Multi-Task Learning for Indirect Answer Classification using BERT

Luuk Kaandorp
Universiteit van Amsterdam
MSc Artificial Intelligence
luuk.kaandorp
@student.uva.nl

Damiaan Reijnaers
Universiteit van Amsterdam
MSc Artificial Intelligence
damiaan.reijnaers
@student.uva.nl

Casper Wortmann
Universiteit van Amsterdam
MSc Artificial Intelligence
casper.wortmann
@student.uva.nl

Abstract

In Multi-Task Learning (MTL) a single model is trained on multiple tasks simultaneously. As such, knowledge from within one domain can be used in another domain. In this paper a MTL model for the interpretation of indirect answers is introduced. We provide an analysis and comparison into the extent to which different auxiliary tasks aid their classification. We find that a textual entailment prediction task benefits performance, while other tasks harm it. Moreover, results show that information of the auxiliary task already gets encoded by the main task to some extent. Lastly, MTL models seem to be better able to generalize to underrepresented classes than pre-trained models.

1 Introduction

Machine Learning (ML) has been employed in many Natural Language Processing (NLP) problems recently, *e.g.* sentiment prediction and automatic summarization. Even though ML can solve many tasks, these tasks need to be explicitly created by humans. Seemingly trivial everyday activities, such as engaging in dialogue, often involve subconscious processes. These processes are complex to comprehend, and consequently, mould into a consistent set of formal rules for ML models to use. It is therefore a challenge to provide models with the intuition to solve such tasks. An example of such a task is indirect answer classification. For humans, an intuitive and straightforward interpretation of the response *"I would like Italian."* to the question *"Do you want to go for dinner?"* is *"Yes."* Likewise, the answer *"I prefer lunch first."* to the same question is interpreted as *"No."* However, for ML models this is not a trivial task.

In their recent work, Louis et al. (2020) propose the Circa dataset for indirect answer classification. This dataset consists of labelled polar question-answer pairs. They achieve good accuracy (around 85% test and development) using a pre-trained BERT model (Devlin et al., 2019). However, there is still some improvement to be made on this dataset.

In the last years, Multi-Task Learning (MTL) has regained interest within the research community. In MTL, a model is trained on additional tasks (*auxiliary tasks*) that provide underlying knowledge, which may assist the main task. However, auxiliary tasks do not necessarily need to be directly beneficial to the main task; additional tasks can also be formulated to allow for better generalization. Liu et al. (2019) have proven that MTL can be beneficial in NLP tasks. Their proposed MTL BERT model achieves new state-of-the-art results on the GLUE benchmark with 82.7% accuracy, boasting an impressive 2.2% absolute improvement over the base pre-trained BERT model.

In this paper, we investigate whether the application of MTL improves the performance of indirect answer classification on the Circa dataset. We experiment with different auxiliary tasks to provide further insight into which additional tasks suit MTL on indirect answer classification. Louis et al. (2020) aim to improve results by pre-training on auxiliary datasets, before targeting the Circa dataset. We will compare the effectiveness of MTL to pre-training. Concretely, we will address the following research questions:

1. What auxiliary tasks encode additional knowledge that aids the classification of indirect answers to polar questions?
2. To what extent is the knowledge required for the auxiliary task already captured by the main task?
3. To what extent does Multi-Task Learning improve performance compared to pre-training?

On the one hand, the results show that a textual entailment prediction task aids the classification

of indirect answers in a MTL setup, boasting it up with a 2.34% absolute accuracy improvement. On the other hand, the other proposed auxiliary tasks harm performance compared to the baseline model. If this is the case, this indicates that the main task does not 'capture enough information' to perform well in this task. This seems to indicate that these auxiliary tasks are not *relevant* for indirect answer classification. Moreover, the results show a small improvement in performance of MTL models compared to pre-trained models, with their main advantage being better generalizability to underrepresented target classes.

2 Related Work

In early work, de Marneffe et al. (2009) propose the Indirect Question-Answer Pair (IQAP) corpus. IQAP is a relatively small dataset consisting of 215 indirect question-answer pairs. This dataset was its time well ahead. At the time, models were not able to effectively work with the dataset. Nowadays, the dataset is too small for more advanced implementations, such as Transformer-based models.

In a more recent study, Louis et al. (2020) introduce the Circa dataset. This is a more extensive dataset, consisting of 34,268 question-answer pairs. These pairs consist of 3,431 unique questions with up to 10 answers each. This dataset is sufficiently sized to accommodate current-day models with enough data to adequately learn the indirect answer classification task. State-of-the-art Transformer models, such as BERT, achieve already high accuracy on this dataset.

Multi-Task Learning dates back as far as 1997 (Caruana, 1997) and takes inspiration from the theory that humans often apply knowledge from previous tasks to help learn a new task. In ML, this translates to training models jointly on multiple tasks, so that the model can benefit from knowledge learned in one task to excel in another task.

More recently, Liu et al. (2019) present the Multi-Task Deep Neural Network (MT-DNN), which is an adaption of the BERT Transformer model. The model is applied to the GLUE benchmark (Wang et al., 2018) and boasts an impressive 82.7% accuracy, which is an absolute improvement of 2.2% over a pre-trained BERT model.

3 Methodology

This section describes the methodology for the research conducted in this paper.

3.1 BERT Model

Following Devlin et al. (2019), in this paper we will use BERT as the basis for our model. Specifically, we use the *bert-base-uncased* version from the Huggingface library. This model consists of 12 layers, a hidden dimension of 768, 12 attention heads, and 110 million parameters.

To make the model capable of MTL, we add a list of linear classification layers for the different tasks. This means that each task has its own linear classification layer. Figure 1 provides a visualization of the new multi-task BERT model.

