# Enhancing Model Performance by Using Supporting-tasks Components

**Oded Mousai**
oded.mousai@mail.huji.ac.il

**Niv Amos**
niv.amos@mail.huji.ac.il

**Erez Badash**
erez.badash@mail.huji.ac.il

## 1 Introduction

In recent years, deep learning has emerged as a dominant paradigm in Natural Language Processing (NLP) and has achieved remarkable success in a wide range of language-related tasks. However, one of the significant limitations of deep learning models is their voracious appetite for annotated data. To effectively fine-tune a deep learning model for a specific NLP task and enable it to generalize well on unseen data, an extensive amount of labeled data is required. This poses a significant challenge for domains where obtaining annotated texts is difficult, expensive, or time-consuming, particularly due to the need for domain-expert annotators.

Some automatic techniques for addressing data shortage in NLP include Data Augmentation (increasing the diversity of examples without explicitly collecting new ones) and Zero-shot Learning (using models on tasks they were not explicitly trained for). However, these approaches have their disadvantages: Augmented data may not perfectly represent real-world examples, and zero-shot learning might not be as good as task-specific models due to domain differences. Another approach, Weak Supervision (using noisy or weakly annotated data) also has its respective drawbacks, including the risk of reduced model performance due to the presence of noisy labels.

In this study, we propose to deal with the data scarcity problem in a specific task by harnessing existing annotated datasets from related tasks. Our approach involves training a model concurrently on both the main task and these related tasks, which we term "supporting tasks." Our underlying belief is that "no data is redundant," as the model can extract valuable information from the supporting tasks to enhance its performance on the main task. This concept draws inspiration from the way humans learn and generalize knowledge across different domains. For example, learning to play the piano enhances hand-eye coordination and fine motor skills, which can translate into improved basketball performance through precise ball handling and shooting techniques. It is important to note that our goal is solely focused on succeeding in the main task, distinct from the multi-task learning paradigm in which it is desired to achieve good performance across all tasks simultaneously.

The problem of data shortage is particularly evident in the Medical NLP domain. Medical NLP aims to extract valuable information from clinical texts, electronic health records, medical literature, and other healthcare-related sources. It enables healthcare professionals to efficiently analyze and utilize vast amounts of unstructured medical data, potentially leading to improved diagnostics, treatment planning, and medical knowledge discovery. However, clinical data is inherently sensitive and tightly regulated, causing restricted access and licensing constraints, which results in small and fragmented datasets across institutions.

For these reasons, we choose to investigate our approach on clinical NLP tasks: Medication Attribute Extraction as the main task and Clinical Sense Disambiguation as the supporting task. To bolster the validity of our claim, we also experimented with our approach on different domain tasks: three GLUE tasks, which each task taking turns as the main task while the other two acted as supporting tasks.

Our code is available at: https://github.com/NivAm12/Enhancing-By-Subtasks-Components/tree/main

## 2 Data

### 2.1 Medication Attribute Extraction

Medication Attribute Extraction is a task that aims to automatically identify and extract specific attributes or properties associated with medications mentioned in unstructured text, such as medical records, clinical notes, or drug descriptions. In this work, we focus on five such attributes: Dosage (amount prescribed), Route (administration method), Frequency (how often), Duration (length of treatment), and Reason (purpose of use). Solving this task typically involves two steps: Named Entity Recognition (NER) is first used to identify the spans of medications and attributes, and then Relation Extraction (RE) is employed to classify the relationship between each medication and its corresponding attributes.

For training and validation, we used the data from the National Clinical NLP Challenges [1] that includes annotations for this task. We extracted 261 clinical notes and segmented each into shorter snippets of size 51 tokens. We filtered out snippets that included no medication annotations, and medication or attributes that had a mistake in their annotation. This resulted in a dataset of 2277 samples. Note that each sample may contain multiple medication mentions and their corresponding attributes. For the NER task, we processed this dataset to include a label for each token following the BIO (Beginning, Inside, Outside) format. For the RC task, we extracted from this dataset triplets of positive samples (snippet, medication, attribute) where the medication and attribute are truly related, and for each added a negative sample in which the medication and attribute are not related. For the test, we used data from (Agrawal et al., 2022) which is available in Hugginface [2].

### 2.2 Clinical Sense Disambiguation

The task of Clinical Sense Disambiguation (CSD) involves taking a medical note and an abbreviation present in the note and expanding the abbreviation to its full definition based on the context provided in the note, enabling a better understanding of the medical information. We formulate this task as a binary classification task: two sentences are given, one with the abbreviation and the other with a possible expanded term, and the goal is to classify whether the sentences are related or not. We took data from The Clinical Acronym Sense Inventory dataset (Moon et al., 2014), processed it to support the formulation above, and augmented it with negative samples.

### 2.3 GLUE tasks

We used the following datasets from GLUE (Wang et al., 2018) for additional experiments: **MRPC** (Dolan and Brockett, 2005), a dataset for semantic equivalence task, **RTE**, a dataset for textual entailment task, and **STSB** (Cer et al., 2017), a dataset for natural language inference (NLI) task.

Table 1 displays the dataset sizes.

| Task | Train examples | Validation examples |
|---|---|---|
| NER | 2000 | 200 |
| RC | 16000 | 1600 |
| CSD | 16000 | 1600 |
| MRPC | 2400 | 400 |
| RTE | 2400 | 400 |
| STSB | 2400 | 400 |

Table 1: Number of train and validation examples per task.

