significantly updated from their initialised values during training. To eliminate the effect of the parameter initialisation, we also explored using the difference between the final and initial parameter values of the adapters as the task embeddings, which provided a more diverse range of similarity values (0.11–0.31). We used these embeddings to obtain the scores of the adapter similarity method shown in Table 4.3. Since this still performed poorly, we did not try to compute the embeddings using only the few-shot datasets, since this would surely not have been more effective.

Finally, to evaluate the few-shot evaluation method of adapter ranking, we can artificially generate many ‘experiments’ from each single adapter evaluation: we randomly partition the full set of predictions made by a given adapter on a given dataset into a

few-shot set and a much larger 'unseen' set, and look at the correlation between the average precision scores obtained on over these two sets for 100 randomly generated partitions for each adapter evaluation of each dataset. We find that the score over the 50-item few-shot set provides almost no information on the score over the larger set, with near-zero correlations between the two. This is related to our choice of average precision as the target evaluation metric, which is particularly difficult to estimate from a small sample. Unlike an accuracy estimate, where every sampled item has an equal influence, an average precision score can be disproportionally influenced by a single negative class item if it is given a higher positive probability by the model than most positive class items in the sample, which results in a high variance when the score is estimated from a small sample.

4.6 End-to-end comparison of methods

In the previous sections we separated the concerns of how to select optimal datasets to train on for task generalisation (Section 4.5) and how to best combine learning from a given collection of datasets (Sections 4.3–4.4). It is now natural to test the most successful techniques in combination on the full problem of performing abstract screening with zero or very limited labelled data (50 items, with at least 5 positive) from the target review, making use of available labelled data corresponding to other reviews.

While we did not find a particularly effective indicator of transferability between review tasks, domain similarity obtained the best results out of the tested metrics, regardless of whether any labels were available for the target SR data. We therefore test a method of combining the best three review-specific adapters via AriMerge as in section 4.4, but this time selecting the top adapters based on the degree of similarity between the paper abstracts of their corresponding training datasets and those of the target dataset. We refer to this overall strategy as '**expert merging**'. We compare this approach again with the MTL approach; while this was seen to slightly underperform expert merging when the expert modules were selected based on target dataset performance, this may no longer be the case now that expert selection is based on domain similarity.

In the zero-shot setting, we simply evaluate on the target dataset using the three selected adapters, merged by AriMerge, and compare with the MTL results of Section 4.4. In the few-shot setting, we make use of the target review data to additionally fine-tune a new LoRA adapter, trained on top of the model that is already adapted by either the MTL or review-specific adapters. We use the same hyperparameters that

proved most effective when fine-tuning on the same amount of data in Section 4.2; while the optimal values may in theory vary between reviews and between the MTL and merged experts cases, we do not have access to validation data to make these choices individually. In order to evaluate the effect of learning from other reviews, we compare these methods to the base model fine-tuned with the few-shot data in the few-shot setting, and the base model evaluated in a zero-shot manner in the zero-shot setting. The results are shown in Figure 4.4.

Target review	Zero-shot			Few-shot		
	Baseline	MTL	Expert merge	Baseline	MTL	Expert merge
A	0.565	**0.590**	0.580	0.595	0.600	**0.636**
B	0.819	0.814	**0.830**	0.850	0.835	**0.883**
C	0.550	**0.626**	0.612	0.639	**0.658**	0.657
D	0.580	**0.649**	0.558	0.575	**0.635**	0.594
E	0.572	0.592	**0.593**	0.667	0.654	**0.693**
F	0.283	**0.350**	0.314	**0.378**	0.377	0.371
G	0.613	**0.663**	0.659	0.695	0.740	**0.789**
Average	0.569	**0.612**	0.592	0.628	0.643	**0.660**

Table 4.4: Comparison of results for the MTL and expert merging to new review generalisation, in both zero- and few-shot settings. Each row corresponds to a model being trained using data from all reviews other than the indicated target review. 'Baseline' corresponds to base model evaluation under 'Zero-shot' and to the base model fine-tuned with few-shot data under 'Few-shot'.

In the zero-shot case, using the sentence embedding similarity metric for adapter selection in place of target review evaluation scores leads to a drop in performance, resulting in lower scores than MTL on 5 of the 7 target reviews and on average by a difference of 0.02. This is as expected expected given the weak performance of the similarity metric in predicting transferability (Section 4.5). Nevertheless, this score is better than the baseline unadapted model score by a similar margin.

Given these observations, it is surprising that the trend reverses for the few-shot setting, with the expert merging approach now obtaining an average 0.017 higher than MTL. There is nothing different about the two approaches between the two settings, other than the additional fine-tuning with the same few-shot data from the target

review. Yet it appears that this few-shot fine-tuning provides a far bigger performance improvement to the merged experts model (+0.068) than to the MTL model (+0.031). We speculate that this difference may be owing to the different ways the underlying MTL and merged expert models are created. In the MTL case, the model is optimised with a multi-task objective, so it is well prepared to deal with new tasks (within the general abstract screening meta-task) even in the zero-shot setting. In the expert merging case, all previous training was task-specific, and while the merging of several modules equips the model with the skills necessary to handle a range of reviews, a sub-optimal merge may prevent these skills from being effectively utilised in the zero-shot case, while some additional fine-tuning on the target task allows the model to adjust to effectively utilise the representations gained from the review-specific merged modules.

4.7 Detailed evaluation of best model

We conclude our experiments with a more detailed analysis of the performance of our expert merging approach in the few-shot setting, comparing this to the performance of GPT-4o prompted with in-context examples, as this may represent a practical alternative in a real-world scenario. We carry out this test on dataset I, which has been held out of all other experiments such that our model design has not been unfairly optimised towards it. Due to the per-token cost of running model evaluations with the OpenAI API, we only use a sample of 300 randomly selected items from this set, including 67 positive class items.

Figure 4.5 shows the trade-off between precision and recall for all possible classification thresholds, for both models. Impressively, with our expert-based fine-tuning regime, Llama 3 8B performs better than the much more powerful GPT-4o model at all classification thresholds that achieve a recall of at least 0.2. Llama achieves an average precision score of 0.57 compared to GPT-4o's 0.47, and its precision value when recall = 0.95 is 0.31 compared to GPT-4o's 0.22. With a classification threshold of 0.5, the F_1 score of Llama is 0.47, while that of GPT-4o is 0.46.

It is very difficult to make judgements about the models' general abstract screening ability from these results due to the very limited size and variety of the test dataset. As is clear from our experiments on different review datasets, different screening tasks vary widely in their difficulty depending on their inclusion criteria and on the ratio of relevant papers in their screening pool. Nevertheless, the results are good evidence that fine-tuning a model with data from past SRs using a merged experts approach is

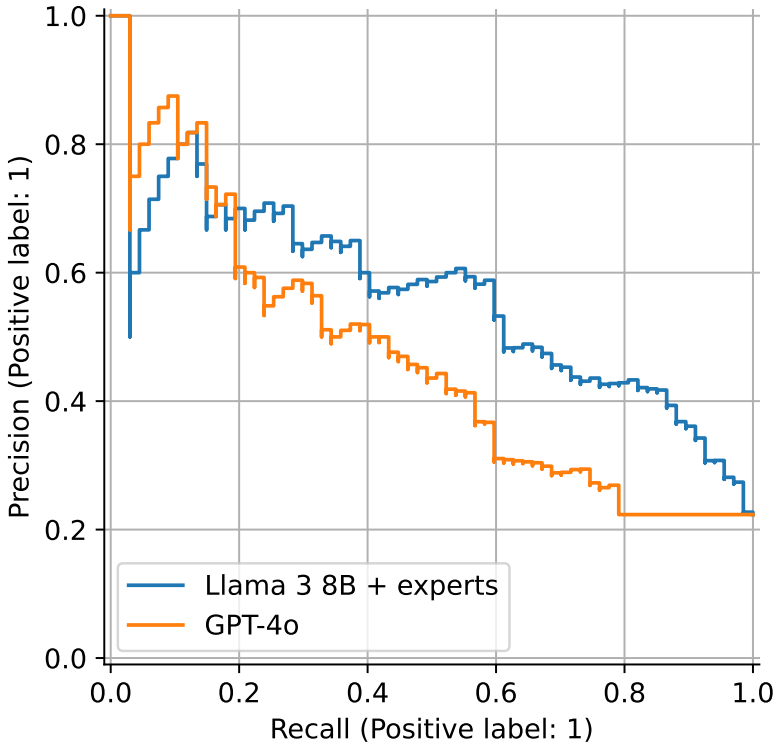

Figure 4.5: Precision and recall values achieved across different possible classification thresholds by the GPT-4o model, provided with 4 in-context examples, and the Llama 3 8B model adapted with review-specific experts as described in the text. The models are evaluated on a multi-review dataset.

an effective way of improving generalisation to new SRs, allowing a relatively small model such as Llama 3 8B to surpass the performance of GPT-4o.

Chapter 5

Discussion

In light of our experimental results, we now return to the key questions of the project, regarding the effectiveness of fine-tuning a PLM to the abstract screening task for a particular review making use of very little labelled data from the target review and abundant data from other reviews. We consider our results within the broader literature on task and domain generalisation and model merging, discuss how the findings could translate into real impacts on the SR process while also noting limitations and possible extensions of our work.

5.1 Comparison to prior research

As discussed in Chapter 2, our work has combined previous insights from research on model merging and on cross-task transfer to develop an approach for task-level generalisation based on combining task-specific experts. Prior work focusing on task generalisation through model merging does not generally compare this to approach to MTL as an alternative, presumably due to the assumption that MTL would always be expected to deliver better results due to directly optimising for multiple tasks, and that an expert merging approach would be used in situations where multi-task training is not possible, e.g. due to lack of concurrent access to training data from all the training tasks [29]. Indeed, [60], who do compare their model merging approach with multi-task learning, report far lower performance for their own method. It is therefore highly surprising that in our experiments, we were able to surpass the performance of MTL by selectively merging task-specific experts, even when the the selection process relied on the fairly weak predictive metric of input data sentence similarity.

