Coarse-to-Fine: Hierarchical Multi-task Learning for Natural Language Understanding

Zhaoye Fei[1,4†], **Yu Tian**[1†], **Yongkang Wu**[1†], **Xinyu Zhang**[1†], **Yutao Zhu**[2], **Zheng Liu**[1], **Jiawen Wu**[4], **Dejiang Kong**[1], **Ruofei Lai**[1], **Zhao Cao**[1*], **Zhicheng Dou**[3] **and Xipeng Qiu**[4]
[1]Huawei Poisson Lab, China
[2]University of Montreal, Montreal, Quebec, Canada
[3]Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
[4]School of Computer Science, Fudan University, Shanghai, China
{wuyongkang7,zhangxinyu35,caozhao1}@huawei.com

Abstract

Generalized text representations are the foundation of many natural language understanding tasks. To fully utilize the different corpus, it is inevitable that models need to understand the relevance among them. However, many methods ignore the relevance and adopt a single-channel model (a coarse paradigm) directly for all tasks, which lacks enough rationality and interpretation. In addition, some existing works learn downstream tasks by stitches skill block (a fine paradigm), which might cause irrational results due to its redundancy and noise. In this work, we first analyze the task correlation through three different perspectives, *i.e.*, data property, manual design, and model-based relevance, based on which the similar tasks are grouped together. Then, we propose a hierarchical framework with a coarse-to-fine paradigm, with the bottom level shared to all the tasks, the mid-level divided to different groups, and the top-level assigned to each of the tasks. This allows our model to learn basic language properties from all tasks, boost performance on relevant tasks, and reduce the negative impact from irrelevant tasks. Our experiments on 13 benchmark datasets across four natural language understanding tasks demonstrate the superiority of our method.

1 Introduction

Pre-trained language models have achieved great success on various natural language processing (NLP) tasks. Meanwhile, *pre-train-then-fine-tuning* has gradually become the mainstream paradigm (Devlin et al., 2019; Liu et al., 2019b; Yang et al., 2019). The pre-training process aims to learn a general language representation from the large-scale corpus. Such representations can be further fine-tuned on downstream datasets and perform specific tasks. Though great performance has been achieved, it is costly to fine-tune and save independent representation for each task. Therefore, researchers propose several multi-task learning methods (Phang et al., 2018; Liu et al., 2019a; Clark et al., 2019b) using a single-channel model (a coarse paradigm) to solve multiple tasks.

Inspired by human learning, multi-task learning believes that tasks can interact and boost each other. Therefore, to obtain a more robust representation that can handle a variety of tasks, a straightforward idea is to fine-tune a pre-trained model on many tasks simultaneously. Numerous previous studies have been conducted along this path. For example, MT-DNN (Liu et al., 2019a) uses a transformer-based model as a shared encoder and trains it on multiple downstream tasks (as shown in the left side of Figure 1). By this coarse paradigm, the representation model can be more generalized and robust. Although it has been observed that some combinations of tasks yield improvements, this is not always the case. Researchers (Aribandi et al., 2021) have also found that the impact between tasks is a double-edged sword, meaning that some tasks may also hurt others. Indeed, modeling heterogeneous tasks often require distinct representation spaces. For example, naively training natural language inference (NLI) tasks with different hypothesis types together can lead to performance declines.

Some recent work proposes sparsely activating multiple modules for different tasks (Tang et al., 2022) to mitigate the negative effects across tasks. These fine paradigms activate distinct modules according to predefined skills for learning downstream tasks (as shown in the middle part of Figure 1). However, the number of parameters is also multiplied when multiple modules are activated for a task. This problem becomes even more severe when large pre-trained language models are applied.

To address the aforementioned problems, we

†These authors contributed equally to this work.
*Corresponding author.

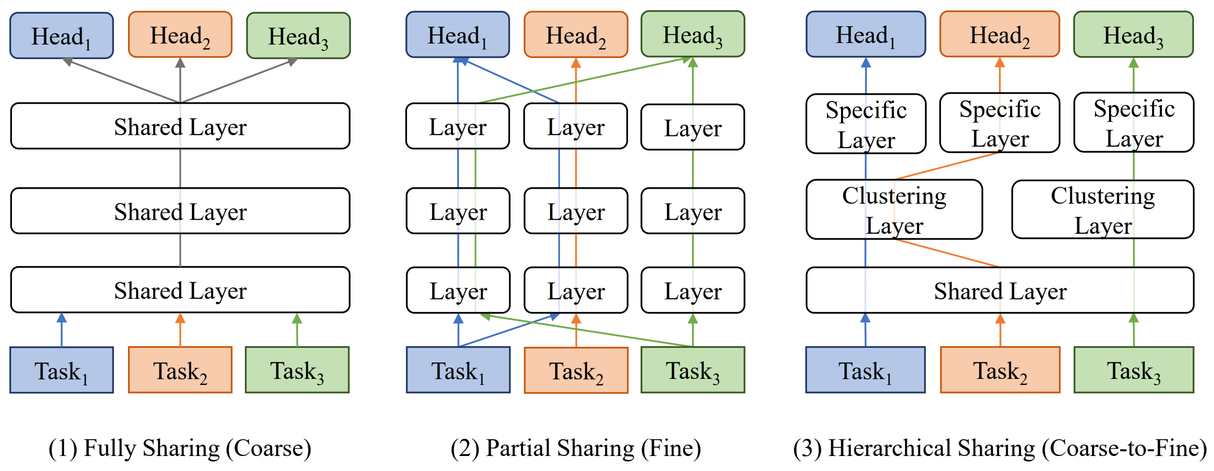

Figure 1: The illustration of hard sharing structure, partial sharing structure, and our hierarchical sharing structure. The three tasks and the corresponding activated paths are in different colors. In our approach, the first and second are two relevant tasks, so they share the same task-clustering layers. Finally, they are fed into different task-specific layers.

first conduct a correlation analysis on four natural language understanding (NLU) categories (a total of 13 datasets) and find that some tasks can complement one another, while others cannot. Based on these observations, we propose three measures (*i.e.*, data property, manual design, and model-based relevance) to categorize the tasks into different groups.

