KU_ai at MEDIQA 2019: Domain-specific Pre-training and Transfer Learning for Medical NLI

Cemil Cengiz **Ulaş Sert** **Deniz Yuret**
Koç University
Artificial Intelligence Laboratory
İstanbul, Turkey
`ccengiz17,usert17,dyuret@ku.edu.tr`

Abstract

In this paper, we describe our system and results submitted for the Natural Language Inference (NLI) track of the MEDIQA 2019 Shared Task (Ben Abacha et al., 2019). As KU_ai team, we used BERT (Devlin et al., 2018) as our baseline model and pre-processed the MedNLI dataset to mitigate the negative impact of de-identification artifacts. Moreover, we investigated different pre-training and transfer learning approaches to improve the performance. We show that pre-training the language model on rich biomedical corpora has a significant effect in teaching the model domain-specific language. In addition, training the model on large NLI datasets such as MultiNLI and SNLI helps in learning task-specific reasoning. Finally, we ensembled our highest-performing models, and achieved 84.7% accuracy on the unseen test dataset and ranked 10th out of 17 teams in the official results.

1 Introduction

Natural Language Inference (NLI) is one of the central problems in artificial intelligence. It requires understanding two input sentences and forming an inference relationship between them. Concretely, given a premise sentence p, and a hypothesis sentence h, NLI is the task of determining the inference relationship from p to h. In MedNLI, this relationship is one of the *neutral*, *entailment* and *contradiction* labels. Therefore, our task can be considered as a three-class sentence pair classification problem.

In previous research, sequence encoders connected with a classifier head have been commonly used as NLI systems (Conneau et al., 2017). Traditionally, the encoder layer has been an RNN-based model such as LSTM (Hochreiter and Schmidhuber, 1997). Vaswani et al. (2017) proposed the Transformer as an alternative model to the RNN. Since the Transformer is based on a self-attention mechanism rather than recurrent layers, it is much faster to train in parallel and can capture distant dependencies better. Therefore, the recent models originated from Transformer replaced the RNN-based encoders in many systems trained for natural language understanding tasks such as NLI, Question Answering, Common Sense Reasoning (Radford et al., 2018), and Neural Machine Translation (Lakew et al., 2018). As a Transformer based model, BERT uses self-attention to capture the relationships within the text during encoding, which can include one or more sentences (Devlin et al., 2018). Therefore, it can learn a joint representation for a premise-hypothesis pair, which can be fed to a classifier layer to predict the inference relation between them.

Recent studies have explored different ways of inference prediction instead of a straightforward classifier layer. Liu et al. (2018) proposed to use an answer module performing multi-step inference by iteratively refining its prediction. Likewise, models that aim to solve multiple problems simultaneously have gained attention due to their impressive performances (Liu et al., 2019). However, we concentrated on a single task, and wanted to keep the prediction layer simple. Therefore, we used neither of these approaches.

To succeed in NLI, a system must have strong reasoning skills and a good understanding of the language (MacCartney, 2009). If a large annotated dataset is available, the system can be trained from scratch for NLI, learning both the language and reasoning simultaneously. However, such data is often not available. Without seeing many syntactic variation and inference relation combinations, learning both is a hard task. Separating the two by training a language model first, and adjusting it for NLI later is a more effective approach. Due