A plausible explanation for this finding is that our problem scenario involved tasks

that are particularly similar to each other, and could in fact be considered different variants of a single general abstract screening task. This means there is less of a risk of interference when merging models than there would be if the tasks were more different, and as a result the TIES-Merging method to mitigate parameter conflicts does not bring benefits, as seen in Section 4.4. Another factor that likely contributed towards the lack of difficulties when merging was that, as suggested in Section 4.5, our adapters appeared to be greatly overparametrised for their fine-tuning task, leading to very few parameters changing from their initialised state. The greater the redundancy in parameters, the smaller the risk of important parameters coming into conflict when different modules are merged.

On the other hand, our attempts to identify factors that determine transferability between tasks proved less sucessful than in prior work. The metric that proved to be the best predictor of transferability out of those we tested was the sentence embedding cosine similarity approach used in AdapterSoup [13], but its performance in ranking adapters was still disappointingly low. We would have expected the inclusion criteria of different tasks to play an important role in determining transferability, as reviews that share parts of their inclusion criteria should intuitively benefit more from shared skills, but we were not able to identify such a pattern. Future work could test the use of more sophisticated recent techniques to find strong expert module combinations in a few-shot setting, such as [29, 28, 20], or the zero-shot method of [42], which we were not able to implement and test in the time available for this project.

5.2 Practical implications of results

The end goal of an automated abstract screening system is to reduce time spent by human experts manually screening abstracts, while ensuring that papers relevant to a SR do not get missed out. It is therefore instructive to interpret our results in terms of the rough time savings they could enable. Supposing that the classification threshold of a particular screening system is always set such that the recall of the system is fixed at some value, R, then an increase in the precision of the system, P, entails a drop in the number of false positives predicted by the system. More specifically, one can show that:

$$F'_+ - F_+ = NR\left(\frac{P - P'}{P'P}\right) \tag{5.1}$$

where F_+ and F'_+ are the total number of false positives predicted by the system

on some dataset before and after some change to the system, respectively, N is the total number of true positives in the dataset, R is the constant recall value, and P and P' are the precision values before and after the change. Hence, let us take as an example the finding from Section 4.2 that fine-tuning on a dataset of 100 (balanced) labelled examples led to an increase 0.44 to 0.57 in the precision achieved by the model at a recall level of 0.95 or above. For the given dataset, which had 1248 positive examples (beside the 25 used for training) such an improvement would correspond to $1248 \cdot 0.95 \left(\frac{0.57-0.44}{0.57 \cdot 0.44}\right) \approx 615$ fewer false positive predictions. As such, the time spent manually labelling 100 paper abstracts to train the model is clearly worthwhile given the resulting reduction in irrelevant papers that would need to be read and discarded in the following stages of the SR. However, this is an illustrative example, and the trade-off between time spent curating target SR data versus time spent through automation on carrying out that SR will heavily depend on characteristics of the particular review, such as the complexity of the inclusion criteria and the number of abstracts that must be screened.

While our results are not sufficient to verify the current suitability of fine-tuned PLMs for SR abstract screening, we hope that they may motivate further research into this opportunity. Given that we have shown that an 8-billion-parameter model can greatly benefit from training on a small handful of existing SRs, it seems highly likely that far more capable models could be developed by extending the methods we have put forward to more powerful models and larger datasets combining a wider range of SRs.

5.3 Limitations

A major limitation in our experimental methods was the lack of a broad, diverse range of of different review datasets on which to train and evaluate methods. Typically, development of multi-task and task-generalisation methods is performed using a large set of different tasks which are divided into a disjoint sets of training tasks, development tasks and test tasks. Having only 9 review datasets available, we were only able to hold out a single dataset for evaluation. This means that our final results may have been determined by the specific properties of this review task and not be representative of the models' performance on abstract screening tasks in general. Moreover, having only 8 datasets on which to investigate transferability between reviews was likely part of the reason why it was difficult to identify consistent patterns in this area. Given our positive preliminary results regarding the effectiveness of multi-review training for abstract

screening, we would encourage future efforts to develop large-scale, widely-accessible datasets of SR screening decisions, on which our preliminary findings could be further developed.

Furthermore, we made two important simplifications in our treatment of the task generalisation problem by assuming that all training datasets were of equal size and internally balanced between classes. While this was useful in order to allow us to focus on inherent properties of the review tasks and domains, both dataset size and class balance are likely to be important factors to that should be considered alongside other dataset and expert properties when selecting which experts should be selected for a given target task. Related to the issue of class balance, we have also avoided delving into the problem of model calibration by seeking to improve general model performance across all possible classification thresholds and assuming that in a downstream test scenario, there will be a suitable method for identifying the classification threshold that obtains the required level of recall. In fact, this would be difficult to do without access to a large amount of target review data, and data from other reviews is likely of limited use given the large differences in class balances between reviews. This issue is unlikely to be easily solvable, since neural classifier calibration is an open research problem, with methods always requiring held out validation data [23, 51].

Chapter 6

Conclusions

The contributions of this work have been twofold. On one hand, we have demonstrated practical approaches that make it possible to boost the performance of a PLM on SR abstract screening using data from prior SRs and possibly a small amount of labelled data from the target SR. While we have used the relatively small Llama 3 8B model in our experiments, it is likely that the results would also apply to models of much larger sizes and similar architecture, creating an opportunity for even stronger systems. On the other hand, we have contributed results relevant to more general research in task-level generalisation, showing that an approach of merging task-specific expert LoRA modules performs far better than expected from prior research, at least in our particular problem setting involving closely related tasks that are variants of a single meta-task.