Then, inspired by recent studies (Kovaleva et al., 2019; Rogers et al., 2020) that different layers learn information in different levels, we design a novel hierarchical sharing framework, dubbed HMNet, a coarse-to-fine multi-task learning paradigm for natural language understanding (as shown in the right side of Figure 1). Our method is a coarse-to-fine paradigm, where the layers are divided from the bottom up into shared, task-clustering, and task-specific levels. We design the bottom layers as shared, which are optimized by all tasks. Thereafter, we employ distinct layers for different groups obtained in the previous step. In this way, the relevant tasks grouped in the same cluster can boost each other, while the negative impact from tasks in other groups can be avoided. The top layers are totally separated for distinct tasks, so that some task-specific information can be well-captured. It is worth noting that each task only activates a single module in each layer, so our model does not require additional parameters during inference. Extensive experiments on 13 datasets show that our method can achieve better performance on most NLU tasks, demonstrating its superiority over existing hard sharing or partial sharing structures.

Our main contributions are three-fold:

(1) We perform a series of correlation analyses (*i.e.*, data property, manual design, and model-based relevance) on 13 NLU tasks, which sheds light on the positive and negative effects among the different tasks.

(2) We design a novel hierarchical sharing framework, dubbed HMNet, a coarse-to-fine multi-task learning paradigm for natural language understanding. It can better leverage the positive interactions among different tasks while reducing the negative influence.

(3) We conduct experiments on 13 commonly used NLU datasets and validate the effectiveness of our proposed method. The results demonstrate that our framework achieves highly competitive performance while saving over 34% parameters than the partial sharing structure.

2 Related Work

In this section, we briefly introduce some Transformer-based pre-trained models and recent work on multi-task learning.

2.1 Transformer-based Pre-trained Models

Transformer is a neural structure consisting of multiple stacked self-attention modules (Devlin et al., 2019; Liu et al., 2019b; Yang et al., 2019). With the bidirectional attention mechanism, the model can capture contextual information from both sides effectively. BERT (Devlin et al., 2019) proposes a masked language modeling objective to pre-train a Transformer encoder, and achieves dramatic performance on several natural language understanding tasks (Bowman et al., 2015; Williams et al., 2018a; Wang et al., 2019b; Zhu et al., 2021a,b). Since then, the *pre-train-then-fine-tuning* paradigm has gradually become mainstream. Researchers propose various new pre-training strategies to facilitate the model in several aspects (Yang et al., 2019; Lewis et al., 2020; Raffel et al., 2020). For example, in order to obtain a more robust model, RoBERTa (Liu et al., 2019b) adapts several tailored training strategies and employs more data. There are also many methods extending the pre-training on Transformer decoders. GPT-2 (Radford et al., 2019) is a decoder-only structure that is pre-trained in accordance with the language generation objectives. It also performs exceptionally well on many text generation tasks. Though pre-trained models can be fine-tuned for distinct tasks, it is costly to maintain a separate model for each task, especially when the model is huge.

2.2 Multi-task Learning

Multi-task learning is an integrated learning method in which multiple tasks share the same structure for training simultaneously. It can enhance the generalization and performance of each task. Consequently, multi-task learning can also be applied to pre-trained language models. A typical practice is to fine-tune the pre-trained language models by conducting multiple tasks concurrently (Liu et al., 2019a). It is reported, however, that not all tasks can boost each other, and that noise may also be introduced (Aribandi et al., 2021). Therefore, it is essential to analyze the relationship between tasks before training them jointly. Some recent work proposes addressing this issue by sparsely activating distinct modules for different tasks (Tang et al., 2022). This allows the modules to be trained by relevant tasks. Unfortunately, these approaches usually rely on predefined activation paths, and they have to activate multiple modules in order to achieve high performance during the inference stage. This undoubtedly increases the delay and cost of model application.

The main differences between our method and others are: (1) We analyze the task correlation, and use it to group the tasks. Relevant tasks within the same group will be used to fine-tune the same modules, while irrelevant tasks will not interfere with one another. (2) Our model only activates a single module in each layer for each task, therefore it has the same number of parameters as a single model.

3 Proposed Method

3.1 Overview

The overview of our method is illustrated in the right side of Figure 1. In general, there are three different kinds of layers in our structure: shared layers, task-clustering layers, and task-specific layers. The shared layers are in the bottom and optimized by all tasks. The middle part is the task-clustering layers. Tasks that are classified as relevant (introduced in the next section) will optimize the same group of layers. The top layers are task-specific, meaning that each task has its own module.

Our method adopts a coarse-to-fine paradigm. In this manner, the model can obtain generalized text representations in shared layers (a coarse-grained manner) and effectively interact and learn according to task correlation in the task-clustering layer (a fined-grained manner), then send it to the task-specific layer for specific task training.

3.2 Task Relevance Analysis

Existing multi-task learning framework assumes that all tasks can facilitate each other. However, researchers have also reported the negative effect caused by irrelevant tasks (Aribandi et al., 2021). To investigate the influence of each task, we devise three methods from different perspectives to measure the task relevance.

Data Property Since all tasks we study are based on natural language texts, we first analyze the relevance between tasks through the property of their data. Specifically, we consider the syntactic information and employ vocabulary co-occurrence to measure task relevance. Formally, given datasets of two tasks denoted as $\mathcal{D}_\text{s}$ and $\mathcal{D}_\text{t}$, where s and t stand for *source* and *target*, we compute the vocabulary co-occurrence as follows:

$$r_\text{cs} = |V_\text{s} \cap V_\text{t}|/|V_\text{s}|, \tag{1}$$

$$r_\text{ct} = |V_\text{s} \cap V_\text{t}|/|V_\text{t}|, \tag{2}$$

where V_s and V_t are the vocabulary of the source and target datasets, respectively, r_cs measures the ratio of words in the source dataset that are shared with the target datasets, and r_ct represents the opposite. Intuitively, a higher value of r_cs indicates a higher impact of the target task on the

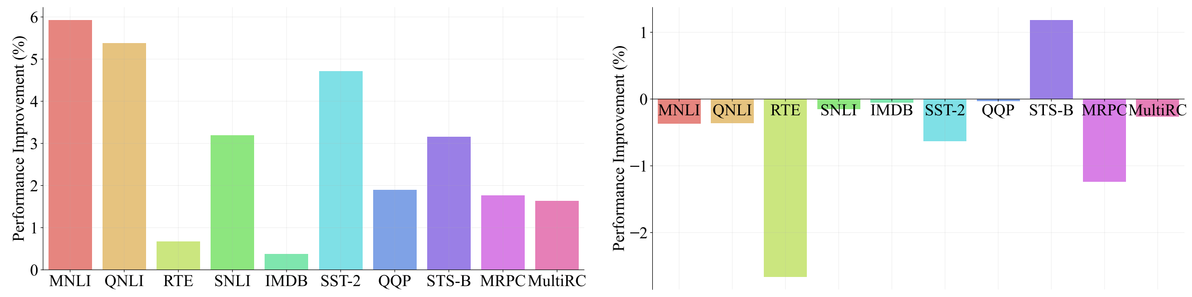

Figure 2: Influence of other tasks on the BoolQ dataset (left) and influence of the BoolQ dataset on others (right).

source task. Note that this metric also considers the data size, as a larger dataset usually contains more words. For example, MNLI (Williams et al., 2018b), as the largest dataset, has a large vocabulary co-occurrence with others. It can provide some basic language knowledge for other tasks. Conversely, WNLI (Wang et al., 2019b) is a small dataset with only hundreds of samples, and it may have limited impacts on other tasks. Due to the space limitation, we show the entire results in Appendix A.

Manual Design Inspired by a recent work that designs various paths for different tasks (Tang et al., 2022), we also examine the tasks artificially. Based on the manually designed task purpose, we divide them into four groups. *i.e.*, Natural language inference, sentiment classification, similarity and paragraphing, and question answering. We assume tasks that have the same purpose supply the same systematic information for models. For instance, the NLI tasks generally rely on the models' judgment according to the deep semantics information between premise and hypothesis, while the question answering tasks ask the model to infer the answer from the passage according to the question. These tasks and their category will be reported in Section 4.1.

Model-based Relevance In addition, to manually analyze the task or data, we further propose measuring task relevance using neural networks. Previous studies have demonstrated the advantages of multi-task learning (Liu et al., 2019a), notably that relevant tasks can enhance each other's performance. Consequently, task relevance can be inferred by comparing the performance difference between training two tasks independently and training them jointly. Concretely, for a source and a target task, we fine-tune two models (we use BERT in our experiments) respectively for them. Their average performance is denoted as f_s and f_t. Afterward, we fine-tune another model (BERT) for the two tasks through standard multi-task learning. The corresponding performance is denoted as f_{js} and f_{jt}. Finally, we can compute the task relevance as the performance improvement:

$$r_{ms} = (f_{js} - f_s)/f_s, \tag{3}$$
$$r_{mt} = (f_{jt} - f_t)/f_t, \tag{4}$$

where r_{ms} reflects the influence of the target task on the source task, while r_{mt} represents the opposite. We calculate the relevance for all tasks, and the results about BoolQ as an example in Figure 2 to highlight the asymmetrical relevance between tasks. The complete form is shown in Appendix A. In Figure 2, we find that most tasks have a positive impact on BoolQ, even though the data property and the purpose of these tasks are very different. BoolQ, on the other hand, harms other tasks. One possible reason is learning for QA needs more complicated information and learning on other tasks can supply this information to improve the performance of BoolQ. But as for other tasks, the information obtained by learning in BoolQ like noise damages the learning of themselves.

Since the asymmetrical relevance between task pairs, we consider the fully shared model is harmful to some tasks in multi-task learning. Inspired by Rogers et al., we consider that the improvement of word-level information training in the low layer occurs a positive impact mostly, whereas the more negative impact is sourced from the damage of the deep semantic learning in the middle and top layers. As a result, we propose a hierarchical sharing method for multi-task learning. It is based on a hierarchical sharing of the task with different relevance to leverage more positive interaction and reduce the negative impact of multi-task learning. More detail about the structure is described in the next section.

3.3 HMNet

We propose a hierarchical multi-task learning framework (HMNet) based on task relevance. It is built on Transformer (Vaswani et al., 2017), which has been widely applied in NLP tasks.[1] We omit the details of Transformer and refer readers to the original paper. As shown in the right side of Figure 1, HMNet has three different kinds of Transformer layers from the bottom up. All tasks will first pass the shared layers, then, each task will go through task-clustering layers according to their task group (introduced later). Finally, each task has its own task-specific layer, and the associated head is used to accomplish the task.

In the following paragraphs, we first introduce the details of the layers, followed by the task clustering process.

Shared layers The shared layers are stacked in the bottom of HMNet. We design bottom layers as shared since it has been demonstrated that they capture low-level semantic or structural information in existing pre-trained models (Jawahar et al., 2019). We believe that such information is universally contained in all texts, so all tasks collaborate to optimize the shared layers.

Task-clustering layers Based on our task relevance analysis, we cluster the tasks into various groups. For tasks within the same cluster, the same (set of) clustering layers will be optimized. As illustrated on the right side of Figure 1, the first two tasks are grouped into one cluster, so they optimize the first set of clustering layers. On the contrary, only the third task tunes the second set of clustering layers. With the task-clustering layers, relevant tasks can optimize the same set of parameters, enabling the sharing of their knowledge. In the meantime, the irrelevant tasks can be isolated, thereby eliminating the noise.

Task-specific layers Task-specific layers are in the top of our HMNet. According to recent studies (Jawahar et al., 2019), the top layers of pre-trained language models typically learn task-specific knowledge. Therefore, we separate all tasks and let them have their own Transformer layers. There is also a head associated with each task-specific layer to accomplish the task. For example, a linear classifier with an activation function is often applied for classification tasks.

Clustering Process In our task relevance analysis, we propose three methods to measure the task relevance. Among them, the manual design can directly group the tasks into different clusters. For the other two methods, we employ an unsupervised clustering method, *i.e.*, k-means. Particularly, given a source task S, and n target tasks $\{T_1, \cdots, T_n\}$, we can compute the relevance scores $\{r_{\text{ST}_1}, \cdots, r_{\text{ST}_n}\}$ by Equation (2) or (4).[2] Then, we treat the n scores as features and apply k-means clustering algorithm. Finally, the tasks can be grouped into k clusters. The grouping results are given in the Appendix A.

Remark Different from traditional multi-task learning that shares all layers among all tasks, our HMNet only shares layers at the bottom (a coarse-grained manner). The higher layers are shared by only relevant tasks or used alone(a fined-grained manner). Since the layers close to specific tasks are separated, our method can alleviate the contradiction between tasks. On the other hand, each task only activates one channel at each layer, so the total parameters are comparable with a single-channel model. This is much more efficient than sparsely activated structures.

3.4 Optimization

HMNet is based on a multi-layer Transformer, which is similar to existing pre-trained language models, such as BERT (Devlin et al., 2019). Therefore, we use BERT as the backbone model to initialize the parameters in each layer. Then, each task has its own path (as described earlier), and we can use it to fine-tune the corresponding layers. The training process our method is summarized in Algorithm 1. In each epoch, a mini-batch b_t is packed, and the HMNet is updated by the path for the dataset $\mathcal{D}_i$.

4 Experiment

4.1 Datasets and Evaluation Metrics

We conduct experiments on 13 NLU tasks and compare the performance of our HMNet with other baselines. These tasks can be grouped into four categories:

[1]Other structures, such as convolutional or recurrent networks, may also be compatible with our method. This is left for future work.

[2]Since the task correlations are asymmetric, scores such as r_{TS} may also work.

Algorithm 1: Training Process

```
Initialize model parameters from pre-trained BERT;
Set the max number of training epoch $E_m$;
**Prepare data**
**for** $i$ *in* $1, 2, \cdots, N$ **do**
    Pack the dataset $\mathcal{D}_i$ into minibatch $B_i$;
**end**
**Multi-task learning**
**for** epoch $t$ in $1, 2, \cdots, E_m$ **do**
    Merge all the datasets: $B = B_1 \cup B_2 \cdots \cup B_N$;
    Shuffle $B$;
    **for** $b_t$ *in* $B$ **do**
        // $b_t$ is a mini-batch of task $t$;
        Feed $b_t$ into shared layers $\rightarrow \mathbf{b}_t^s$;
        Feed $\mathbf{b}_t^s$ into task-clustering layers $\rightarrow \mathbf{b}_t^c$;
        Feed $\mathbf{b}_t^c$ into task-specific layers $\rightarrow \mathbf{b}_t^t$;
        Feed $\mathbf{b}_t^t$ into task-specific head;
        Compute loss and gradient;
        Update model;
    **end**
**end**
```

(1) **Natural Language Inference (NLI):** These tasks aim to determine whether a *hypothesis* is entailed, contradicted, or undetermined given a *premise*. They require the model to measure the logical coherence between two sentences. We select six datasets: MNLI (Williams et al., 2018b), QNLI, RTE, WNLI (Wang et al., 2019b), CB (Wang et al., 2019a), and SNLI (Bowman et al., 2015).

(2) **Sentiment Classification:** These tasks ask the model to classify the sentiment polarity of a sentence (*i.e.*, positive or negative). We use IMDB (Maas et al., 2011) and SST-2 (Socher et al., 2013) datasets.

(3) **Similarity and Paraphrase:** These tasks aim at determining whether two sentences have the same or similar meaning. The semantic relationship between two sentences is vital in these tasks. We choose three commonly used datasets: QQP (Wang et al., 2019b), STS-B (Cer et al., 2017), and MRPC (Dolan and Brockett, 2005).

(4) **Question Answering:** This task requires the model to answer a question by reasoning on a given paragraph. BoolQ (Clark et al., 2019a) and MultiRC (Khashabi et al., 2018) datasets are used.

Detailed statistics of benchmark datasets are mentioned in Appendix B.

4.2 Baseline

We compare with our HMNet with three other training methods:

Single-Task fine-tuning: We fine-tune a pre-trained BERT for each task independently. As a result, 13 models are obtained in total.

Multi-Task fine-tuning: Following MT-DNN (Liu et al., 2019a), we use the pre-trained BERT as a shared encoder and add a task-special head for each task. During fine-tuning, all parameters are optimized by all tasks jointly. With multi-task learning, only one model needs to be trained and saved.