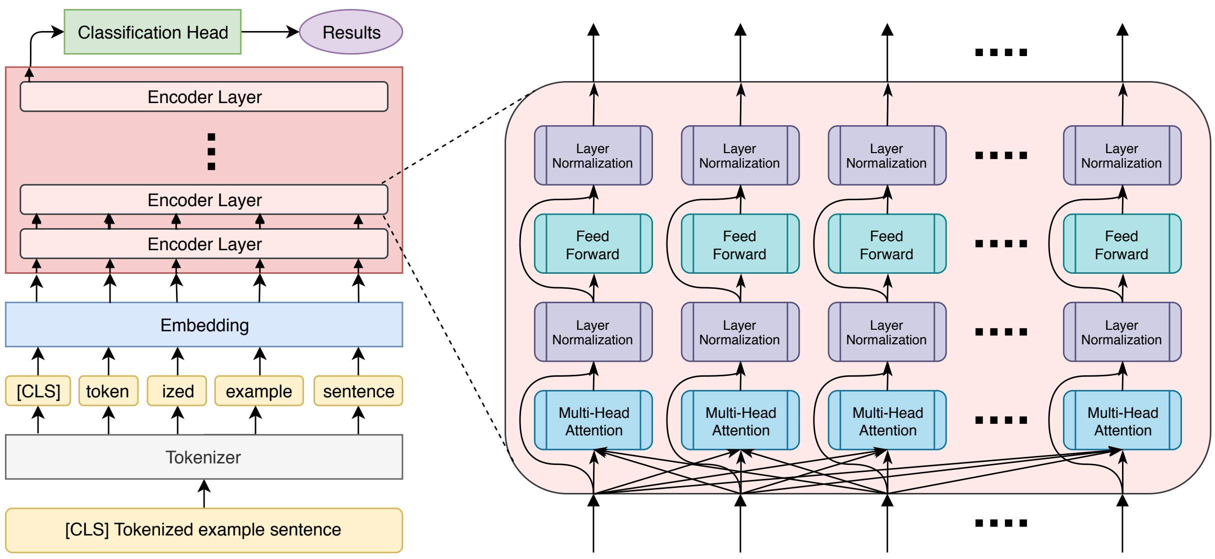

Figure 1: Baseline model architecture. On the left is the overview of the model, which includes the tokenizer, embedding layer, BERT encoder, and the classification head. The BERT encoder consists of twelve encoder layers. On the right are the details of what an encoder layer consists of. Each input and output of an encoder layer corresponds to a single token. Modules that are side-by-side share the same weights, only differing in inputs.

to the development of powerful language models that can be trained on unlabeled data in an unsupervised manner, many pre-trained context-aware encoders such as BERT (Devlin et al., 2018) and GPT (Radford et al., 2018) are publicly available. Most notably, Devlin et al. (2018) showed that BERT can be effectively used on many natural language understanding tasks, including NLI. Combining a pre-trained BERT encoder with a task-specific head, and then fine-tuning the entire model on the target task achieved state-of-the-art results on a number of tasks.

Directly applying BERT to NLI yields high accuracy if the dataset is large, and from a general domain (Phang et al., 2018). However, the dataset used in this shared task, MedNLI (Romanov and Shivade, 2018), is based on clinical notes (i.e. patient histories) and limited by size, thus it is a particularly challenging NLI task. To address this problem, we started with BERT, trained it on large NLI datasets, such as MultiNLI (Williams et al., 2018) and SNLI (Bowman et al., 2015), to support the inference reasoning, and then trained it further on our target task, MedNLI. This kind of intermediate training was shown to be effective when BERT is trained on a target with limited data (Phang et al., 2018). We call our approach *two-stage transfer learning*. In the first stage, we transfer the knowledge of the task, NLI, into our pre-trained model. During the second stage, we specialize our model on the MedNLI dataset. We hypothesized that this approach will produce higher accuracy on MedNLI compared to direct application of BERT.

By conducting extensive experiments, we explored the impact of different pre-trained model weights and transfer learning strategies. As a result, we signficantly improved the performance of BERT on MedNLI by initializing it from weights pre-trained on corpora close to clinical domain and applying two-stage transfer learning.

2 Model

Our baseline model is a BERT encoder (Devlin et al., 2018), combined with a classification head. The head outputs probabilities from a three-way softmax, corresponding to the three possible labels a sentence pair can have. The overview of this architecture can be seen from Figure 1.

2.1 BERT Encoder

We used BERT to encode our input tokens. Utilizing Transformer layers and self-attention (Vaswani et al., 2017), BERT looks at how the tokens are related, and outputs a hidden vector for each token inside of the input sequence. Therefore, BERT can process the sentence pair together as a whole sequence and output the encoded representation for all of it in one pass.

One of the reasons why we opted for a model

Weight Set	Model Size	Corpus Domain	Corpus Size
BERT_{BASE}	110M	Wikipedia	2.5B
		Books	0.8B
BERT_{LARGE}	340M	Wikipedia	2.5B
		Books	0.8B
BioBERT	110M	Biomedical	18.0B
		(+ BERT_{BASE})	
SciBERT	110M	Biomedical	2.5B
		CompSci	0.6B

Table 1: Comparison of pre-trained BERT weight sets.

like the BERT encoder is that it completely avoids relying on recurrence and convolution operations. Replacing those with simple operations of self-attention (e.g. plain matrix multiplications) makes it more parallelizable and thus faster to train. Another reason is the strong starting point of BERT. It is a language modeling architecture, successfully trained on massive corpora. Lastly, there are a number of pre-trained weight sets for BERT from different domains. These weights are publicly available, which gave us the ability to test different starting points with minimal tinkering.

2.2 Pre-Trained Weights

We have considered four different pre-trained BERT weights as our starting point. From Devlin et al. (2018)'s work, we examined BERT_{BASE} and BERT_{LARGE}. Both models were trained from scratch on English Wikipedia articles and books, and have the same vocabulary. However, the latter has more than triple the parameter size of the former.

Next we looked at BioBERT (Lee et al., 2019), which was trained on biomedical text. However, rather than randomly initialized weights, BioBERT utilizes BERT_{BASE} as its starting point. Nevertheless, both versions use the same vocabulary for tokenization.

Lastly, we included SciBERT (Beltagy et al., 2019) in our experiments. It was trained on large-scale, annotated data from the scientific domain. It has the same size as BERT_{BASE}, and was trained from scratch, but uses a different vocabulary. Although the vocabulary size is the same as the original BERT, they have a total of 42% overlap between them. A direct comparison of the pre-trained BERT weights can be found in Table 1.

2.3 Input and Tokenization

During its pre-training process, one of the tasks BERT has to learn is the next sentence prediction (Devlin et al., 2018). It is a two-sentence classification task which requires predicting if the given sentences follow each other in the original text. Since our task is also a two-sentence classification task, we mimicked the inputs BERT receives during the pre-training. Thus, our input sentence pair is represented as ”[*CLS*] *Premise* [*SEP*] *Hypothesis* [*SEP*]”. The [*CLS*] and [*SEP*] are special tokens, denoting ”Classification” and ”Seperator” respectively. Since those are the tokens used during pre-training, they are kept in the same format to fully utilize BERT encoder's understanding of sentence pairs.

BERT uses WordPiece tokenizer (Wu et al., 2016), which divides the words in the input sequence into wordpieces (i.e. subwords). It maintains a good compromise between the character based representation's flexibility and the word based representation's efficiency. This balance improves the overall accuracy of the natural language system. Moreover, since it splits the infrequent words into wordpieces, it naturally handles the rare words problem (Wu et al., 2016).

2.4 Classification Head

To transform the token representations obtained from the encoder into label predictions, we used a small classification head. Following the convention of Devlin et al. (2018), we used the representation of the first token, [*CLS*], as the summary of the whole sequence. Then, this vector is linearly projected into a three-dimensional space such that each dimension represents the score for a label. Finally, a softmax operation is applied to convert the scores into class probabilities.

3 Datasets

The main dataset used in this shared task is MedNLI. Additionally, we used MultiNLI and SNLI to improve our accuracy via transfer learning.

3.1 MultiNLI and SNLI

MultiNLI and SNLI (Williams et al., 2018; Bowman et al., 2015) are general domain datasets, containing significantly more example pairs than MedNLI. The MultiNLI training dataset consists of five different genres of written and spoken

Dataset	Genre	Training Set Size
MedNLI	Patient Records	11,232
MutliNLI	Fiction	77,348
	Government	77,350
	Slate	77,306
	Telephone	83,348
	Travel	77,350
	Total	392,702
SNLI	Image Captions	550,152

Table 2: Comparison of NLI datasets by their genres and sizes.

1	Her a[** Location **]e and PO intake have been normal.
2	on [* * 1 − 31 * *] Dr. [* * Name (NI) **] documented that there was 1 positive ascitic fluid culture

Table 3: Two partial examples from MedNLI sentence pairs. Note the de-identification artifacts.

1	her, a, [, *, *, location, *, *,], e, and, po, intake, have, been, normal, .
2	on, [, *, *, 1, −, 31, *, *,], dr, ., [, *, *, name, (, ni,), *, *,], documented, that, there, was, 1, positive, as, ##cit, ##ic, fluid, culture

Table 4: The results of tokenizing the examples provided in the Table 3. The commas are used to separate different tokens.

1	Her ae and PO intake have been normal
	her, ae, and, po, intake, have, been, normal, .
2	on Dr. documented that there was 1 positive ascitic fluid culture
	on, dr, ., documented, that, there, was, 1, positive, as, ##cit, ##ic, fluid, culture

Table 5: The results of pre-processing the examples provided in Table 3, and their respective tokenizations.

English, such as telephone conversations, travel guides and press releases from government websites. All examples in the SNLI training dataset are created from image captions, hence SNLI is regarded as a single genre. We have only used the training sets of MultiNLI and SNLI in our experiments to teach our model general domain NLI. A detailed comparison of the NLI datasets can be found in Table 2.

3.2 MedNLI

Our target dataset is MedNLI (Romanov and Shivade, 2018). We have used the provided splits without change. We trained our models on the training set, evaluated them on the development set. However, we did not use the testing set during training or hyperparameter selection.

MedNLI is created from text in the clinical domain, particularly patient records. To keep the confidentiality of various parties, names (of patients and places) and dates in the source texts have been de-identified. Therefore, the dataset contains artifacts, some examples of which are shown in Table 3.

In the example sequences, "a[** Location **]e", "[* * 1 − 31 * *]" and "[* * Name (NI) **]" are de-identification artifacts. This hurts the performance of our model since when WordPiece tokenizer segments the de-identified words into subwords, an excessive number of tokens are generated. The tokenization outputs of the examples are shown at Table 4.

Since BERT tokenizer treats the special characters such as "[" and "*" as wordpieces, the resulting tokenization contains an inflated number of unnecessary tokens. These tokens include the special characters and de-identified place-holders such as "Name", "Location". We suspected that WordPiece tokenizer makes the de-identification artifacts harmful to the performance since their tokenizations introduce too many erroneous tokens. To validate this observation, we performed some experiments where we removed the de-identified tokens from the MedNLI dataset while pre-processing. Resulting text and tokenizations of the sample sequences after the removal can be found in Table 5. As a result, the accuracies improved as shown in Section 5. Hence, we conducted the remaining experiments by performing this pre-processing step.

4 Training Strategy

4.1 Sequential Transfer Learning

Following the advice of Romanov and Shivade (2018), we experimented with transfer learning techniques to boost the accuracy of our model. In

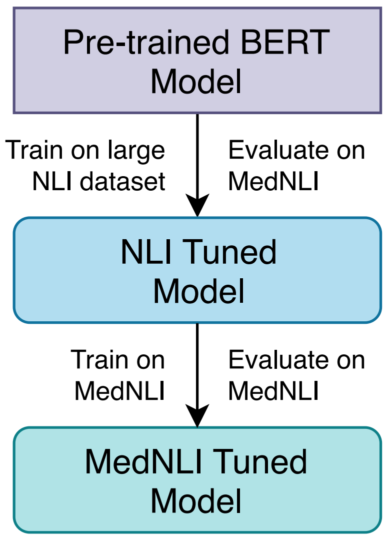

Figure 2: Two-stage sequential transfer learning strategy.

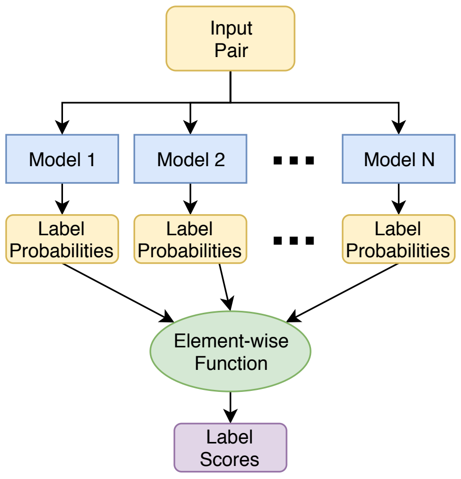

Figure 3: Ensembling procedure of N different models for an input premise-hypothesis pair.

fact, our baseline approach is itself a type of sequential transfer learning. We take a pre-trained BERT, append a classifier head on top of it, and fine-tune the whole model on the MedNLI training dataset. Since BERT is a powerful sequence encoder, this approach alone yielded better results on the MedNLI development dataset compared to the published baselines (Romanov and Shivade, 2018). However, because the dataset is limited in size, performances of these approaches are restricted. Therefore, instead of training our model directly on the target task, we added an intermediate training step. During this step, the model is trained on a large NLI dataset from a general domain such as MultiNLI and SNLI. Our aim is to help the model learn task specific reasoning skills using the large number of training examples. Then, the model is further fine-tuned on the MedNLI training dataset so that it can adapt its parameters to the clinical domain and MedNLI style. We call this strategy *two-stage sequential transfer learning*, which is shown in Figure 2. In both stages of the learning, the model is trained until its accuracy on the MedNLI development set is maximized.

Since there are multiple large and general domain NLI datasets available, we wanted to leverage them. Therefore, we also experimented with three-stage transfer learning. In the first two stages, we trained our model on MultiNLI and SNLI successively. In the third stage, we finally trained it on MedNLI. However, this configuration yielded slightly worse results compared to two-stage transfer, as discussed in Section 5. Therefore, we changed our method and utilized an ensembling approach to combine the benefits of the available datasets.

4.2 Ensembling

We perform ensembling on independently trained models to get the benefit of multiple datasets. First, these models are fully trained by two-stage transfer learning with different general domain datasets. After that, we compute a set of label probabilities for the sentence pairs using these models. Then, we combine the probabilities with an element-wise operation such as *sum*, *product*, or *max* to obtain a single score for each label. Figure 3 summarizes this process for a single input. Finally, we report the label corresponding to the highest score as our prediction for each example. This approach might be regarded as a soft version of *majority voting*, a commonly used ensembling method. As Section 5 shows, this approach yielded the best results obtained by our models.

5 Results and Discussion

5.1 Implementation Details

We used PyTorch (Paszke et al., 2017) as our deep learning framework and the BERT implementation provided by Hugging Face[1]. Our models are based on the *BertForSequenceClassification* class. We used the *BertAdam* optimizer from the same

[1]https://github.com/huggingface/pytorch-pretrained-BERT

Training Strategy	Training Set	Sequence Length	Batch Size	Optimizer	Initial Learning Rate
Direct Training					
	MedNLI	256	16	$BertAdam$	2×10^{-5}
Task pre-training					
	MultiNLI	256	32	$BertAdam$	2×10^{-5}
	SNLI	256	32	$BertAdam$	2×10^{-5}
Domain fine-tuning					
	MedNLI	256	16	$BertAdam$	2×10^{-5}

Table 6: The hyperparameters for different training settings.

repository, which imitates the Adam implementation of the original BERT.

We always evaluated our models according to the accuracy on the MedNLI development set. For all experiments, we trained a model until its accuracy in the last four epochs did not improve over its best accuracy. If a model kept improving, we stopped the training after 80 epochs. All model weights are updated during the training phases of the experiments.

The training procedure was stable, there were no serious failure cases with unexpectedly low accuracies. Nevertheless, to mitigate the possible negative effects of the randomness on the optimization, we repeated each experiment with three different seeds and selected the run with the best accuracy.

The primary hyperparameters of our model are the sequence length of the encoder layers, the batch size, and the initial learning rate of the optimizer, *BertAdam*. We kept the remaining hyperparameters such as dropout rate the same as the original implementation. To determine the sequence length, we counted the number of resulting tokens from MultiNLI, SNLI, and MedNLI sentence pairs after tokenization. Since the maximum token count is 256, we naturally set the sequence length to 256. When we trained the model on MedNLI, we used batches with size of 16. In contrast, when a large dataset (e.g. MultiNLI and SNLI) is used, we used batch size of 32 to speed up the training process. In the initial experiments, we tried $2 * 10^{-5}$, $3 * 10^{-5}$, and $5 * 10^{-5}$ as *BertAdam*'s initial learning rate which resulted in very similar accuracies. Nevertheless, since $2 * 10^{-5}$ generally yielded slightly better results, we conducted the remaining experiments with this initial learning rate. Table 6 summarizes the hyperparameters used in the training phases.

Weight Set	Dev Accuracy
BERT_{BASE}	82.3%
BERT_{LARGE}	82.9%
BioBERT	83.4%
SciBERT	**83.7%**

Table 7: Results of directly training on MedNLI data, starting from various pre-trained weight sets. The highest accuracy is indicated with bold.

5.2 Experiments

We have conducted a number of experiments to test our model and the effectiveness of different training strategies. We report the resulting percent accuracies on the MedNLI development dataset.

To start with, we trained our model on MedNLI directly. We initialized it with various pre-trained weights to compare their performances. The results of the experiment can be seen from Table 7. It shows that the weights pre-trained on domains related to the task, such as biomedical or scientific data, have a noticeable advantage over the weights obtained from general text corpora, such as Wikipedia. Moreover, the effect of pre-training is more significant compared to the model size. BioBERT and SciBERT surpassed BERT_{LARGE} although they are three times smaller.

Next, we tested the two-stage sequential transfer learning method using all combinations of pre-trained weights and rich NLI datasets mentioned before. Table 8 shows the results of this experiment. As expected, all models benefit from two-stage transfer learning. Moreover, the trend of specialized pre-trained weights having an advantage continues on this experiment as well. However, BioBERT outperforms SciBERT, unlike the pre-

Weight Set	MultiNLI	SNLI
BERT$_{BASE}$	82.9%	82.4%
BERT$_{LARGE}$	83.7%	84.4%
BioBERT	85.6%	**85.8**%
SciBERT	85.4%	85.6%

Table 8: Development set accuracies achieved by performing two-stage sequential transfer learning, utilizing different intermediary datasets, and starting from various pre-trained weights. The highest accuracy is indicated with bold.

	MultiNLI	SNLI
BERT$_{BASE}$	+0.7%	±0.0%
BERT$_{LARGE}$	+0.6%	+2.2%
BioBERT	+1.2%	+0.6%
SciBERT	+0.4%	+1.3%

Table 9: Accuracies gained by pre-processing the MedNLI dataset, after performing two-stage sequential transfer learning with different starting points and datasets.

vious results. We suspect that since BioBERT is pre-trained starting from BERT$_{BASE}$, it benefits more from general domain task training.

In order to test the effect of pre-processing described in Section 3.2, we repeated the two-stage training experiment without removing the de-identification artifacts. We compared the results of this experiment with the previous results to see how the accuracy is affected. The improvements obtained from the pre-processing can be found in Table 9. It increases the accuracy in all cases, except for BERT$_{BASE}$ - SNLI, where the accuracy is unchanged. Note that all other experiments are conducted with the pre-processing is enabled.

We have also tested the performance of three-stage sequential transfer learning on BioBERT. Training on SNLI and MultiNLI sequentially before MedNLI produced lower accuracies compared to the two-stage transfer learning experiment. Moreover, switching the training order of SNLI and MultiNLI did not change the resulting accuracy. We suspect that further training on a second intermediate dataset brings the model closer to a worse local optimum. A comparison between different training strategies on BioBERT can be found in Table 10.

Lastly, we experimented with ensembling. We tested four ensemble models, one for each pre-

Training Strategy	Dev Accuracy
Direct Training on MedNLI	83.4%
Two-Stage Transfer:	
MultiNLI $\rightarrow$ MedNLI	85.6%
SNLI $\rightarrow$ MedNLI	**85.8**%
Three-Stage Transfer:	
MultiNLI $\rightarrow$ SNLI $\rightarrow$ MedNLI	84.9%
SNLI $\rightarrow$ MultiNLI $\rightarrow$ MedNLI	84.9%

Table 10: Single model development set accuracies of different training strategies, starting from BioBERT. The highest accuracy is indicated with bold.

Weight Set	Sum	Product	Max
BERT$_{BASE}$	83.3%	83.4%	83.3%
BERT$_{LARGE}$	85.1%	85.0%	85.2%
BioBERT	**86.1**%	86.0%	**86.1**%
SciBERT	85.9%	85.8%	85.8%

Table 11: Development set accuracies resulting from ensembling two models starting from the same pre-trained weights. The highest accuracies are indicated with bold.

trained BERT variant. For each of these starting points, we combined the two models obtained from the two-stage transfer learning experiment. One of these models is trained with MultiNLI, the other with SNLI. We tried three different element-wise operations for ensembling, and compared their effects. Table 11 shows that this approach yielded better results compared to the two-stage transfer methods. Therefore, we effectively combined the benefits of training on different, rich datasets.

Among all the experiments, the best result we obtained is 86.1% accuracy by ensembling BioBERT models trained with two-stage transfer learning. That accuracy was obtained by combining the output probabilities with element-wise sum operation.

Therefore, we participated in the shared task with that ensembled model. Consequently, our model achieved 84.7% accuracy on the unseen test dataset reserved for the shared task.

5.3 Error Analysis

Following Romanov and Shivade (2018), we conducted a similar error analysis on MedNLI development set to understand how the different methods improve the baseline. We chose BioBERT for

Ground Truth Label	**CON**		**ENT**		**NTR**	
Predicted Label	**ENT**	**NTR**	**CON**	**NTR**	**CON**	**ENT**
Direct Training on MedNLI	33	22	19	84	20	53
Two-Stage Transfer:						
MultiNLI → MedNLI	16	14	24	77	22	48
SNLI → MedNLI	23	14	15	65	18	63
Three-Stage Transfer:						
MultiNLI → SNLI → MedNLI	19	17	18	83	22	51
SNLI → MultiNLI → MedNLI	21	20	13	73	24	59
Ensemble Model:						
MultiNLI + SNLI with Sum	20	15	16	67	20	56

Table 12: Label breakdown of errors made in MedNLI development set by different models. All models were trained with different strategies starting from BioBERT. ”CON”, ”ENT” and ”NTR” label abbreviations mean ”Contradiction”, ”Entailment”, and ”Neutral” respectively.

Category	**ABB**	**MED**	**NUM**	**WOR**
Direct Training on MedNLI	32	126	30	43
Two-Stage Transfer:				
MultiNLI → MedNLI	24	113	29	35
SNLI → MedNLI	22	111	28	37
Three-Stage Transfer:				
MultiNLI → SNLI → MedNLI	23	116	30	41
SNLI → MultiNLI → MedNLI	25	116	30	39
Ensemble Model:				
MultiNLI + SNLI with Sum	23	106	30	35

Table 13: Category breakdown of errors made in MedNLI development set by different models. All models were trained with different strategies starting from BioBERT. The category abbreviations mean ”Abbreviation”, ”Medical Knowledge”, ”Numerical Reasoning”, and ”World Knowledge” respectively.

the analysis since it is the starting point of our highest-scoring model. We compared the errors made by the models resulting from direct training, two-stage transfer learning, three-stage transfer learning, and ensembling.

In the first study, we concentrated on analyzing the distribution of the misclassified examples over labels, whose results are shown at Table 12. First thing to notice is that the errors mostly originated from confusion between the entailment and the neutral labels. For all models, this confusion causes approximately 60% of the errors. Two-stage transfer results show that intermediate training on a large dataset helps in identifying the contradiction relation. The error counts of the ensemble model are lower than or around the averages of the models that are ensembled.

The second analysis is separating the errors into four broad categories. These involve ”Abbreviation”, ”Medical Knowledge”, ”Numerical Reasoning”, and ”World Knowledge”. ”Abbrevation” represents the existence of medical abbrevations critical to decode the inference relation. ”Medical Knowledge” refers to the requirement of reasoning with medical knowledge. ”Numerical Reasoning” denotes that the inference type depends on the value of the number(s) present in the sentence pair. ”Word Knowledge” indicates the need for common sense or general domain knowledge to understand the inference. We manually categorized each misclassified sentence pair.

Table 13 shows the results of the error categorization. ”Numerical Reasoning” errors are almost the same across all models. We believe that this is because the scale of the numerical values highly depends on the context. Adding general domain knowledge does not seem to help in learning numerical scales on the medical domain. As the two-stage transfer results show, the intermediate training on a general domain NLI dataset decreases the error rates on the remaining three categories. However, training with three-stage transfer learning does not improve the "Medical Knowledge" and "World Knowledge" categories as much as the two-stage transfer. Although getting poorer results after training with more data seems counter-intuitive, Romanov and Shivade (2018) observed a similar trend on transfer learning using SNLI, and the genres of MultiNLI. In both findings, the model performance does not directly correlate with the size of the intermediate training data.

Lastly, most of the improvements introduced by ensembling fall under "Medical Knowledge". Since the component models are trained on different datasets on intermediate training step, the errors they made on the MedNLI development set differ. We suspect that the models are not very confident in some of their "Medical Knowledge" errors, hence the other model may correct these mistakes to an extent.

6 Conclusion

In this paper, we showed that a pre-trained BERT encoder, combined with a classifier head forms a strong baseline for MedNLI task. More importantly, we also demonstrated that the model's accuracy can be remarkably improved by utilizing a two-stage transfer learning strategy. The success of the final model depends on the initial BERT weights, as well as the particular transfer learning method. Finally, we showed that ensembling two separate models trained on different NLI datasets is more effective than using these datasets to train a single model. Our best model, BioBERT ensembled, achieved 86.1% accuracy on the MedNLI development set, and 84.7% accuracy on the unseen test dataset reserved for the MEDIQA 2019's NLI Shared Task.

While empirical results show that contextualized sequence encoders enhanced with transfer learning are strong, their performances might be further improved with external knowledge integration. As future work, we would like to extend BERT's contextualized word vectors using semantic relationships.

Acknowledgments

Cemil Cengiz is supported by Huawei Turkey R&D Center through the Huawei Graduate Research Support Scholarship.

References

Iz Beltagy, Arman Cohan, and Kyle Lo. 2019. Scibert: Pretrained contextualized embeddings for scientific text. *Computing Research Repository*, arXiv:1903.10676.

Asma Ben Abacha, Chaitanya Shivade, and Dina Demner-Fushman. 2019. Overview of the mediqa 2019 shared task on textual inference, question entailment and question answering. In *Proceedings of the BioNLP 2019 workshop, Florence, Italy, August 1, 2019*. Association for Computational Linguistics.

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*. Association for Computational Linguistics.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Computing Research Repository*, arXiv:1810.04805.

Sepp Hochreiter and Jrgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Surafel Melaku Lakew, Mauro Cettolo, and Marcello Federico. 2018. A comparison of transformer and recurrent neural networks on multilingual neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 641–652, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Computing Research Repository*, arXiv:1901.08746.

Xiaodong Liu, Kevin Duh, and Jianfeng Gao. 2018. Stochastic answer networks for natural language inference. *Computing Research Repository*, arXiv:1804.07888.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. *Computing Research Repository*, arXiv:1901.11504.

Bill MacCartney. 2009. *Natural Language Inference*. Ph.D. thesis, Stanford University, Stanford, CA, USA. AAI3364139.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS-W*.

Jason Phang, Thibault Févry, and Samuel R. Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *Computing Research Repository*, arXiv:1811.01088.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alexey Romanov and Chaitanya Shivade. 2018. Lessons from natural language inference in the clinical domain. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1586–1596, Brussels, Belgium. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. 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, Volume 1 (Long Papers)*. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *Computing Research Repository*, arXiv:1609.08144.