However, as noted in the previous section, the reported experiments were limited by the quantity and variety of data available, and as such the results cannot be interpreted as conclusive without further validation. Moreover, while our results suggest that fine-tuned PLMs are a promising and powerful tool for accelerating the progress of SRs and reducing the countless hours spent by expert academics sifting through irrelevant literature, it is critical to examine the performance of any such system from a wide range of different perspectives, including potential blind spots and biases of the model, before considering any real-world practical application. We envisage that in a similar fashion to currently in-use SR automation systems [43, 54], PLMs currently hold most promise as tools providing increased efficiency within human-in-the-loop systems, which limit the risk of costly automated mistakes and provide an opportunity for continual learning and improved calibration using feedback from a human expert.

Bibliography

[1] Alessandro Achille et al. “Task2Vec: Task Embedding for Meta-Learning”. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019, pp. 6430–6439. URL: https://openaccess.thecvf.com/content_ICCV_2019/html/Achille_Task2Vec_Task_Embedding_for_Meta-Learning_ICCV_2019_paper.html (visited on 08/17/2024).

[2] AI@Meta. *Introducing Meta Llama 3: The Most Capable Openly Available LLM to Date*. Meta AI. URL: https://ai.meta.com/blog/meta-llama-3/ (visited on 07/29/2024).

[3] AI@Meta. “Llama 3 Model Card”. In: (2024). URL: https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

[4] Jason Ansel et al. “PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation”. In: *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*. ASPLOS ’24: 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2. La Jolla CA USA: ACM, Apr. 27, 2024, pp. 929–947. ISBN: 9798400703850. DOI: 10.1145/3620665.3640366. URL: https://dl.acm.org/doi/10.1145/3620665.3640366 (visited on 07/28/2024).

[5] Alexandra Bannach-Brown et al. “Machine Learning Algorithms for Systematic Review: Reducing Workload in a Preclinical Review of Animal Studies and Reducing Human Screening Error”. In: *Systematic Reviews* 8.1 (Jan. 15, 2019), p. 23. ISSN: 2046-4053. DOI: 10.1186/s13643-019-0942-7. URL: https://doi.org/10.1186/s13643-019-0942-7 (visited on 08/13/2024).

[6] Hilda Bastian, Paul Glasziou, and Iain Chalmers. “Seventy-Five Trials and Eleven Systematic Reviews a Day: How Will We Ever Keep Up?” In: *PLOS Medicine* 7.9

(Sept. 21, 2010), e1000326. ISSN: 1549-1676. DOI: 10.1371/journal.pmed.1000326. URL: https://journals.plos.org/plosmedicine/article?id=10.1371/journal.pmed.1000326 (visited on 07/27/2024).

[7] Magdalena Biesialska, Katarzyna Biesialska, and Marta R. Costa-jussà. *Continual Lifelong Learning in Natural Language Processing: A Survey*. Dec. 17, 2020. DOI: 10.18653/v1/2020.coling-main.574. arXiv: 2012.09823 [cs]. URL: http://arxiv.org/abs/2012.09823 (visited on 08/21/2024). Pre-published.

[8] Steven Bird, Edward Loper, and Ewan Klein. *Natural Language Processing with Python*. O'Reilly Media Inc., 2009. URL: https://www.nltk.org/ (visited on 08/11/2024).

[9] Rohit Borah et al. "Analysis of the Time and Workers Needed to Conduct Systematic Reviews of Medical Interventions Using Data from the PROSPERO Registry". In: *BMJ Open* 7.2 (Feb. 1, 2017), e012545. ISSN: 2044-6055, 2044-6055. DOI: 10.1136/bmjopen-2016-012545. pmid: 28242767. URL: https://bmjopen.bmj.com/content/7/2/e012545 (visited on 04/15/2024).

[10] Tom Brown et al. "Language Models Are Few-Shot Learners". In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. URL: https://papers.nips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html (visited on 07/29/2024).

[11] Weilin Cai et al. *A Survey on Mixture of Experts*. Aug. 8, 2024. arXiv: 2407.06204 [cs]. URL: http://arxiv.org/abs/2407.06204 (visited on 08/21/2024). Pre-published.

[12] Guanzheng Chen et al. "Revisiting Parameter-Efficient Tuning: Are We Really There Yet?" In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2022. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 2612–2626. DOI: 10.18653/v1/2022.emnlp-main.168. URL: https://aclanthology.org/2022.emnlp-main.168 (visited on 07/28/2024).

[13] Alexandra Chronopoulou et al. *AdapterSoup: Weight Averaging to Improve Generalization of Pretrained Language Models*. Mar. 28, 2023. DOI: 10.48550/

arXiv.2302.07027. arXiv: 2302.07027 [cs]. URL: http://arxiv.org/abs/2302.07027 (visited on 03/30/2024). Pre-published.

[14] Michael Crawshaw. *Multi-Task Learning with Deep Neural Networks: A Survey*. Sept. 10, 2020. arXiv: 2009.09796 [cs, stat]. URL: http://arxiv.org/abs/2009.09796 (visited on 07/08/2024). Pre-published.