SkillNet-style fine-tuning: This is a sparsely activated multi-task learning structure proposed by Tang et al. (2022). They design seven basic skills, such as getting the semantic meaning of a sequence and understanding a question. Then, they divide typical NLU tasks into five categories, each requiring a unique skill combination. The skill module is implemented by adding multiple FFN layers into the original Transformer structure. This model is also initialized by a pre-trained BERT. However, as each task activates multiple skill modules, compared with our method and other baselines, on average 1.0× more parameters (*i.e.*, 223M vs. 110M) are used in inference.

4.3 Implementation details

For a fair comparison, all methods are initialized by the `bert-base-uncased` checkpoint with the same seed. The batch size is 32, the max length of sequence is 512, and the initial learning rate is 2e-5, which is linearly decayed. AdamW (Loshchilov and Hutter, 2019) optimizer is applied. We train all methods for three epochs.

For the NLI and text similarity & paraphrase tasks, similar to the vanilla BERT, we concatenate the sentence pair by adding a separator token `[SEP]` and a head token `[CLS]`. For the sentiment classification task, the input is a single sentence, so we only add a head token `[CLS]`. For the QA task, we concatenate the question, passage, and answer, then separate them by two `[SEP]` tokens. A head token `[CLS]` is also added at the beginning of the sequence. All tasks are performed in similar ways, namely processing the embedding of the `[CLS]` token by a linear layer with a softmax activation function, and output the probability of each category. The default setting of HMNet has eight shared layers, two task-clustering layers, and two task-specific layers.

4.4 Experimental Results

Table 1 shows the results on all datasets. In general, our HMNet performs better than other baselines on most datasets and achieves the best result in terms of the average score. This clearly demonstrates the

	Metric	Single-task	Multi-task	SkillNet	HMNet$_{\mathrm{d}}$	HMNet$_{\mathrm{md}}$	HMNet$_{\mathrm{m}}$
NATURAL LANGUAGE INFERENCE							
MNLI (m/mm)	Acc.	**85.0**/84.6	84.5/84.8	84.5/84.6	84.8/84.6	84.7/**85.2**	84.8/84.9
QNLI	Acc.	**91.6**	90.9	90.7	90.8	91.0	91.1
RTE	Acc.	66.1	79.7	77.6	73.3	79.4	**81.2**
WNLI	Acc.	56.3	56.3	56.3	56.3	56.3	56.3
CB	Acc.	67.8	80.3	85.7	**87.5**	82.1	82.1
SNLI	Acc.	91.0	91.3	91.1	91.0	**91.4**	91.1
SENTIMENT CLASSIFICATION							
IMDB	Acc.	93.9	93.9	**94.1**	93.8	93.9	93.9
SST-2	Acc.	92.3	93.1	92.5	**93.2**	92.3	93.0
SIMILARITY AND PARAPHRASE							
QQP	Acc.	90.9	90.7	**91.0**	90.9	**91.0**	90.9
STS-B	Corr.	85.8	85.5	86.0	86.9	**87.5**	87.4
MRPC	Acc.	83.3	81.4	85.7	88.2	**90.4**	88.5
QUESTION ANSWERING							
BoolQ	Acc.	71.4	77.9	**80.7**	79.0	79.1	80.3
MultiRC	F1$_{\mathrm{a}}$	65.2	68.4	68.1	**68.9**	66.7	68.4
Average Score	-	80.4	82.8	83.5	83.5	83.7	**84.0**
# Params Activated	-	110M	110M	>166M	110M	110M	110M
# Overall Params	-	110M	110M	450M	231M	240M	231M

Table 1: Results of all methods on 13 datasets. The best results are in **bold**. HMNet is our proposed method, and the subscript stands for three task relevance metrics, *i.e.*, data property (d), manual design (md), and model-based relevance (m). During inference, SkillNet has to activate at least two channels, thus the number of parameters is larger than 166M. In contrast, our HMNet only activates one channel at each layer, which requires less parameters.

superiority of our proposed HMNet. We further have the following observations:

(1) Comparing the performance between single-task and multi-task learning, it is evident that the latter can bring improvement for most tasks. Furthermore, this enhancement is related to the size of the datasets. Specifically, for the large dataset, such as MNLI and QQP (both of which have more than 300k training samples), the single-task learning can perform slightly better than multi-task learning. This implies that not all tasks can complement each other, and for those tasks with sufficient training data, adding extra tasks may not improve or even degrade performance. The potential reason is that the data in other tasks are collected from different domains and require models with distinct capabilities. Simply combining them will result in noise. For QA tasks, multi-task learning often leads to better performance. Indeed, QA is more complicated than other tasks, thus training on other tasks can be beneficial (*e.g.*, better capture the relationship between question and answer). Some recent studies (Aribandi et al., 2021; Aghajanyan et al., 2021) have reported that incorporating massive tasks may alleviate the negative effect between tasks. It is interesting to investigate if our model can be further improved by adding more tasks, and we leave this as future work.

(2) Sparsely activated (SkillNet-style) methods can achieve better performance than single-/multi-task learning. Its advantages stem from two perspectives: First, this method groups tasks according to their requiring skills, so the tasks rely on similar skills can enhance one another (*e.g.*, both NLI and semantic similarity judgment rely on the skill of understanding how two text segments interact), while avoiding noise from other used skills. On the other hand, since multiple skill modules are activated, more parameters improve the model's capacity.