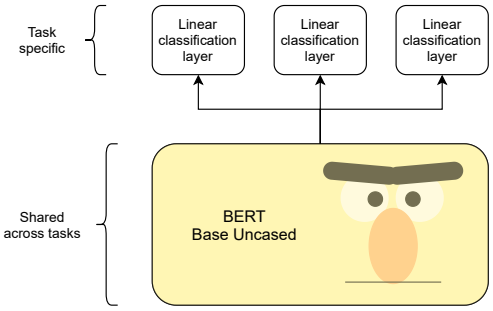

Figure 1: Schematic of the multi-task BERT model.

3.2 Settings

Following Louis et al. (2020), in this paper we will also make a distinction between the *matched* and *unmatched* settings. In the matched setting, the response scenario is assumed to be seen during training. This is achieved by randomly dividing the Circa dataset into 60% training and 20% each for development and testing. In the unmatched setting, we assume that the scenario is not seen during training. This is achieved by reserving one scenario for testing while using 80% of the remaining nine scenarios for training and 20% for development.

We also distinct between the *strict* and *relaxed* labels. In the strict setting, we use all 6 different classes as labels. In the relaxed setting, some classes are combined, resulting in 4 different classes.

3.3 Datasets

For our experiments, we will make use of several different datasets. These are explained in their respective subsections.

3.3.1 Circa

The Circa dataset by Louis et al. (2020) consists of labelled question-answer pairs, focused on in-

direct answer classification. Instances classified as *"Other"*, *"I'm not sure"*, or *N/A* were removed from the dataset. The same splits were used as in the original paper, namely `60%` train, `20%` validation, and `20%` test in the matched setting. For the unmatched setting, 8 scenarios were used as the train set, 1 scenario for validation, and 1 scenario as the test set. As said before, there are either 4 or 6 classes, depending on the setting.

3.3.2 Indirect Question-Answer Pair corpus

The IQAP dataset, as proposed by de Marneffe et al. (2009), contains labelled indirect question-answer pairs. We assign labels based on majority judgements over the different categories ('definite yes', 'probable yes', 'definite no', 'probable no'). This results in a total of 4 classes. The dataset already provides a train and development split. The development split is halved into a development and test set. In this paper, IQAP is used for the auxiliary task of indirect answer classification.

3.3.3 BoolQ

The BoolQ dataset is a dataset for classifying polar question and answers. As described by the authors Clark et al. (2019), questions and answers are considered to be naturally occurring, since they were generated under unconstrained settings. The dataset consists of 15942 examples of questions and answers. Each question-answer pair is labelled true ('yes') or false ('no'), meaning that there are 2 classes. In this paper, the BoolQ dataset is used for the auxiliary task of binary question answering.

3.3.4 Multi NLI

The Multi-Genre Natural Language Inference (Multi NLI) corpus serves the topic of sentence understanding through Natural Language Inference. Proposed by Williams et al. (2018) the dataset is generated through crowd-sourced collection and contains 433k sentence pairs and their textual entailment. Each sentence pair is labelled either contradiction, entailment, or neutral. Therefore, there are 3 labels. In our setup, it is used for textual entailment prediction, as an auxiliary task.

3.3.5 Stanford Sentiment Treebank V2

The Stanford Sentiment Treebank V2 (SST-2) dataset is a dataset for sentiment classification, as proposed by Socher et al. (2013). It contains 215k phrases and corresponding sentiment. The labels are cast from a regression problem into a binary classification problem, with a threshold of `0.5`. This means there are 2 labels. In this paper, the SST-2 dataset is used for the auxiliary task of sentiment classification.

3.4 WordNet Topic Annotations

Intuitively, learning about topical information of indirect answers aids to their semantic understanding, *e.g.* a question "do you like pets?" followed by an indirect answer topically marked by 'cat' is likely positively answered. To ascertain whether such tasks indeed help, we have synthetically generated topic annotations for the indirect answers present in the Circa dataset, which we employ in a self-created auxiliary topic classification task.

The first step in the annotation process is to infer the most important word of every answer. We extract the noun with the highest `TF-IDF`-value (Ramos et al., 2003). For all extracted words, a lexical hypernym tree is then constructed, based on the WordNet database (Fellbaum, 1998), using NLTK (Loper and Bird, 2002). Trees are traversed to organize all lemmas (including synonyms) in an ordered list by frequency of occurrence, such that a matching subset of topics exists for every answer sample. By constraining the size of this 'pool' of most-seen lemmas, the maximum number of possible labels can be controlled. This way the topic labels remain as specific as possible, and a matching label for every answer is guaranteed, as all words at least share the same root node, 'entity,' which is the most frequently observed lemma. The annotation process, and its intermediary results, is described in more detail in Appendix A.

3.5 Experiments

This study addresses three research questions. The corresponding experiments are explained in their respective subsections.

3.5.1 Effect of Auxiliary Tasks

To answer the first research question, we will research which auxiliary tasks encode prior knowledge that aids the classification of indirect answers to polar questions. We will compare five different models:

- **BERT-YN**: pre-trained BERT model fine-tuned on the Circa dataset only.
- **BERT-IQAP-YN**: finetunes a pre-trained BERT model on the IQAP and Circa datasets in a multi-task fashion.

- **BERT-BOOLQ-YN**: finetunes a model on both the BoolQ and Circa datasets using MTL.
- **BERT-MNLI-YN**: model finetuned on both the Multi NLI and Circa datasets in a multi-task fashion.
- **BERT-SST2-YN**: finetunes a model on the SST-2 and Circa datasets using MTL.
- **BERT-TOPICS-YN**: finetunes a model on the annotated WordNet topic labels and Circa datasets using MTL.

The models are compared in terms of test and development accuracy across the different settings.

3.5.2 Knowledge Encoded by Main Task

To answer the second research question, we will research if a non-improvement of performance by introducing an auxiliary task entails that such knowledge is already encoded by training on the main task. We will compare the same five models for this experiment. However, this time only the classification layers of the auxiliary tasks are trained. These classification layers will therefore act much like a probe, that will show how well the model already captures information to perform this task.

3.5.3 Multi-task vs. Pre-training

To answer the third research question, we will compare multi-task training and pre-training on the auxiliary tasks. This comparison will be used to investigate to what extent MTL improves performance compared to pre-training. To acquire pre-training results, an adaptation to our setup will be implemented. In this setup, the auxiliary dataset will be trained before training on the Circa dataset.

4 Experimental Setup

This section describes our experimental setup in more detail. The full implementation of the research can be found in this GitHub repository.

4.1 Training

All models train on both the questions and answers present in the Circa datasets, as this yielded the best results, as proven by Louis et al. (2020). This study aims to improve upon these results, and therefore this setup will be regarded as a baseline. The weights of the epoch with the best development accuracies will be used to evaluate upon the test set. The used hyperparameters are summarized in Table 1. We employ the cross-entropy and mean-squared-error loss for classification and regression tasks respectively. We set the learning rate to `3e-05` for the main task when only training on the Circa dataset, and set the learning rate to `5e-05` when using an auxiliary task. For each task, a different optimizer instance is used. This allows us to train only the classification layers for the auxiliary tasks, for our second experiment.

Hyperparameter	Value
Batch size	`8`
Max. no. epochs	`5`
Patience	`3`
Optimizer	`AdamW`
Loss function	`CE / MSE`
Main task lr	`3e-05 / 5e-05`
Aux task lr	`2e-05`
Seed	`1234`

Table 1: Training hyperparameters.

4.2 Effect of Auxiliary Tasks

To test the effect of auxiliary tasks on the model performance, we compare multiple MTL models with the baseline BERT model fine-tuned only on the Circa dataset. For each model, we report the mean accuracy sampled over three runs.

4.3 Knowledge Encoded by Main Task

To test whether the main task already encodes knowledge on the auxiliary task, we again compare the different models with the baseline, and with the results from our previous experiments. This time, only the classification layers of the auxiliary tasks are trained. This way, BERT is not fine-tuned on the auxiliary tasks directly, but only the final classification layers of the auxiliary task learn to use the BERT embeddings efficiently. We again measure performance by averaging the accuracy scores over three consecutive runs. Specifically, we will measure the difference in mean accuracy between training the entire model and training only the classification layer.

4.4 Multi-task vs. Pre-training

To investigate the difference in performance between pre-training and MTL, we will pre-train on the auxiliary task, before training on the main task. Apart from the order in which the datasets are trained, the setup is very similar to the MTL approach, to allow us to compare the results. The

batch size, optimizer, patience, epochs, and seed as presented in Table 1 are used. However, as a learning rate 3e-05 is used for both the auxiliary and main task. This is done since this yielded the highest results for most datasets in the setup of Louis et al. (2020). As is the case for the MTL setup, all models are trained on both questions and answers, and the same auxiliary datasets will be used. The mean accuracy sampled over three runs, and the F1 scores will be compared.

5 Results & Discussion

This section discusses the results of the different experiments. Each subsection addresses the results of one of the research questions.

5.1 Effect of Auxiliary Tasks

For our first experiment, we compare the different MTL models with the baseline across the different settings. First, we analyze the results on the *matched* setting with *strict* labels, which are summarized in Table 2. The results show that the model that is multi-task trained with MNLI performs best, beating the single-task baseline with an absolute increase of 1.52% on the mean test accuracy. This seems to indicate that MNLI is a task that is helpful for the indirect answer classification task. Interestingly, all other MTL models perform worse than the baseline model. This seems to indicate that these tasks are not helpful for the indirect answer classification task, but are rather harmful.

Based on the results for the *matched* setting with *relaxed* labels, we observe a similar trend. These results are summarized in Table 3. Again, the MTL MNLI model outperforms the baseline with an absolute increase of almost 2% on the test accuracy. The other MTL models also underperform on this setting, compared to the baseline.

Table 4 shows the results of the unmatched setting. MNLI yields the highest mean, max, and min accuracy. This is in line with the results of the matched setup. For all models, there is a big difference between the maximum and minimum accuracy reached. This indicates that not all scenarios are classified equally well.

Table 12 in Appendix B provides insight into which scenarios are classified better than others. All models have the most difficulty in classifying scenarios 8 and 9 when unseen in training. A possible explanation for this could be that these scenarios differ in setting, compared to the other scenarios. The first 7 scenarios are social situations in which the annotators supposedly know each other well (according to the scenario they are friends, or colleagues). In scenario 8 and 9, the annotators supposedly do not know each other, or have not seen each other for a long time. One could easily imagine this could result in indirect answers that are more difficult to classify.

Besides differences in achieved accuracy between scenarios, we also observe differences in how well a model classifies specific labels. In Tables 2 and 3 the F1 scores are presented for each of the labels. The labels *Yes*, *No*, and *Yes, subject to some conditions* are more accurately classified than the other labels. It is no surprise that *Yes* and *No* yield the highest F1 score: these labels are much better represented in the Circa dataset. For instance, 42.3% of the labels is *Yes*, while only 1.9% is *In the middle, neither yes nor no*.

It is surprising to see however, that *Yes, subject to some conditions* is classified so easily since it takes up only 7.5% of the dataset. A possible explanation could be that this class entails an answer of a more complex nature than the other target groups. A conditional answer often refers to something outside the scope of the conversation. Perhaps this knowledge is captured by the model, or simply the length and complexity of the answer are used for classification.

For five of the six labels, multi-task training on MNLI results in the highest F1 score. Only for the *In the middle, neither yes nor no* label, the MTL MNLI model is outperformed by MTL on SST2. Training on SST2 could be useful for this class because a *In the middle, neither yes nor no* answer contains a clear absence of sentiment, which translates to a very low sentiment score in SST2.