[15] CRD. *Systematic Reviews: CRD's Guidance for Undertaking Reviews in Health Care*. York: Centre for Reviews and Dissemination, University of York, Jan. 2009. ISBN: 978-1-900640-47-3.

[16] Nico Daheim et al. "Model Merging by Uncertainty-Based Gradient Matching". In: The Twelfth International Conference on Learning Representations. Oct. 13, 2023. URL: https://openreview.net/forum?id=D7KJmfEDQP (visited on 04/02/2024).

[17] José de la Torre-López, Aurora Ramírez, and José Raúl Romero. "Artificial Intelligence to Automate the Systematic Review of Scientific Literature". In: *Computing* 105.10 (Oct. 1, 2023), pp. 2171–2194. ISSN: 1436-5057. DOI: 10.1007/s00607-023-01181-x. URL: https://doi.org/10.1007/s00607-023-01181-x (visited on 04/15/2024).

[18] Fabio Dennstädt et al. "Title and Abstract Screening for Literature Reviews Using Large Language Models: An Exploratory Study in the Biomedical Domain". In: *Systematic Reviews* 13.1 (June 15, 2024), p. 158. ISSN: 2046-4053. DOI: 10.1186/s13643-024-02575-4. URL: https://doi.org/10.1186/s13643-024-02575-4 (visited on 08/13/2024).

[19] Tim Dettmers et al. "QLoRA: Efficient Finetuning of Quantized LLMs". In: *Advances in Neural Information Processing Systems* 36 (Dec. 15, 2023), pp. 10088–10115. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/1feb87871436031bdc0f2beaa62a049b-Abstract-Conference.html (visited on 03/11/2024).

[20] Wenfeng Feng et al. "Mixture-of-LoRAs: An Efficient Multitask Tuning Method for Large Language Models". In: *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*. LREC-COLING 2024. Ed. by Nicoletta Calzolari et al. Torino, Italia: ELRA and ICCL, May 2024, pp. 11371–11380. URL: https://aclanthology.org/2024.lrec-main.994 (visited on 08/13/2024).

[21] Zihao Fu et al. “On the Effectiveness of Parameter-Efficient Fine-Tuning”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 37.11 (11 June 26, 2023), pp. 12799–12807. ISSN: 2374-3468. DOI: 10.1609/aaai.v37i11.26505. URL: https://ojs.aaai.org/index.php/AAAI/article/view/26505 (visited on 07/28/2024).

[22] Seraphina Goldfarb-Tarrant et al. “Scaling Systematic Literature Reviews with Machine Learning Pipelines”. In: *Proceedings of the First Workshop on Scholarly Document Processing*. Sdp 2020. Ed. by Muthu Kumar Chandrasekaran et al. Online: Association for Computational Linguistics, Nov. 2020, pp. 184–195. DOI: 10.18653/v1/2020.sdp-1.21. URL: https://aclanthology.org/2020.sdp-1.21 (visited on 04/14/2024).

[23] Chuan Guo et al. *On Calibration of Modern Neural Networks*. Aug. 3, 2017. arXiv: 1706.04599 [cs]. URL: http://arxiv.org/abs/1706.04599 (visited on 08/21/2024). Pre-published.

[24] Eddie Guo et al. “Automated Paper Screening for Clinical Reviews Using Large Language Models: Data Analysis Study”. In: *Journal of Medical Internet Research* 26.1 (Jan. 12, 2024), e48996. DOI: 10.2196/48996. URL: https://www.jmir.org/2024/1/e48996 (visited on 04/15/2024).

[25] Soufiane Hayou, Nikhil Ghosh, and Bin Yu. *LoRA+: Efficient Low Rank Adaptation of Large Models*. Feb. 19, 2024. arXiv: 2402.12354 [cs, stat]. URL: http://arxiv.org/abs/2402.12354 (visited on 06/21/2024). Pre-published.

[26] Kaiming He et al. “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification”. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015 IEEE International Conference on Computer Vision (ICCV). Santiago, Chile: IEEE, Dec. 2015, pp. 1026–1034. ISBN: 978-1-4673-8391-2. DOI: 10.1109/ICCV.2015.123. URL: http://ieeexplore.ieee.org/document/7410480/ (visited on 07/29/2024).

[27] Edward J. Hu et al. *LoRA: Low-Rank Adaptation of Large Language Models*. Oct. 16, 2021. arXiv: 2106.09685 [cs]. URL: http://arxiv.org/abs/2106.09685 (visited on 03/11/2024). Pre-published.

[28] Chengsong Huang et al. *LoraHub: Efficient Cross-Task Generalization via Dynamic LoRA Composition*. Jan. 18, 2024. DOI: 10.48550/arXiv.2307.13269.

arXiv: 2307.13269 [cs]. URL: http://arxiv.org/abs/2307.13269 (visited on 03/30/2024). Pre-published.

[29] Weisen Jiang et al. *BYOM: Building Your Own Multi-Task Model For Free*. Feb. 3, 2024. DOI: 10.48550/arXiv.2310.01886. arXiv: 2310.01886 [cs]. URL: http://arxiv.org/abs/2310.01886 (visited on 04/01/2024). Pre-published.

[30] Xisen Jin et al. "Dataless Knowledge Fusion by Merging Weights of Language Models". In: The Eleventh International Conference on Learning Representations. Sept. 29, 2022. URL: https://openreview.net/forum?id=FCnohuR6AnM (visited on 04/02/2024).