(3) Our HMNet with three different task relevance measurements can consistently outperform all baseline methods. Specifically, HMNet brings more than 0.7% absolute improvement over multi-task learning in terms of average score. We attribute this improvement to our architecture of task-clustering and task-specific layers. Instead of sharing all layers, HMNet gradually separates the tasks in higher layers, so that the general language knowledge can be accumulated while the noise can be filtered out. Different from SkillNet-style methods, our HMNet only activates one channel at each layer, so its number of parameters is identical to a vanilla BERT. Surprisingly, HMNet with fewer parameters can even perform better. This demonstrates again the effectiveness of our proposed task clustering and the hierarchical sharing structure.

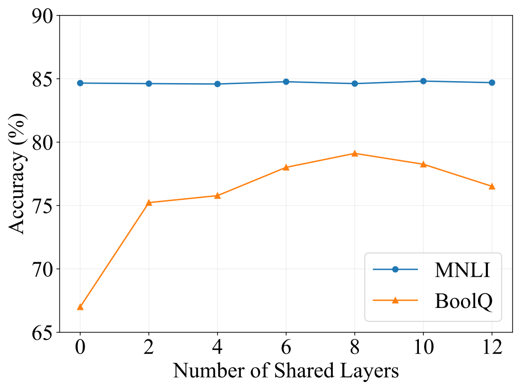

Figure 3: Influence of different numbers of shared layers in HMNet.

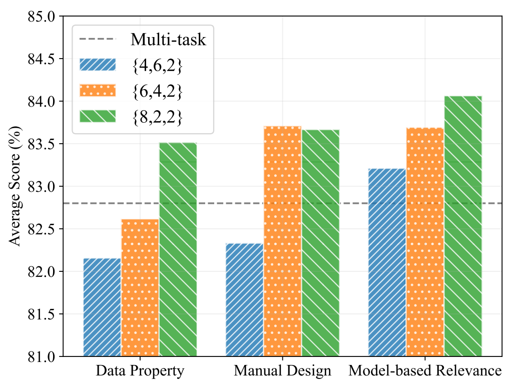

Figure 4: Results of average score with respect to different layer combinations.

4.5 Further Analysis

Influence of Shared Layers In HMNet, we design the bottom layers as fully shared, whilst the higher layers are only shared by task clusters or are task-specific. This approach is motivated by the hypothesis described in Section 3.2, which argues that the positive influence between tasks comes from capturing more general language knowledge at the bottom layer, and the negative impact comes from the noise brought by task interaction in the middle and top layers. To prove this hypothesis, we experiment on using different numbers of shared layers to investigate their effect. This experiment is conducted on the MNLI and BoolQ datasets. We train HMNet on them and report their performance accordingly. Since only two tasks are considered, we categorize them into two clusters, so the task-clustering layers are transformed into task-specific. The results are shown in Figure 3. We can observe that the performance of MNLI has minor changes as it contains sufficient training data. On the contrary, the result of BoolQ increases significantly when shared layers are used (from zero to eight). This reflects the advantage of multi-task learning. However, when more shared layers are employed (more than eight), the performance degrades. This confirms our assumption that task-specific knowledge is often learned in the upper layers, which supports our design of gradually separating tasks from the bottom up.

Different Metrics for Task Relevance In Section 3.2, we devise three different metrics for task relevance, *i.e.*, data property, manual design, and model-based relevance. Their performance is shown in Figure 4 and the right side of Table 1. We can see: First, HMNet can outperform other baselines with any task relevance assessment. This highlights the significance of task clustering in multi-task learning. Moreover, the result demonstrates the adaptability of our method with regard to the evaluation of task relevance. As an early exploration of the effect of task relevance on multi-task learning, the three metrics we have provided are very preliminary. We believe that a more accurate task relevance could bring further improvement. Second, the model-based similarity performs the best. With the help of deep neural models, the task relevance in high dimensions can be better captured. Such relevance is hard to be observed by humans or extracted from shallow data properties. In addition, quantifying the tasks' relevance from the perspective of the model can narrow the gap between task clustering and multi-task learning. Notably, though better performance is obtained, the model-based metric needs additional cost on model training. The other two metrics can avoid it.

Influence of Different Layer Combinations HMNet has three different kinds of layers, so their different combinations may influence the performance. We conduct an experiment by tuning the number of layers at each level. For clarity, we use $\{x, y, z\}$ to denote a HMNet with x shared layers, y task-clustering layers, and z task-specific layers. Experimental results are shown in Figure 4. Using eight shared layers yields the optimal performance. In comparison to traditional multi-task learning that shares all layers, HMNet with more than six shared layers can obtain better results. This implies that sharing all layers is not an effective strategy for multi-task learning, and our hierarchical sharing structure can mitigate the negative effect across tasks. Besides, when equipping with the model-based similarity, HMNet outperforms multi-task learning with any combination of shared, task-clustering, and task-specific layers. This reflects the robustness of our method and validates that the excellent performance of our method stems from our HMNet architecture and the consideration of task relevance rather than finely tuned hyperparameters.

5 Conclusion and Future Work

In this work, we explore task correlation and built a hierarchical multi-task learning framework. Our framework adopts a coarse-to-fine manner, in which the tasks are gradually separated from the bottom up. By doing so, it can reap the benefits of multi-tasking learning at the lower layers while avoiding its harmful impact on the upper layers. Extensive experiments on several challenging NLU datasets showed that our model achieves better performance than existing strategies. Further experiments indicated that our methods are flexible with the choice of task relevance metrics, and robust with the hyperparameter selection. As a preliminary study on incorporating task relevance into multi-task learning, there are several potential future directions, such as new backbone models and task relevance metrics.

References

Armen Aghajanyan, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. 2021. Muppet: Massive multi-task representations with pre-finetuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5799–5811. Association for Computational Linguistics.

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q. Tran, Dara Bahri, Jianmo Ni, Jai Prakash Gupta, Kai Hui, Sebastian Ruder, and Donald Metzler. 2021. Ext5: Towards extreme multi-task scaling for transfer learning. *CoRR*, abs/2111.10952.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 632–642. The Association for Computational Linguistics.

Daniel M. Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*, pages 1–14. Association for Computational Linguistics.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019a. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2924–2936. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Urvashi Khandelwal, Christopher D. Manning, and Quoc V. Le. 2019b. Bam! born-again multi-task networks for natural language understanding. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5931–5937. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing, IWP@IJCNLP 2005, Jeju Island, Korea, October 2005, 2005*. Asian Federation of Natural Language Processing.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3651–3657. Association for Computational Linguistics.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 252–262. Association for Computational Linguistics.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. 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 2019, Hong Kong, China, November 3-7, 2019*, pages 4364–4373. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4487–4496. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 142–150. The Association for Computer Linguistics.

Jason Phang, Thibault Févry, and Samuel R. Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *CoRR*, abs/1811.01088.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how BERT works. *Trans. Assoc. Comput. Linguistics*, 8:842–866.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL.

Duyu Tang, Fan Zhang, Yong Dai, Cong Zhou, Shuangzhi Wu, and Shuming Shi. 2022. Skillnet-nlu: A sparsely activated model for general-purpose natural language understanding.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3261–3275.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018a. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018b. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Yutao Zhu, Jian-Yun Nie, Zhicheng Dou, Zhengyi Ma, Xinyu Zhang, Pan Du, Xiaochen Zuo, and Hao Jiang. 2021a. Contrastive learning of user behavior sequence for context-aware document ranking. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pages 2780–2791. ACM.

Yutao Zhu, Kun Zhou, Jian-Yun Nie, Shengchao Liu, and Zhicheng Dou. 2021b. Neural sentence ordering based on constraint graphs. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 14656–14664. AAAI Press.

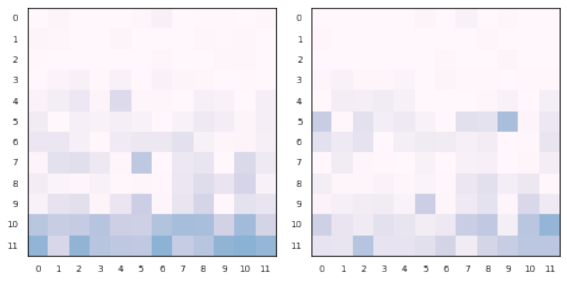

Figure 5: Per head attention maps' cosine similarity between fine-tuned model and co-trained model using positive task pair(QQP & RTE). Darker colors means greater differences. left: QQP right: RTE

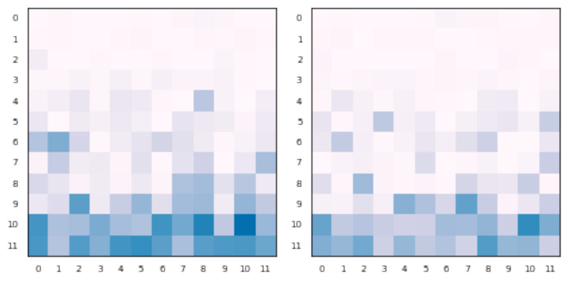

Figure 6: Per head attention maps' cosine similarity between fine-tuned model and co-trained model using negative task pair(MRPC & RTE). Darker colors means greater differences. left: MRPC right: RTE

A The Clustering Results

As described in Section 3.2, we group these tasks from three prospects, *i.e.*, data property, manual design, and model-based relevance.

As for the data property of datasets, Table 2 reports the task relevance of all 13 datasets. Based on these features, we cluster all tasks into three groups using k-means. {WNLI, CB}, {MultiRC, RTE, SST-2, MRPC, STS-B}, {QQP, QNLI, BoolQ, IMDB, SNLI, MNLI} is the grouping results.

As for the model-based relevance, the results are shown in Table 3. Similarly, we cluster these tasks using k-means. The outcomes are {CB, WNLI, QQP, RTE}, {MRPC, QNLI, BoolQ, IMDB, SST-2}, {MultiRC, STS-B, SNLI, MNLI}.

B Dataset Statistics

The statistic of all datasets are shown in Table 4.

C Attention Map Similarity

Following previous work (Liu et al., 2019a; Rogers et al., 2020), we compare the attention maps between fine-tuned model and the co-trained model to explore the behavior during multi-task learning. As shown in Table 3, the relevance between RTE and QQP is relatively positive and that between RTE and MRPC is relatively negative. We conduct some experiments using these tasks to explore the training difference between positive task-pair and negative pairs. Figure 5 compares the similarity between positive task pair (QQP & RTE), and Figure 6 compares the negative task-pair (MRPC & RTE). As shown in these figures, regardless of the positive pair or negative pair, the attention map of the bottom six layers is very similar, while that of the top two layers is different. Compared Figure 5 with Figure 6, we can see the attention map in the positive pair is more similar in the middle layers. Therefore, when considering the model structure, we set the bottom layers as shared to learn general knowledge from all tasks. Accordingly, the higher layers are designed as more task-specific.