Figure 2 shows the confusion matrix of the MTL model with MNLI as the auxiliary task. It is representative of how all presented models classify target groups. For a complete overview of all confusion matrices of the matched setting, we refer to Appendix C. The *Probably no* target group is often classified as *No*. Similarly, *Probably yes* is often classified as *Yes*. This result is as expected, since the distinction between these classes is not very clear, even for human annotators. Likewise, it is not surprising that *In the middle, neither yes nor no* is often classified as any of the other target classes.

	Accuracy		F1 score					
	Dev	Test						
	Mean	Mean	Yes	P.yes	C.yes	No	P.No	Mid
BERT-YN	83.11	82.19	87.1	50.0	87.6	84.3	22.1	48.8
BERT-IQAP-YN	80.61	80.97	86.4	44.4	89.7	81.8	14.2	47.6
BERT-BOOLQ-YN	80.70	80.92	84.8	46.1	88.3	80.9	14.8	36.8
BERT-MNLI-YN	**84.03**	**83.71**	**89.0**	**54.4**	**91.1**	**84.8**	**25.8**	48.0
BERT-SST2-YN	81.38	80.66	86.2	40.2	88.9	82.3	19.9	**54.4**
BERT-TOPICS-YN	83.58	80.16	85.8	46.0	88.3	81.4	16.7	42.2

Table 2: Accuracy and F1 scores of the different models on the *matched* setting with *strict* labels. The accuracy scores are over three runs. The F1 scores are from a single run.

	Accuracy		F1 score			
	Dev	Test				
	Mean	Mean	Yes	C.yes	No	Mid
BERT-YN	87.57	87.53	89.1	85.4	87.4	**52.8**
BERT-IQAP-YN	86.20	85.76	88.6	90.1	84.9	43.7
BERT-BOOLQ-YN	86.41	86.32	87.4	90.0	84.5	46.2
BERT-MNLI-YN	**89.97**	**89.49**	**92.2**	**90.5**	**90.5**	41.4
BERT-SST2-YN	86.77	86.74	88.4	88.3	86.0	47.2
BERT-TOPICS-YN	86.28	86.43	88.8	88.9	86.1	41.2

Table 3: Accuracy and F1 scores of the different models on the *matched* setting with *relaxed* labels. The accuracy scores are over three runs. The F1 scores are from a single run.

	Accuracy		
	Mean	Max	Min
BERT-YN	79.64	84.86	72.45
BERT-IQAP-YN	78.34	83.33	69.93
BERT-BOOLQ-YN	77.90	83.22	69.64
BERT-MNLI-YN	**81.98**	**85.79**	**75.13**
BERT-SST2-YN	78.60	83.19	70.63
BERT-TOPICS-YN	77.73	84.45	69.78

Table 4: Test accuracy of the different models on the *unmatched* setting. *Strict* labels are used.

5.2 Knowledge Encoded by Main Task

For our second experiment, we compare performance of only training the final classification for the auxiliary task, to training the entire model. These results are summarized in Table 5.

When training only the classification layers, the model outperforms random chance across all auxiliary tasks. The only exception to this is the IQAP task: it performs worse than random chance and its accuracy highly fluctuates across different runs. We hypothesize that this phenomenon can be attributed to the characteristics of the dataset. The relatively small size of the dataset might result in poor performance. For the other auxiliary tasks, performance shows that the main task already encodes some information that helps with these auxiliary tasks.

When training the entire model, the model achieves significantly better performance on all tasks. Notably, the BOOLQ task shows only a relatively small increase in performance (9.98%). This seems to indicate that training on the main task already encodes a lot of information on binary question answering. On the other hand, the MNLI task shows the biggest increase in performance (32.92%). This suggests that training on the main task does not encode a lot of information on the textual entailment prediction task. Interestingly, when only training the classification layer, the model performs better on the topics classification task. This seems to suggest that the BERT model already encodes such information in the main task. The fact that the topic labels are not 'hand-crafted' annotations, but rather stemming from linguistic relationships, further confirms this assumption: it has been proven by many (*e.g.* Tenney et al. (2019a); Tenney et al. (2019b)) that BERT encodes various linguistic concepts, without such knowledge being explicitly 'fed'. Moreover, Ravichander et al. (2020) have demonstrated that BERT indeed latently encodes the linguistic concept we consider in this auxiliary task, namely hypernymy relationships.

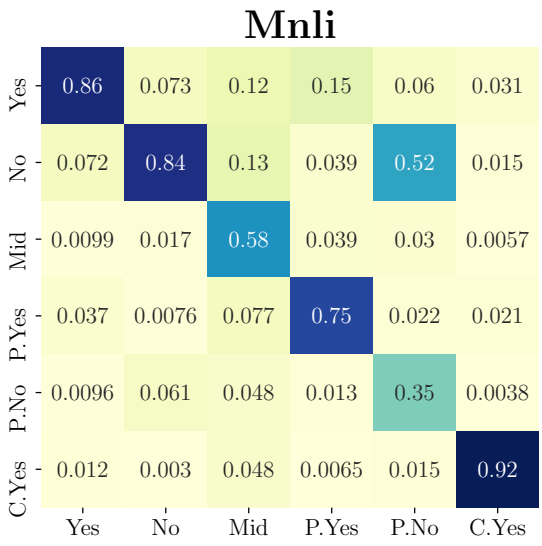

Figure 2: Confusion matrix for the MTL model with MNLI as auxiliary task.

Comparing the difference in performance for training only the classification layer or the entire model, with the results on the main task (Table 2 and 3), we see no clear relation. A big difference in auxiliary task performance does not relate to better performance on the main task. Unfortunately, we can therefore not judge the usefulness of an auxiliary task by the extent to which the main task already encodes information on this auxiliary task.

If the usefulness of a task can not be judged in this manner, this raises the question why some auxiliary tasks aid the performance of the model, while others harm it. Intuitively, as some auxiliary tasks are less relevant for the main task, optimizing for irrelevant tasks might steer the gradients into a sub-optimal direction.

	Mean	
Model	Accuracy	Δ_{acc}
BERT-IQAP-YN	42.23 / 20.73	+21.50
BERT-BOOLQ-YN	72.59 / 62.61	+9.98
BERT-MNLI-YN	81.96 / 49.04	+32.92
BERT-SST2-YN	82.67 / 65.93	+16.74
BERT-TOPICS-YN	83.08 / 86.36	-3.28

Table 5: Test performance (full learning / classification layer only) of the different models on their auxiliary tasks. Means are calculated over 3 consecutive runs.

5.3 Multi-task vs. Pre-training

In the pre-training setup, MNLI also achieves the best mean accuracy (see Table 6). Pre-training on MNLI yields slightly lower results, when compared to MTL. The difference is only 0.5% mean test accuracy (compare Tables 9 and 6).

This difference in mean accuracy between BERT-MNLI-YN in the pre-trained and multi-task setup can also be seen in the F1 scores. In the multi-task setup, MNLI dominates the other auxiliary tasks, resulting in the highest F1 scores for 5 out of 6 classes. However, in the pre-trained setup, this is not the case. The model misclassifies the categories *Probably yes*, *Probably no* and *In the middle, neither yes nor no* more often. These classes are also the most underrepresented classes in the Circa dataset. This indicates the pre-trained model has more trouble classifying underrepresented target groups, whereas the MTL model generalizes better to these underrepresented classes.

Since both the pre-training and MTL models use the exact same datasets, the question arises what causes the differences in results between the two approaches. In a pre-training setup, the auxiliary task is only seen before training on the main task, whereas in MTL, the model remains continuously trained on this auxiliary task. This could result in knowledge from the auxiliary task being 'overwritten'. In MTL, knowledge encoded by the auxiliary task stays relevant. This could cause the model to reach an optimum for both tasks jointly, instead of overfitting on the main task.

6 Conclusion

This section elaborates on the conclusions of the different research questions in their respective subsections.

6.1 Effect of Auxiliary Tasks

We found that the model that is multi-task trained with MNLI is the best performing model presented in this paper. The model outperforms the baseline and other models across the different settings. The model boasts an impressive absolute improvement of 1.52%, 1.96%, and 2.34% on the mean test accuracy across the *matched strict*, *matched relaxed*, and *unmatched strict* settings. However, the other multi-task models saw a decrease in performance across all of the settings. As such, an answer can be provided to the question of what auxiliary tasks encode additional knowledge that

	Accuracy		F1 score					
	Dev	Test						
	Mean	Mean	Yes	P.yes	C.yes	No	P.No	Mid
BERT-YN	83.11	82.19	87.1	**50.0**	87.6	84.3	22.1	**48.8**
BERT-IQAP-YN	82.14	81.97	87.1	44.3	**90.2**	83.8	15.4	39.8
BERT-BOOLQ-YN	81.55	81.66	87.0	47.2	88.4	83.3	16.4	40.8
BERT-MNLI-YN	**83.13**	**83.24**	**88.3**	44.0	90.0	**84.6**	18.2	40.9
BERT-SST2-YN	80.63	80.26	85.6	45.2	88.4	81.6	**25.9**	42.5
BERT-TOPICS-YN	79.06	79.45	84.9	46.1	88.6	79.3	**28.2**	43.8

Table 6: Accuracy and F1 scores of the different models after *pre-training* on the auxiliary task, before training on the main task (Circa). The accuracy scores are over three runs. The F1 scores are from a single run.

aids the classification of indirect answers. The textual entailment task provides useful information for the indirect answer classification task, while all other investigated auxiliary tasks (sentiment prediction, indirect answer classification, polar answer classification, and topic classification) harm the performance on the main task.

6.2 Knowledge Encoded by Main Task

The results show that most auxiliary tasks benefit from training the entire model, as oppose to only training the final classification layer. This seems to indicate that the information required for these tasks is not yet fully encoded by the main task. However, the model performs better than random chance on these auxiliary tasks when only training the classification layer. This indicates that the main task already encodes some useful information for the auxiliary tasks, but not enough to perform well. The one exception is the topic classification task, here performance is better when only training the classification layer. This seems to indicate that the BERT model has already captured all required information when trained on the main task. When answering the question to what extent the knowledge required for the auxiliary task is already captured by the main task, we can conclude that the model already captures some information on most of the auxiliary tasks but not enough to achieve good performance. The only exception is the topic classification task, on which the model already achieves good performance when trained on the main task.

6.3 Multi-task vs. Pre-training

The results show that both the MTL and pre-trained models improve accuracy compared to the baseline. However, the MTL models slightly outperform the pre-trained ones. More notably, on the one hand, the MTL MNLI model achieves the best F1 scores for all classes when compared to all other MTL models and the baseline. On the other hand, the pre-trained MNLI model only achieves the best F1 scores for two out of the six classes. The pre-trained MNLI model achieves the most gains on the most represented classes in the dataset: *Yes* and *No*. However, the gains on the other classes are relatively small. In comparison, the MTL MNLI model improves performance more considerably on all classes. This could indicate that MTL contributes to generalizability to underrepresented classes.

With this, an answer can be provided to the third research question, which is concerned with the extent to which Multi-Task Learning improves performance compared to pre-training. There seems to be a small improvement in accuracy when MTL is used instead of pre-training. There is an indication that is due to the fact that MTL leads to better generalizability. Further research is necessary in order to draw decisive conclusions.

7 Future Work

Future research could look at more auxiliary tasks that could be helpful for the indirect answer classification task. An example of an interesting auxiliary task could be Named-Entity Recognition.

Another direction of further research could be a more extensive comparison of multi-task models against pre-trained models in a range of domains. This could provide a stronger conclusion on the advantages of Multi-Task Learning over pre-training.

Research could also focus on new meta-learning techniques like Proto-MAML (Triantafillou et al., 2019) or LEOPARD (Bansal et al., 2020) to achieve models that generalize better over multiple datasets.

References

Trapit Bansal, Rishikesh Jha, and Andrew Mccallum. 2020. Learning to few-shot learn across diverse natural language classification tasks. pages 5108–5123.

Rich Caruana. 1997. Multitask learning. *Machine Learning*, 28.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. 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, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496, Florence, Italy. Association for Computational Linguistics.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. In *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics*.

Annie Louis, Dan Roth, and Filip Radlinski. 2020. "I'd rather just go to bed": Understanding indirect answers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7411–7425, Online. Association for Computational Linguistics.

Marie-Catherine de Marneffe, Scott Grimm, and Christopher Potts. 2009. Not a simple yes or no: Uncertainty in indirect answers. In *Proceedings of the SIGDIAL 2009 Conference*, pages 136–143, London, UK. Association for Computational Linguistics.

Vilém Mathesius. 2013. *A Functional Analysis of Present Day English on a General Linguistic Basis*. De Gruyter Mouton.

Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer.

Abhilasha Ravichander, Eduard Hovy, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2020. On the systematicity of probing contextualized word representations: The case of hypernymy in BERT. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 88–102, Barcelona, Spain (Online). Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew 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*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, page 4593–4601. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. 2019. Meta-dataset: A dataset of datasets for learning to learn from few examples. *CoRR*, abs/1903.03096.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

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)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Appendices

A Topics Auxiliary Task

In addition to the auxiliary tasks introduced in section 3, we introduce a self-designed task that is relatively straightforward in terms of linguistics (so that we can test the hypothesis whether the BERT model already latently encodes the 'knowledge' needed to solve our task), while still encapsulating additional semantic meaning useful for the main task. As briefly explained in section 3.6, our `TOPICS`-task aims to extract the topic of an indirect answer and is based on two simple steps: (1) extracting the 'most important noun' of the indirect answer; and (2) classifying that noun into a pre-defined topical category. In this appendix, we will outline the involved methodology and discuss the additional findings resulting from it.

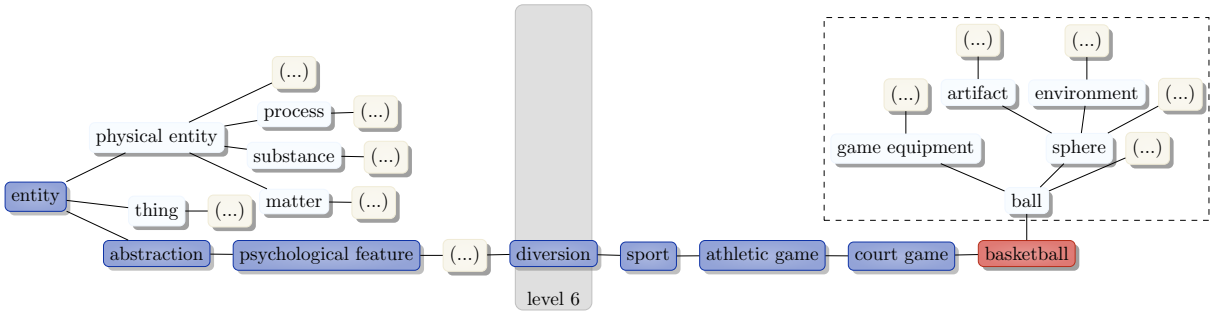

Figure 3: Truncated WordNet synset-tree for the lemma 'basketball.' The dashed rectangle represents a second meaning of 'basketball': the *ball* instead of the *game*. In this subtree, more polysemy is encountered: *sphere* amounts to seven different synsets. In the bottom-up traversal pool approach, all encountered lemmas are collected in a 'pool,' which is compressed to contain only frequently occurring lemmas, from which topic labels are selected. In the naive approach, the blue lemma within the gray box is chosen as the topic label.

A.1 Methodology

Important Word Annotations We distinct between two different methods: (1) naively extracting the *last noun* of a sentence, based on the idea that 'new information' is usually put at the end of a sentence (Mathesius, 2013, p. 83, 157); (2) extracting the noun with the highest `TF-IDF`-value (Ramos et al., 2003), which roughly comes down to the noun that occurs the *least frequent* across the Circa dataset, based on the intuition that this noun naturally conveys the highest degree of *added* information. In both cases, we tag parts of speech (to determine which tokens are nouns) using NLTK (Loper and Bird, 2002).

Topic Label Annotations For translating important words into topic labels, we adopt a method based on the insight that nouns involved in a hypernym relations semantically embody some sort of 'umbrella concept,' which is intuitively related to the notion of a 'topic.' An auxiliary task can then be formulated as a trivial classification problem. Evidently, as Circa contains a large number of samples, the possible amount of topic labels is also large (depending on the annotation method). To make this feasible for a deep learning model, we need to 'control' the number of classification labels, while at the same time preserving their degree of 'topical expressiveness.' We have experimented with two different strategies based on the structure of the WordNet Synset tree (Fellbaum, 1998). In figure 3, we have visualized a part of such tree for a single sample obtained by the TF-IDF method, as explained in the previous step.

- In the **naive case**, we a-priori decide on a tree level (highlighted with a gray box in the referred figure) from which the topic label is chosen based on the 'left-most' path of the tree (visualized by the blue-coloured lemmas). The 'lower' the box (tree level) is chosen, assuming a top-down approach, the larger the number of resulting distinctive topic labels. Note that, as the hypernymy relations can be structured differently for different lemmas, the highlighted 'box of interest' can contain labels of a varied degree of abstraction, which is a significant limitation of this naive method.

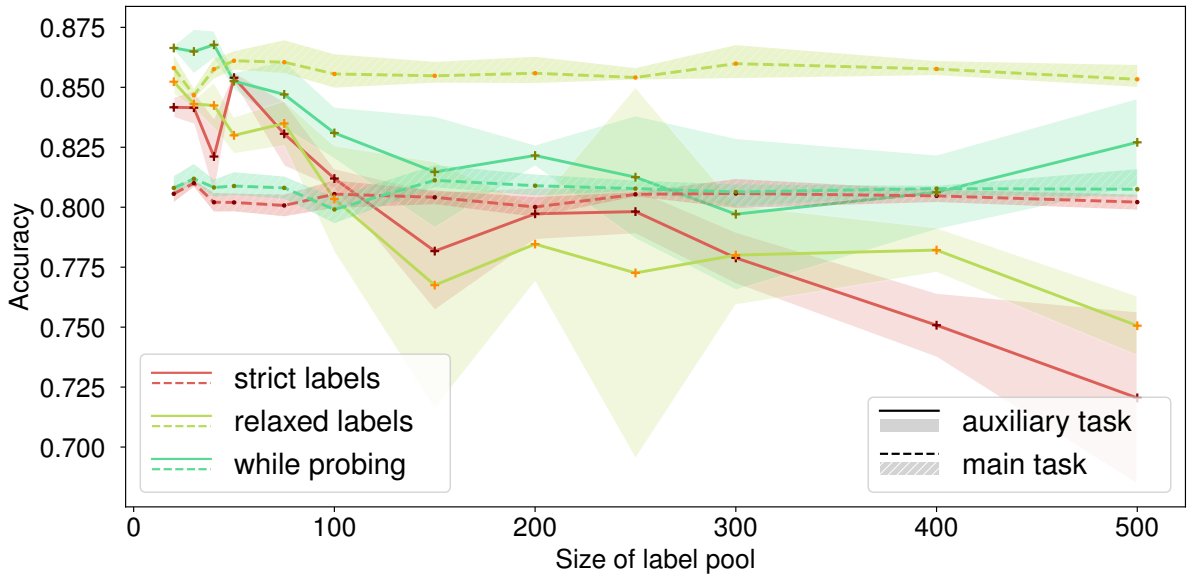

Figure 4: Experimental results for the main task (*matched* setting) and the TOPICS-auxiliary task.

- In the more robust, **bottom-up traversal pool** approach, we traverse the entire tree in a bottom-up fashion, starting from the node that represents the lemma of the extracted word. For every such word in the dataset, accounting for every polysemic unit, we recursively obtain a set of different lemmas with which the root lemma, the 'most important word,' is indirectly connected. An example subtree with additionally such considered lemmas is (dashed) outlined in the figure 3. The occurrence frequency of every lemma, within all sets of lemmas for every sample in de dataset, is counted so that a 'pool' is constructed with the top-m frequent lemmas among all data samples. Afterwards, for every sample, the most *specific* lemma (that is, the closest node to the root lemma) is selected from the pool. The value of m thus denotes the density of the *allowed labels* for every sample. This method preserves a maximum degree of non-abstraction, while still controlling the number of resulting labels. Note that this method of tree traversal is *unidirectional* – trees are traversed strictly bottom-up, without further exploring additional branches in downwards direction.

For the purpose of providing further insight in the workings of the listed algorithms, we have hand-picked a few example answers (and its annotations) and outlined these in table 8.

	Run			
Task	First	Second	Third	Mean best
Relaxed labels (main)	300 / 88	30 / 19	30 / 19	30 / 19
Strict labels (main)	150 / 88	300 / 182	75 / 39	50 / 25
Probing (main)	500 / 280	30 / 19	150 / 88	30 / 19
Strict labels (aux)	50 / 25	50 / 25	50 / 25	20 / 12
Relaxed labels (aux)	20 / 12	20 / 12	20 / 12	20 / 12
Probing (aux)	40 / 22	30 / 19	20 / 12	40 / 22

Table 7: Best scoring 'pool sizes' for all experimental tasks. Format: `pool size / number of resulting topic labels`.

A.2 Additional Results and Discussion

At first glance, the labeled 'topics' present in table 8 look far from 'ground truth.' These topics should, however, be accurate *in the light of* the indirect answer classification task that we attempt to improve. In this context, the label 'living thing' is *spot-on* for the indirect answer 'I have two cats at home.'

As we are interested in the *general applicability* of an auxiliary task of this kind in a multi-task learning context like considered in this paper, we are only concerned with the *matched* setting using both strict and relaxed labels (see §3.2) combined with a probing task (§3.4.2). As is evident from table 8, topic labels vary in degree of abstraction based on the size of the 'pool of allowed labels.' Therefore, we study 12 different levels of label abstraction and run experiments with pool sizes (m) of $20, 30, 40, 50, 75, 100, 150, 200, 250, 300, 400$ and 500. All experiments were run three times, costing about 180 GPU hours on the Lisa cluster. Mean accuracy scores and corresponding standard deviations are reported in figure 4. Best performing pool sizes are further outlined in table 7.

The *overall* best performing pool size is **30**, which we use to compute the advanced metrics reported in the main part of this paper. While we observe a clear trend in accuracy scores with relation to the number of involved topic labels (the 'difficulty' of the classification task), the best scoring pool density, in terms of performance on our main task, appears to be random. This explains the relative stability in performance of the main task given different difficulty levels for multi-task training on our auxiliary task – the BERT model seems to already embed tokens of the indirect answers on such a way, that its topic annotations can eventually be derived. The similarly high accuracy on the probing task further confirms this assertion. The fact that the topic annotations directly originate from linguistic concepts makes this assumption even more plausible: BERT is know to 'embed' linguistic knowledge (Tenney et al. (2019a); Tenney et al. (2019b)), including hypernyms (Ravichander et al., 2020). The fact that the probing task actually scores *higher* when not in a multi-task learning environment, might indicate that simultaneously fine-tuning on other tasks actually *hinders* the learning of some tasks; in particular, the results seem to indicate that multi-task learning on the Circa dataset seem to slightly 'corrupt' the latent topical embeddings in the BERT model. In general, the topics task itself performs remarkably well.

An obvious limitation is that only answers containing nouns can be classified. This causes a significant chunk of the dataset to remain unused and a certain class of answers (those that do not contain a noun) to be neglected during training process. From 10,027 samples (out of 34,269) a noun could not be extracted. Furthermore, not all nouns have a WordNet counterpart. Finally, the task trained on 20,107 samples.

Due to a lack of time, we could not provide an extensive analysis and comparison of all methods outlined in the previous section. For this reason, we empirically set the topic depth (in the naive topic annotation task) to **6**, even though we decided, in likewise manner, to not employ the mentioned naive tasks. As the applicability of similar auxiliary tasks might be useful to a broader range of fields in dialogue modeling, research into which methods are of preference in different linguistic contexts might prove fruitful.

Indirect answer	**Important words**		**Topic annotations**			
	last	TF-IDF	naive	Bottom-up traversal pool		
				$m = 20$	$m = 300$	$m = 500$
Basketball is a lot of fun	fun	basketball	diversion	human activity	sport	athletics
I'm a veterinary technician	technician	technician	person	whole	person	worker
I'll be having dinner at home	home	dinner	meal	physical entity	nutrition	nutrition
My cousin's wedding is Saturday.	Saturday	wedding	ceremony	human activity	function	affair
I managed a restaurant in college.	college	restaurant	building	artifact	edifice	edifice

Table 8: Hand-picked example 'important word' and subsequent 'topic' annotations for five indirect answers.

B Matched Results and Unmatched Results per Scenario

Model	Run			
	First	Second	Third	Mean
BERT-YN	83.06 / 82.24	83.16 / 82.14	83.11 / 82.19	83.11 / 82.19
BERT-IQAP-YN	79.94 / 80.42	80.37 / 81.05	81.51 / 81.45	80.61 / 80.97
BERT-BOOLQ-YN	81.09 / 81.18	80.56 / 80.90	80.45 / 80.67	80.70 / 80.92
BERT-MNLI-YN	84.54 / 83.70	83.64 / 83.60	83.91 / 83.83	**84.03 / 83.71**
BERT-SST2-YN	81.40 / 81.24	81.61 / 80.86	81.14 / 79.87	81.38 / 80.66
BERT-TOPICS-YN	83.79 / 80.61	84.87 / 79.12	82.09 / 80.74	83.58 / 80.16

Table 9: Accuracy (dev / test) of the different models on the *matched* setting with *strict* labels.

Model	Run			
	First	Second	Third	Mean
BERT-YN	87.97 / 87.13	87.06 / 88.03	87.68 / 87.42	87.57 / 87.53
BERT-IQAP-YN	86.16 / 85.13	85.22 / 85.25	87.21 / 86.90	86.20 / 85.76
BERT-BOOLQ-YN	85.74 / 86.07	86.83 / 86.65	85.67 / 86.24	86.41 / 86.32
BERT-MNLI-YN	89.78 / 90.11	90.44 / 89.19	89.69 / 89.18	**89.97 / 89.49**
BERT-SST2-YN	86.56 / 86.20	86.57 / 86.52	87.17 / 87.50	86.77 / 86.74
BERT-TOPICS-YN	85.36 / 85.96	87.14 / 86.76	86.33 / 86.57	86.28 / 86.43

Table 10: Accuracy (dev / test) of the different models on the *matched* setting with *relaxed* labels.

Model	Run			
	First	Second	Third	Mean
BERT-YN	83.06 / 82.24	83.16 / 82.14	83.11 / 82.19	83.11 / 82.19
BERT-IQAP-YN	81.80 / 81.80	82.48 / 81.88	82.15 / 82.24	82.14 / 81.97
BERT-BOOLQ-YN	81.65 / 81.54	81.57 / 81.67	81.43 / 81.78	81.55 / 81.66
BERT-MNLI-YN	83.33 / 83.11	83.12 / 83.27	82.95 / 83.33	**83.13 / 83.24**
BERT-SST2-YN	80.80 / 80.47	80.91 / 79.80	80.88 / 80.52	80.63 / 80.26
BERT-TOPICS-YN	79.05 / 79.27	79.63 / 80.33	78.54 / 78.76	79.06 / 79.45

Table 11: Accuracy (dev / test) of the different models after *pretraning*.

	YN	IQAP-YN	BOOLQ-YN	MNLI-YN	SST2-YN	TOPICS-YN
0	78.84	79.51	79.13	**83.93**	78.74	77.27
1	84.86	83.33	83.22	**85.79**	83.19	84.45
2	80.44	80.49	78.38	**82.04**	80.69	79.49
3	80.59	79.48	78.89	**82.98**	79.19	78.46
4	78.42	76.08	75.39	**81.61**	76.68	75.37
5	**84.61**	82.21	84.10	83.07	82.45	82.45
6	80.83	79.31	78.99	**84.25**	80.82	79.41
7	82.38	80.39	81.42	**85.17**	82.39	80.57
8	73.01	72.65	70.84	**75.13**	71.17	70.05
9	72.45	69.93	69.64	**75.87**	70.63	69.78

Table 12: Test accuracy of the different models on the *unmatched* setting. *Strict* labels are used.

C Confusion Matrices (Matched setting with strict labels)

Figure 5: Confusion matrices for the matched setting.