[31] Armand Joulin et al. *Bag of Tricks for Efficient Text Classification*. Aug. 9, 2016. arXiv: 1607.01759 [cs]. URL: http://arxiv.org/abs/1607.01759 (visited on 08/22/2024). Pre-published.

[32] Qusai Khraisha et al. "Can Large Language Models Replace Humans in Systematic Reviews? Evaluating GPT-4's Efficacy in Screening and Extracting Data from Peer-Reviewed and Grey Literature in Multiple Languages". In: *Research Synthesis Methods* 15.4 (2024), pp. 616–626. ISSN: 1759-2887. DOI: 10.1002/jrsm.1715. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/jrsm.1715 (visited on 08/13/2024).

[33] C Lefebvre et al. "Chapter 4: Searching for and Selecting Studies". In: *Cochrane Handbook for Systematic Reviews of Interventions*. version 6.4 (updated October 2023). Cochrane, 2023. URL: https://training.cochrane.org/handbook/current/chapter-04 (visited on 07/27/2024).

[34] Pengfei Liu et al. *Pre-Train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing*. July 28, 2021. arXiv: 2107.13586 [cs]. URL: http://arxiv.org/abs/2107.13586 (visited on 08/20/2024). Pre-published.

[35] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. Jan. 4, 2019. DOI: 10.48550/arXiv.1711.05101. arXiv: 1711.05101 [cs, math]. URL: http://arxiv.org/abs/1711.05101 (visited on 07/31/2024). Pre-published.

[36] Michael S. Matena and Colin A. Raffel. "Merging Models with Fisher-Weighted Averaging". In: *Advances in Neural Information Processing Systems* 35 (Dec. 6, 2022), pp. 17703–17716. URL: https://proceedings.neurips.cc/paper_

files/paper/2022/hash/70c26937fbf3d4600b69a129031b66ec-Abstract-Conference.html (visited on 04/02/2024).

[37] Michael McCloskey and Neal J. Cohen. "Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem". In: *Psychology of Learning and Motivation*. Ed. by Gordon H. Bower. Vol. 24. Academic Press, Jan. 1, 1989, pp. 109–165. DOI: 10.1016/S0079-7421(08)60536-8. URL: https://www.sciencedirect.com/science/article/pii/S0079742108605368 (visited on 08/13/2024).

[38] Francis Sahngun Nahm. "Receiver Operating Characteristic Curve: Overview and Practical Use for Clinicians". In: *Korean Journal of Anesthesiology* 75.1 (Feb. 2022), p. 25. DOI: 10.4097/kja.21209. pmid: 35124947. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8831439/ (visited on 08/21/2024).

[39] Alison O'Mara-Eves et al. "Using Text Mining for Study Identification in Systematic Reviews: A Systematic Review of Current Approaches". In: *Systematic Reviews* 4.1 (Jan. 14, 2015), p. 5. ISSN: 2046-4053. DOI: 10.1186/2046-4053-4-5. URL: https://doi.org/10.1186/2046-4053-4-5 (visited on 08/13/2024).

[40] OpenAI. *OpenAI Platform*. 2024. URL: https://platform.openai.com (visited on 07/30/2024).

[41] OpenAI et al. *GPT-4 Technical Report*. Mar. 4, 2024. arXiv: 2303.08774 [cs]. URL: http://arxiv.org/abs/2303.08774 (visited on 07/30/2024). Pre-published.

[42] Oleksiy Ostapenko et al. *Towards Modular LLMs by Building and Reusing a Library of LoRAs*. May 17, 2024. DOI: 10.48550/arXiv.2405.11157. arXiv: 2405.11157 [cs]. URL: http://arxiv.org/abs/2405.11157 (visited on 07/18/2024). Pre-published.

[43] Mourad Ouzzani et al. "Rayyan—a Web and Mobile App for Systematic Reviews". In: *Systematic Reviews* 5.1 (Dec. 5, 2016), p. 210. ISSN: 2046-4053. DOI: 10.1186/s13643-016-0384-4. URL: https://doi.org/10.1186/s13643-016-0384-4 (visited on 08/13/2024).

[44] *Papers with Code - MMLU Benchmark (Multi-task Language Understanding)*. URL: https://paperswithcode.com/sota/multi-task-language-understanding-on-mmlu (visited on 07/30/2024).

[45] Fabian Pedregosa et al. “Scikit-Learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12.85 (2011), pp. 2825–2830. ISSN: 1533-7928. URL: http://jmlr.org/papers/v12/pedregosa11a.html (visited on 08/21/2024).

[46] Jonas Pfeiffer et al. “AdapterFusion: Non-Destructive Task Composition for Transfer Learning”. In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. EACL 2021. Ed. by Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty. Online: Association for Computational Linguistics, Apr. 2021, pp. 487–503. DOI: 10.18653/v1/2021.eacl-main.39. URL: https://aclanthology.org/2021.eacl-main.39 (visited on 04/16/2024).

[47] Jonas Pfeiffer et al. “Modular Deep Learning”. In: *Transactions on Machine Learning Research* (June 12, 2023). ISSN: 2835-8856. URL: https://openreview.net/forum?id=z9EkXfvxta (visited on 04/04/2024).

[48] Nils Reimers and Iryna Gurevych. “Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks”. 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)*. EMNLP-IJCNLP 2019. Ed. by Kentaro Inui et al. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3982–3992. DOI: 10.18653/v1/D19-1410. URL: https://aclanthology.org/D19-1410 (visited on 08/11/2024).

[49] Girish Sundaram and Daniel Berleant. “Automating Systematic Literature Reviews with Natural Language Processing and Text Mining: A Systematic Literature Review”. In: *Proceedings of Eighth International Congress on Information and Communication Technology*. Ed. by Xin-She Yang et al. Singapore: Springer Nature, 2023, pp. 73–92. ISBN: 978-981-9932-43-6. DOI: 10.1007/978-981-99-3243-6_7.

[50] Hugo Touvron et al. *LLaMA: Open and Efficient Foundation Language Models*. Feb. 27, 2023. arXiv: 2302.13971 [cs]. URL: http://arxiv.org/abs/2302.13971 (visited on 07/28/2024). Pre-published.

[51] Ruslan Vasilev and Alexander D’yakonov. *Calibration of Neural Networks*. Mar. 19, 2023. arXiv: 2303.10761 [cs, stat]. URL: http://arxiv.org/abs/2303.10761 (visited on 08/21/2024). Pre-published.

[52] Ashish Vaswani et al. “Attention Is All You Need”. In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html (visited on 07/28/2024).

[53] Tu Vu et al. “Exploring and Predicting Transferability across NLP Tasks”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. EMNLP 2020. Ed. by Bonnie Webber et al. Online: Association for Computational Linguistics, Nov. 2020, pp. 7882–7926. DOI: 10.18653/v1/2020.emnlp-main.635. URL: https://aclanthology.org/2020.emnlp-main.635 (visited on 08/04/2024).

[54] Byron C. Wallace et al. “Deploying an Interactive Machine Learning System in an Evidence-Based Practice Center: Abstrackr”. In: *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*. IHI ’12. New York, NY, USA: Association for Computing Machinery, Jan. 28, 2012, pp. 819–824. ISBN: 978-1-4503-0781-9. DOI: 10.1145/2110363.2110464. URL: https://dl.acm.org/doi/10.1145/2110363.2110464 (visited on 08/13/2024).

[55] Zirui Wang et al. “GRADIENT VACCINE: INVESTIGATING AND IMPROVING MULTI-TASK OPTIMIZATION IN MASSIVELY MUL- TILINGUAL MODELS”. In: (2021).

[56] Jason Wei et al. “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models”. In: *Advances in Neural Information Processing Systems* 35 (Dec. 6, 2022), pp. 24824–24837. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html (visited on 08/14/2024).

[57] Jason Wei et al. *Finetuned Language Models Are Zero-Shot Learners*. Feb. 8, 2022. DOI: 10.48550/arXiv.2109.01652. arXiv: 2109.01652 [cs]. URL: http://arxiv.org/abs/2109.01652 (visited on 08/20/2024). Pre-published.

[58] Thomas Wolf et al. *HuggingFace’s Transformers: State-of-the-art Natural Language Processing*. July 13, 2020. DOI: 10.48550/arXiv.1910.03771. arXiv: 1910.03771 [cs]. URL: http://arxiv.org/abs/1910.03771 (visited on 07/28/2024). Pre-published.

[59] Lingling Xu et al. *Parameter-Efficient Fine-Tuning Methods for Pretrained Language Models: A Critical Review and Assessment*. Dec. 19, 2023. DOI: 10.48550/arXiv.2312.12148. arXiv: 2312.12148 [cs]. URL: http://arxiv.org/abs/2312.12148 (visited on 04/15/2024). Pre-published.

[60] Prateek Yadav et al. *TIES-Merging: Resolving Interference When Merging Models*. Oct. 26, 2023. DOI: 10.48550/arXiv.2306.01708. arXiv: 2306.01708 [cs]. URL: http://arxiv.org/abs/2306.01708 (visited on 03/30/2024). Pre-published.

[61] Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. "CrossFit: A Few-shot Learning Challenge for Cross-task Generalization in NLP". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2021. Ed. by Marie-Francine Moens et al. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 7163–7189. DOI: 10.18653/v1/2021.emnlp-main.572. URL: https://aclanthology.org/2021.emnlp-main.572 (visited on 08/21/2024).

[62] Jinghan Zhang et al. "Composing Parameter-Efficient Modules with Arithmetic Operation". In: *Advances in Neural Information Processing Systems* 36 (Dec. 15, 2023), pp. 12589–12610. URL: https://proceedings.neurips.cc/paper_files/paper/2023/hash/299a08ee712d4752c890938da99a77c6-Abstract-Conference.html (visited on 03/28/2024).

[63] Wangchunshu Zhou, Canwen Xu, and Julian McAuley. "Efficiently Tuned Parameters Are Task Embeddings". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2022. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 5007–5014. DOI: 10.18653/v1/2022.emnlp-main.334. URL: https://aclanthology.org/2022.emnlp-main.334 (visited on 08/11/2024).

[64] Fuzhen Zhuang et al. *A Comprehensive Survey on Transfer Learning*. June 23, 2020. arXiv: 1911.02685 [cs, stat]. URL: http://arxiv.org/abs/1911.02685 (visited on 08/21/2024). Pre-published.

Appendix A

Dataset composition

The sizes and positive class rates of the datasets are shown in Table A.1, and the titles of the respective reviews are listed below:

- **A**: Is aerosolised HOCl an effective and safe technological approach for the suppression of airborne viral loads in public spaces?
- **B**: What are the adaptations being implemented to address the health impacts of flooding in LMICs in the tropics and what is known about their effectiveness?
- **C**: Risk of serious COVID-19 outcomes among adults and children with moderate-to-severe asthma: a systematic review and meta-analysis
- **D**: The social lives of point-of-care tests in low- and middle-income countries: a qualitative evidence synthesis protocol
- **E**: Compact cities and the Covid-19 pandemic: Systematic review of the associations between transmission of Covid-19 or other respiratory viruses and population density or other features of neighbourhood design
- **F**: Assessing the implementation, outcomes and impact of national suicide prevention strategies: a systematic review
- **G**: Impact of the COVID-19 Pandemic and Scottish Public Sector Response on Refugees and Asylum Seekers
- **H**: A systematic review of the effectiveness of heat adaptation measures for urban areas in an oceanic climate

- **I**: An Evaluation of International Pandemic Recovery Strategies and Identification of Good Practice Relevant to Scotland

Review label	Dataset size	Support
A	3707	0.370
B	1360	0.188
C	2292	0.104
D	6976	0.031
E	1330	0.107
F	4365	0.029
G	643	0.152
H (eval only)	1381	0.019
I (test set)	599	0.217

Table A.1: Composition of the SR datasets used in the experiments.

Appendix B

Hyperparameters for fine-tuning

Throughout our experiments we fine-tune using the general configuration outlined in 4.2 and the following hyperparameters:

- Initial learning rate of $5e^{-5}$ with linear decay schedule.
- Effective batch size of 12
- LoRA α value of 8 with LoRA+ [25] ratio of 16
- LoRA dropout of 0.05

Additionally, we select the number of training epochs and LoRA rank according to the size of the training set, as specified in table B.1

Dataset size	Number of epochs	LoRA rank
< 50	8	2
50	7	4
100	6	4
200–400	5	4
600–1250	3	4
1500-1750	2	4
$>= 2000$	2	8

Table B.1: When fine-tuning with LoRA, we use the hyperparameters above that most closely match the training dataset size

Appendix C

Example of model prompt

Below is an example of a prompt used for the Llama 3 model, including two in-context examples. Abstracts have been truncated here for brevity, but would be included in full in the input to the model.

When using the OpenAI API, the content of the prompt is the same, but the text is provided as JSON list of objects, with one object corresponding to each message that is enclosed within '< |', '| >' tags in the Llama format.

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>

Your task is to analyse the provided title and abstract of
research papers to determine whether they should be included
in a systematic review.

The review is titled "Is aerosolised HOCl an effective and
safe technological approach for the suppression of airborne
viral loads in public spaces?"

The inclusion criteria for the review are as follows:

### Type of study

Include papers, book or report chapters containing empirical
data on HOCl
```

```
### Topic of study

Include studies on HOCl, hypochlorous acid, or any of the
following synonyms: hypochlorous acid, electrolysed water
(EW), acidic electrolysed water (AEW), neutral electrolysed
water (NEW), electrolysed oxidising water (EOW), mixed
oxidant water (MIOX), electrochemically activated water
(ECAW), super-oxidised (1/2 word variations) water/solution
(SOW), NaDCC, SDIC, sodium dichloroisocyanurate, sodium
troclosene, troclosenum natricum. Exclude studies on
hypochlorite.  Include studies on fogging or spraying of
HOCl, efficacy or toxicity. Exclude studies that do not
include any data on HOCl (or synonym) as an antimicrobial
Exclude studies that investigate ONLY a combination of HOCl
+ another method of disinfection.

### Population
Include studies on humans or animals, including food
decontamination.

### Setting

Include studies set in vitro (laboratory based), in vivo
(real life settings), and any interior or public spaces
where microbial pathogens need to be controlled.

Based on the title and abstract provided below, if it seems
possible that the paper fulfills the above criteria, please
respond with 'Yes'. If you are sure that the paper does not
fulfill the criteria, please respond with 'No'.

Here are some examples to get you started:

Title: A systematic review of the effectiveness of
interventions in the management of infection in the diabetic
```

```
foot

Abstract: The International Working Group on the Diabetic
Foot expert panel on infection conducted ...

Should the paper be considered for inclusion? Yes

Title: In vivo bioluminescence: a cellular reporter for
research and industry

Abstract: The detection of specific bacterial pathogens,
indicator microorganisms and antimicrobial substances ...

Should the paper be considered for inclusion? No

This is the research paper to
consider:<|eot_id|><|start_header_id|>user<|end_header_id|>

Title: Development of point-of-use water disinfection
technology using ceramicwater filter and electrochemical
hybrid system

Abstract: The efficiency of water disinfection using a
ceramic water filter and electrochemical ...

Should the paper be considered for
inclusion?<|eot_id|><|start_header_id|>assistant<|end_header_id|>
```