	MNLI	QNLI	RTE	WNLI	CB	SNLI	IMDB	SST-2	QQP	STSB	MRPC	BoolQ	MultiRC
MNLI	100.00%	92.83%	97.64%	99.58%	99.42%	98.50%	95.70%	99.21%	93.46%	98.29%	98.21%	93.83%	98.76%
QNLI	94.40%	100.00%	98.11%	98.32%	98.01%	95.02%	94.00%	96.55%	94.19%	97.71%	98.02%	95.31%	97.55%
RTE	52.42%	51.80%	100.00%	84.72%	81.73%	59.25%	52.99%	65.23%	52.26%	74.24%	74.11%	54.11%	66.54%
WNLI	5.50%	5.34%	8.72%	100.00%	24.80%	7.47%	5.63%	9.90%	5.51%	10.64%	9.77%	5.75%	8.78%
CB	10.62%	10.29%	16.26%	47.93%	100.00%	13.76%	10.87%	18.86%	10.61%	18.95%	18.44%	11.00%	16.15%
SNLI	69.92%	66.33%	78.35%	95.94%	91.44%	100.00%	70.87%	84.65%	68.98%	83.62%	80.12%	69.19%	80.05%
IMDB	93.74%	90.55%	96.68%	99.72%	99.67%	97.80%	100.00%	99.69%	91.86%	97.45%	97.06%	92.13%	97.41%
SST-2	44.45%	42.54%	54.44%	80.31%	79.12%	53.43%	45.60%	100.00%	44.07%	58.34%	57.02%	44.68%	56.31%
QQP	92.93%	92.11%	96.79%	99.09%	98.80%	96.63%	93.25%	97.80%	100.00%	97.67%	97.40%	93.65%	97.13%
STSB	41.08%	40.17%	57.80%	80.52%	74.18%	49.24%	41.59%	54.43%	41.06%	100.00%	67.28%	42.43%	54.52%
MRPC	43.97%	43.15%	61.80%	79.12%	77.30%	50.53%	44.36%	56.97%	43.85%	72.06%	100.00%	45.47%	57.67%
BoolQ	87.67%	87.58%	94.18%	97.20%	96.27%	91.07%	87.88%	93.19%	88.01%	94.84%	94.90%	100.00%	93.29%
MultiRC	57.48%	55.84%	72.14%	92.43%	88.03%	65.64%	57.88%	73.15%	56.86%	75.92%	74.98%	58.11%	100.00%

Table 2: Task relevance based on data property. It is computed by the vocabulary co-occurrence between the task pair as Equation (2).

	MNLI	QNLI	RTE	WNLI	CB	SNLI	IMDB	SST-2	QQP	STS-B	MRPC	BoolQ	MultiRC
MNLI	0.00%	-0.10%	15.96%	50.00%	47.31%	0.75%	-0.29%	0.00%	-0.14%	-0.96%	-1.87%	5.93%	6.42%
QNLI	-0.54%	0.00%	2.13%	21.43%	26.37%	0.24%	-0.09%	-0.62%	-0.06%	0.43%	-0.69%	5.38%	1.58%
RTE	-0.47%	-0.32%	0.00%	7.14%	10.45%	0.25%	0.10%	-0.98%	-0.03%	-0.10%	-1.95%	0.67%	1.03%
WNLI	-0.28%	-0.32%	-3.19%	0.00%	2.62%	0.08%	0.08%	-0.74%	-0.03%	-0.16%	-0.39%	0.59%	-0.18%
CB	-0.35%	0.02%	0.00%	7.14%	0.00%	-0.30%	-0.09%	-0.49%	0.04%	0.35%	-0.50%	-0.17%	0.29%
SNLI	-0.19%	-0.64%	11.70%	64.29%	33.51%	0.00%	-0.17%	-1.85%	-0.12%	-1.23%	-3.16%	3.19%	3.16%
IMDB	0.01%	0.00%	-1.06%	14.29%	30.95%	-0.03%	0.00%	-0.74%	-0.16%	0.52%	0.79%	0.38%	0.68%
SST-2	0.21%	-0.02%	1.06%	14.29%	33.58%	0.09%	-4.24%	0.00%	0.02%	-0.76%	-1.98%	4.71%	0.53%
QQP	-0.77%	-0.82%	3.19%	0.00%	7.48%	0.09%	-0.41%	-2.21%	0.00%	-1.63%	-2.85%	1.89%	-0.19%
STS-B	-0.12%	0.04%	5.85%	57.14%	13.07%	0.13%	0.03%	-1.23%	0.05%	0.00%	1.15%	3.15%	1.48%
MRPC	-0.21%	0.02%	2.66%	21.43%	13.07%	-0.01%	-0.01%	0.25%	-0.11%	-0.23%	0.00%	1.77%	0.22%
BoolQ	-0.35%	-0.36%	-2.66%	21.43%	33.58%	-0.15%	-0.05%	-0.62%	-0.03%	1.18%	-1.24%	0.00%	-0.26%
MultiRC	-0.37%	-0.20%	4.26%	71.43%	10.38%	-0.07%	-0.04%	-1.48%	0.04%	0.70%	0.27%	1.64%	0.00%

Table 3: Model-based task relevance. It is computed by performance improvement as Equation (4).

Corpus	#Train	#Dev	#Test	#Label	Metrics
NATURAL LANGUAGE INFERENCE					
MNLI	393K	20k	20k	3	Accuracy
QNLI	108k	5.7k	5.7k	2	Accuracy
RTE	2.5k	276	3k	2	Accuracy
WNLI	634	71	146	2	Accuracy
CB	250	57	250	2	Accuracy/F1
SNLI	549k	9.8k	9.8k	3	Accuracy
SENTIMENT CLASSIFICATION					
IMDB	25k	0k	25k	2	Accuracy
SST-2	67K	872	1.8k	2	Accuracy
SIMILARITY AND PARAPHRASE					
QQP	364k	40k	391k	2	Accuracy/F1
STS-B	7K	1.5k	1.4k	1	Pearson/Spearman corr
MRPC	3.7k	408	1.7k	2	Accuracy/F1
QUESTION ANSWERING					
BoolQ	9.4k	3.3k	3.2k	2	Accuracy
MultiRC	5.1k	953	1.8k	2	$F1_a$/EM

Table 4: Summary of the 13 datasets.