## 3 Methods

### 3.1 Multi-Head Architecture

To realize our idea of training a main task with supporting tasks, we propose a multi-head architecture. This architecture consists of a pre-trained language model serving as the backbone model, and a separate head for each task which is connected to its last layer. In each step of the training procedure, a batch of samples for each task is inserted through the backbone model and its relevant head, and then a joint weighted loss is calculated. Model architecture can be viewed at Fig. 1.

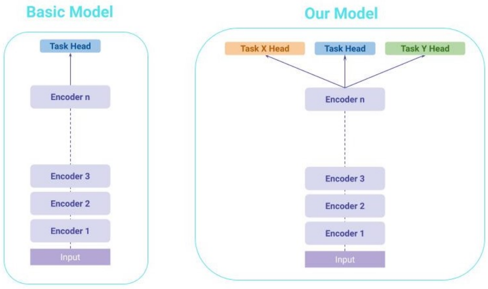

Figure 1: Illustration of our multi-head architecture (right) against a basic one (left)

### 3.2 Experiment 1

We chose as a backbone model PubMedBERT (Gu et al., 2020), a BERT model which was pre-trained only on in-domain texts (PubMed abstracts and full

[1] https://portal.dbmi.hms.harvard.edu/projects/n2c2-nlp/
[2] https://huggingface.co/datasets/mitclinicalml/clinical-ie

texts). We chose the following heads for our clinical tasks: for the NER task we took a CRF model and for the RC and CSD tasks a Linear Classifier with a single output for each. For the RC task, we used the idea of entity markers (Baldini Soares et al., 2019) in which four reserved words are added to the text to mark the beginning and end of each entity mention, and insert to the RC head only the embedding of the start word of each entity. For the CSD task, we put the special tokens "<start>" and "<end>" around the abbreviation and the full term, to encourage the model to pay attention to those terms. We assigned weights of 0.4, 0.4, and 0.2 to the NER, RC, and CSD tasks, respectively, considering the significance of the first tasks for the main task. We used a batch size of 64 for the RC and CSD tasks, and 8 for the NER task. We fine-tuned our multi-head model for 30 epochs. As a baseline, we trained the NER and RC models separately (including a separate backbone model) with no other task, with the same hyper-parameters as above. We perform an evaluation using the F1 score, by considering the proportions of token spans that are shared between each of the predictions and the true annotation.

### 3.3 Experiment 2

Here we took a regular BERT as the backbone model and fine-tuned simultaneously the three tasks from the glue datasets. We compared our multi-head model to baseline models that fine-tuned separability for each task. We used batch size=8, learning rate=1e-5, epochs=7, and all tasks loss weighted equally. We evaluated MRPC and RTE tasks using accuracy and STSB using MSE.

## 4 Results

In experiment 1 we got scores of 0.1 and 0.03 for our multi-head model and the baseline, respectively. By the outputs of the CRF layer, it seems that the NER task failed. We tried to understand these poor results by trying various hyperparameters, checking the metric score, etc. We suspect that this is a data problem, because our multi-head model that trained on more tasks achieved slightly better results then the baseline. In experiment 2 our multi-head model achieved similar or better scores than the baselines (see Table 2), suggesting that it managed to leverage useful information across the tasks. Considering that the baseline consists of three separate models, this result leads to significant parameter savings. More experimental figures can be viewed on "Weights and Biases" platform, [3] for Section 3.2 and [4] for Section 3.3.

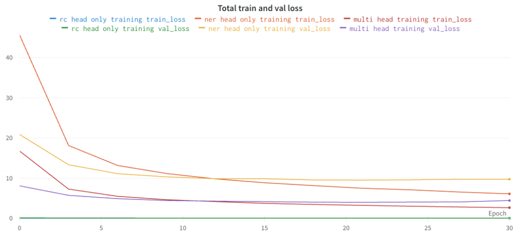

Figure 2: Train and validation losses for Experiment 1

| Task | Model | Metric | Score |
|---|---|---|---|
| MRPC | Baseline | Accuracy | 0.79 |
| | Multi-head-model | Accuracy | **0.80** |
| RTE | Baseline | Accuracy | 0.66 |
| | Multi-head-model | Accuracy | 0.66 |
| STSB | Baseline | MSE | **0.36** |
| | Multi-head-model | MSE | 0.39 |

Table 2: Results for Experiment 2 on three GLUE tasks. It can be seen that our multi-head model achieved similar or better scores than the baselines.

[3] Experiment 1 dashboard
[4] Experiment 2 dashboard

## References

Monica Agrawal, Stefan Hegselmann, Hunter Lang, Yoon Kim, and David Sontag. 2022. Large language models are few-shot clinical information extractors.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy. Association for Computational Linguistics.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Bill Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Third International Workshop on Paraphrasing (IWP2005)*.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2020. Domain-specific language model pretraining for biomedical natural language processing. *CoRR*, abs/2007.15779.

Sungrim Moon, Serguei V. S. Pakhomov, Nathan Liu, James O. Ryan, and Genevieve B. Melton. 2014. A sense inventory for clinical abbreviations and acronyms created using clinical notes and medical dictionary resources. volume 21 2, pages 299–307.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding.