MaChAmp at SemEval-2022 Tasks 2, 3, 4, 6, 10, 11, and 12: Multi-task Multi-lingual Learning for a Pre-selected Set of Semantic Datasets

Rob van der Goot
IT University of Copenhagen
`robv@itu.dk`

Abstract

Previous work on multi-task learning in Natural Language Processing (NLP) often incorporated carefully selected tasks as well as carefully tuning of architectures to share information across tasks. Recently, it has shown that for autoregressive language models, a multi-task second pre-training step on a wide variety of NLP tasks leads to a set of parameters that more easily adapt for other NLP tasks. In this paper, we examine whether a similar setup can be used in autoencoder language models using a restricted set of semantically oriented NLP tasks, namely all SemEval 2022 tasks that are annotated at the word, sentence or paragraph level. We first evaluate a multi-task model trained on all SemEval 2022 tasks that contain annotation on the word, sentence or paragraph level (7 tasks, 11 sub-tasks), and then evaluate whether re-finetuning the resulting model for each task specifically leads to further improvements. Our results show that our mono-task baseline, our multi-task model and our re-finetuned multi-task model each outperform the other models for a subset of the tasks. Overall, huge gains can be observed by doing multi-task learning: for three tasks we observe an error reduction of more than 40%.[1]

1 Introduction

Recently, language models have become the de-facto standard in Natural Language Processing (NLP), where we first train a set of parameters on raw data, which are then finetuned on the task at hand. This in itself is a multi-task setup (language-modeling + target task). However, traditionally, multi-task learning was mainly done between multiple NLP tasks with gold annotation. In this setup, many questions arise: not only how to share the information between different tasks, but also when to share and even which tasks to use, as it is non-trivial

to decide which auxiliary tasks are beneficial for a certain target task (Ruder, 2017; Crawshaw, 2020). Early work on multi-task learning in NLP often used up to a handful of tasks, carefully curated dataset/task combinations, and carefully tuned how to share the information between these tasks (e.g. Hashimoto et al. (2017); Søgaard and Goldberg (2016)).

A line of recent work has shown that an intermediate step can be used to finetune the language model on a set of NLP tasks, which leads to a model that is more apt for learning other NLP tasks. This is also called Supplementary Training on Intermediate Labeled-data Tasks (STILT), and was introduced by Phang et al. (2018). Phang et al. (2018) train on three classification tasks from the GLUE benchmark (Wang et al., 2018), and then retrain for all GLUE tasks, showing a 1.4 point of improvement over all GLUE tasks. Phang et al. (2020) shows that this positive transfer also holds cross-lingually when using multilingual language models and only doing intermediate training on English tasks. Wang et al. (2019). Similar as with earlier models, it remains an open question for STILT models which tasks transfer well to which tasks (Vu et al., 2020; Pruksachatkun et al., 2020; Chang and Lu, 2021). It should be noted that most work on STILS for autoencoder language models is done on text (i.e. sentence) classification only.

Later work used autoregressive language models, which learn to generate texts (as opposed to the autoencoding models, which learn to predict one token at a time, used by the previously mentioned STILT papers). These language models are commonly used for different types of tasks, namely generation tasks (e.g. question answering, machine translation, summarization), whereas autoencoding models are commonly used for classification and word-level tasks (text classification, pos-tagging, parsing etc.). Recent work has shown that many NLP tasks can be converted to generation tasks, and

---

[1] code available at: `https://bitbucket.org/robvanderg/semeval2022`

SemEval Task	Included sub-tasks	Languages	Citation
2: Multilingual Idiomatic-ity Detection	Idiomaticity detection (1-shot)	EN, PT, GL	Tayyar Madabushi et al. (2022, 2021)
3: PreTENS	1: Binary acceptability	EN, IT, FR	Zamparelli et al. (2022)
	2: Regression acceptability	EN, IT, FR	
4: Patronizing and Condescending Language Detection	1: Binary PCL detection	EN	Pérez-Almendros et al. (2022); Perez Almendros et al. (2020)
	2: Multi-label PCL classification	EN	
6: iSarcasmEval	1: Sarcasm detection	EN, AR	Abu Farha et al. (2022)
	2: Irony-labeling	EN	
	3: Paraphrase sarcasm detection	EN, AR	
10: Structured Sentiment Analysis	Expressions, entities and relations	CA, EN, ES, EU, NO	Barnes et al. (2022)
11: MultiCoNER - Multilingual Complex Named Entity Recognition	Named Entity Recognition	BN, DE, EN, ES, FA, HI, KO, MI, NL, RU, TR, ZH	Malmasi et al. (2022)
12: Symlink	Entities and relations	EN	Dac Lai et al. (2022)

Table 1: Overview of all tasks we participate in. Original source of the data of task 10 are Øvrelid et al. (2020); Barnes et al. (2018); Agerri et al. (2013); Wiebe et al. (2005); Toprak et al. (2010).

can then directly be used to (re-)train an autoregressive language model in a multi-task setup (Aribandi et al., 2022; Sanh et al., 2022). In this setup it is easier to exploit a variety of task-types and a much higher amount of datasets (~50-100 datasets) is used compared to previous work.

In this paper, we set out to examine whether we can obtain performance improvements with multi-task learning and re-finetuning after multi-task learning (i.e. STILT) for a pre-defined set of semantically focused NLP tasks. More precisely, we will use the pre-defined set of SemEval 2022 tasks, and train a multi-task model for all text-based SemEval tasks that include annotation on the word, sentence or paragraph level.[2] We compare a strong single task baseline to a default multi-task learning model, where the encoder is shared, each task has its own decoder, and training is done on all tasks simultaneously (shuffled batches). Finally, we use the parameters from the multi-task model to train a task-specific model for each task again. We seek to answer the following research question:

- Can we exploit a pre-selected combination of NLP tasks in a multi-task setup to improve the ability of an autoencoder language model to learn NLP tasks?

To the best of our knowledge, we are the first to participate in more than 2 SemEval tasks simultaneously, by participating in 7 tasks and a total of 11 tasks including sub-tasks. In our multi-task model, we model a total of 19 tasks if we train on the full data from the tasks (some tasks are modeled as multi-task by themselves), and 54 tasks if we separate them by language or dataset. We will release the finetuned multi-task language models on the hugginface hub (Wolf et al., 2020) for future use, which we dub: Sem-mmmBERT (SemEval MaChAmp multi-task multi-lingual BERT)[3] based on mBERT (Devlin et al., 2019) and Sem-RemmmBERT (SemEval Rebalanced MaChAmp multi-task multi-lingual BERT)[4] based on Rem-BERT (Chung et al., 2021).

2 Datasets

An overview of the datasets for the tasks included in our setup is shown in Table 1. For task 2, the regression task has no gold training data, so it was left out. Furthermore, we did not participate in any constrained tracks, as we are mainly interested in setups where we also trained on other data. The languages used in the tasks have some overlap, but also some unique languages. English is present in all tasks.

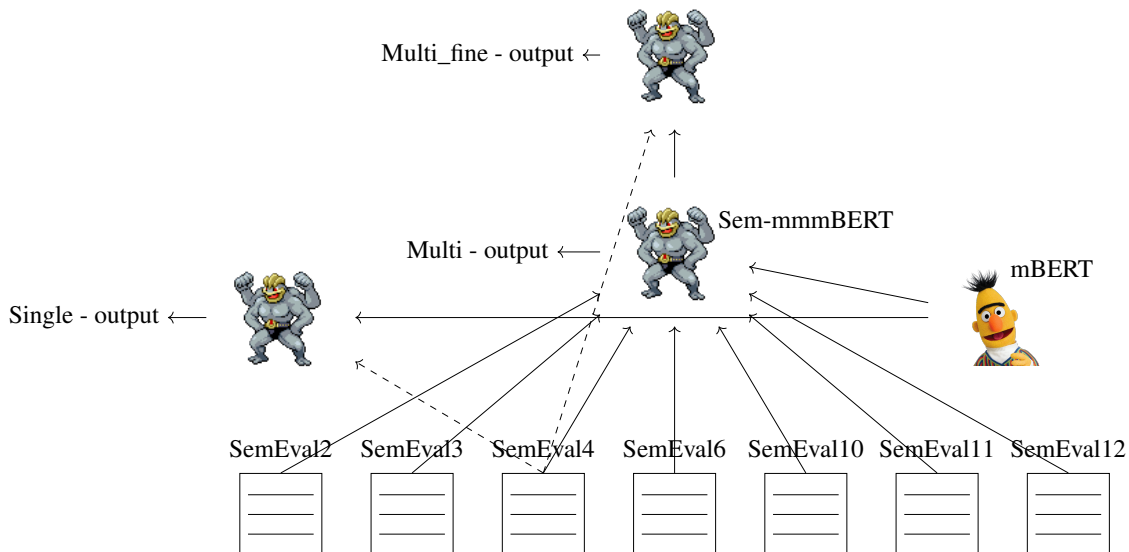
[2] document level annotation is excluded, as it is non-trivial to model in current autoencoder language models

[3] https://huggingface.co/robvanderg/Sem-mmmBERT
[4] https://huggingface.co/robvanderg/Sem-RemmmBERT

Figure 1: Schematical overview of our proposed multi-task models and the mono-dataset baseline. Sem-mmmBERT is the BERT model which can also be useful for other downstream tasks, and thus the one we release on the huggingface hub. In this example we show the usage of data when training models for task 4 (dashed arrows). For illustrational purposes we left out the sub-tasks in this figure. The boxes with the lines represent annotated data, and 👹 = a trained MaChAmp model.

Task	MaChAmp task-type	#words	#sents	#sents smoothed
2-a1	classification	10,199	139	2,742
3-1	classification	99,044	11,669	25,131
3-2	regression	4,761	785	6,518
4-1	classification	399,376	8,369	21,283
4-2	classification	135,750	2,202	10,917
6-a	classification	83,266	5,254	16,863
6-b	classification*6	12,183	691	6,115
6-c	classification	29,242	1,287	8,346
10	seq seq seq	1,109,260	58,799	56,413
11	seq_bio	2,768,898	171,300	96,288
12	seq seq	944,176	3,120	12,994

Table 2: The task-types used within machamp for each of the (sub-)tasks, and the data size before and after smoothing.

Table 2 reports the sizes of the datasets, we see that there is a large variety. We attempt to overcome this with dataset smoothing, which is described in more detail in Section 3.8.

3 Model

We implemented all of our models in MaChAmp v0.3beta (van der Goot et al., 2021). MaChAmp is a toolkit that focuses on multi-task learning for NLP task-types based on AllenNLP (Gardner et al., 2018) and the transformers library (Wolf et al., 2020). It supports a wide variety of tasks, and a variety of options for multi-task learning (for within as well as cross-dataset multi-task learning). A typical MaChAmp model consists of a shared encoder (i.e. language model), with multiple decoders on top (one for each task), which all share the same encoder. We use all default hyperparameters of MaChAmp for our experiments, except for the dataset smoothing (Section 3.8). Our general setup is shown in Figure 1. As baseline, we take the data of a single SemEval task, and finetune a MaChAmp model with all default hyperparameters (SINGLE). The first multi-task setup, is a MaChAmp model trained on a combination of all SemEval tasks we consider (MULTI), where each task has its own decoder. Finally we take the hyperparameters from the MULTI model, and refinetune them for a single task at a time (MULTI-FINE).

For the relation extraction tasks (task 10 and 12), we first converted the data to a word-level sequence labeling task, and we contributed a regression task-type in MaChAmp, to be able to tackle task 3-2. For all sub-tasks with multiple languages/datasets, we evaluate also whether learning these in separate decoders is useful (so we split the datasets, and learn them as separate tasks). Below, we describe the choices we made for each of the tasks (the MaChAmp task-types can be found in Table 2), after which we describe our two multi-task setups (Section 3.8 and Section 3.9).

Positive   Negative

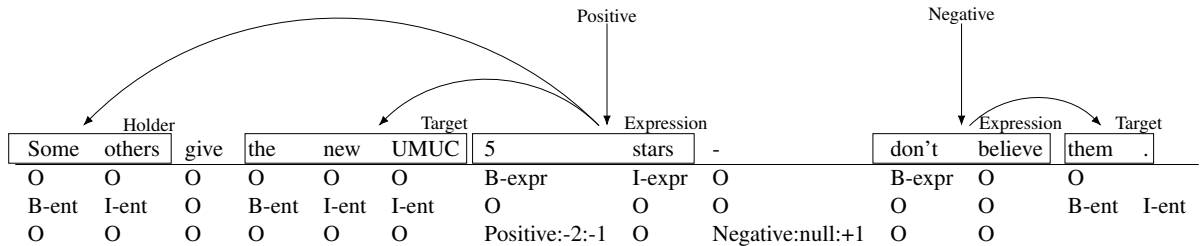| Some	others	give	the	new	UMUC	5	stars	-	don't	believe	them	.
Holder			Target			Expression			Expression		Target	
O	O	O	O	O	O	B-expr	I-expr	O	B-expr	O	O	
B-ent	I-ent	O	B-ent	I-ent	I-ent	O	O	O	O	O	B-ent	I-ent
O	O	O	O	O	O	Positive:-2:-1	O	Negative:null:+1	O	O		

Figure 2: Example of conversion of sentiment graph to sequence labeling for task 10, showing a gold annotated sentence (top of line) and the three layers of annotation that the model is supposed to predict (below the line): expression identification, entity identification and relations.

3.1 Task 2

We only participate in the supervised task (one-shot task a), which is a binary task where the goal is to identify whether a sentence contains an idiomatic expression. We use the classification task-type in MaChAmp, and include the multiword expression (MWE) as well as all 3 sentences (target + context) as input. Note that they will automatically be separated by a special separation token in MaChAmp, and their segment ID's will be all 0's for the MWE and third sentence, and 1's for the second and fourth sentence (for language models supporting segment ID's). We use macro-f1 for model picking as well as for the results we report in Section 4.

3.2 Task 3

Subtask 1 is a binary classification task: is a sentence (semantically) acceptable or not. We use the classification task type in MaChAmp and the macro-f1 metric. The data is divided in folds by the organizers, we use fold 1 and 2 as train data, and 3 as dev data. For subtask 1 we use macro-f1 for model picking, and report macro-f1s from the official evaluation script in Section 4.

Subtask 2 is a regression task, where we predict an acceptability score between 1 and 7. We contributed a regression decoder to MaChAmp, which uses a simple linear layer and mean square error loss. We use fold 0 for training and fold 1 as dev data. For subtask 2 we use pearson correlation for model picking.

3.3 Task 4

Subtask 1 concerns a binary classification task: does an utterance contain patronizing or condescending language or not. Subtask 2 identifies one out of 7 sub-categories of patronizing and condescending language. We model both tasks as clas-sification task in MaChAmp, and split the data for both sub-tasks in 80% train and 20% dev data. Following the official metrics, we use accuracy for task 1 and macro-f1 score for task 2 for model picking.

3.4 Task 6

We use an 80:20 split for each of the tasks. Task A is binary sarcasm detection, task B is a multiclass classification task, in which we model each category as a separate task, so that multiple classes can be predicted. Task C is paraphrase detection between sentence-pairs. We follow the official metrics and use macro-f1 for task A and B, and accuracy for C for model picking.

3.5 Task 10

Task 10 is fine-grained sentiment analysis, in which sentiment graphs are predicted. Each opinion is annotated as a tuple consisting of: an expression, which has a polarity (positive/negative) a link to the target, and potentially a link to a holder/source (the person expressing the sentiment). Inspired by Ramponi et al. (2020), we convert this task to three sequence labeling tasks (see also Figure 2). The first task is expression identifiction, which we model as BIO encoded spans. It should be noted that the spans can overlap. The second task is the identification of the source and targets, which are also encoded as BIO spans, which can also overlap. For each token that is the beginning of an expression (B-label), we include a label describing the relations (the third task), which are triples containing of: polarity, link to holder, link to target. The links to targets are simply counts of the directions to the next identified entities (i.e. +1 for the next identified entity), similar to the relative POS strategy of Strzyz et al. (2019), and the relation extraction implementation of Ramponi et al. (2020). There can be multiple relations for a given

expression. We concatenate the overlapping labels (which all three sub-tasks have) and model these three tasks in MaChAmp as sequence labeling task. We compared this against using the "multiseq" task type, which can output multiple labels per token. However, performance was better when simply doing sequence labeling, in contrast to Ramponi et al. (2020). Perhaps tuning the threshold of the prediction confidence to include labels could lead to better results (we used the default of 0.5), which we leave for future work. After prediction, we convert the data back to the official json format.

For model picking, we take the average over the accuracies of the concatenated labels, in Section 4 we report the official metric, sentiment graph F1 (Barnes et al., 2022).

3.6 Task 11

Task 11 is multi-lingual named entity recognition. We compared running with and without a CRF-layer, and found that the CRF layer is beneficial. We use span-f1; the implementation of AllenNLP for model picking, and the output of the `conlleval.pl` script for results reported in Section 4, because there is no official evaluation script available

3.7 Task 12

Task 12 is the linking of mathematical symbols, which consists of two steps: 1) detect mathematical symbols 2) identify links between them, which are directed and labeled. We use a similar strategy as we used in task 10, where we convert the task to sequence labeling. In contrast to task 10, the data in task 12 is not pre-tokenized, and some of the spans do not allign with the whitespaces. We tokenize with the `_is_punctuation` function from the transformers library (Wolf et al., 2020) to circumvent this, and save where it splits so that we can undo it after prediction. Similar as for task 10, a token can have multiple labels, we attempt to model this with the "multiseq" task-type in MaChAmp, which can predict multiple labels, but obtain better results by concatenating the labels and predict them as one label per token. We used accuracy for both tasks, as the official metric was not released. The results reported in Section 4 are the average of these two sub-tasks.

3.8 MULTI

We compare the single task baselines to models where we exploit multi-task learning (see also Fig-

Task	COMBINED	SEPARATE
task2-a1	**66.00**	61.28
task3-1	**66.77**	66.71
task3-2	**74.07**	72.14
task4-1	42.59	—
task4-2	25.67	—
task6-a	**31.27**	31.25
task6-b	17.05	—
task6-c	90.74	**91.67**
task10	**35.10**	28.90
task11	**79.86**	79.48
task12	96.06	—

Table 3: Scores (dev) of single-task models with mBERT. SEPARATE means that the data from each language (or dataset for task10) has its own decoder. An empty cell (—) means that the task did not consist of multiple datasets/languages, so SEPARATE equals COMBINED.

ure 1). In the first setup, we finetune MaChAmp on all tasks simultaneously, for which we enable the multinomial smoothing in MaChAmp with $\alpha = 0.5$, so that the distribution between tasks becomes more similar (see also Table 2. Note that some SemEval tasks consist of multiple sub-tasks, and some single tasks are modeled as multiple tasks in MaChAmp, we have a total of 19 tasks in the final setting. We evaluate the output of each decoder/task separately for this model.

3.9 MULTI_FINE

After the multi-task model is trained, we save the parameters of the shared encoder, so that they can be re-used for the next step. Finally, we re-finetune the resulting model for each task seperately again, to see whether the multi-task model constitutes a better initialization than the vanilla language model.

4 Results

For all the tasks where the shared task organizers released an evaluation script, we used the official script for the results reported in this section (for the model-picking we used internal equivalent metrics, see Section 3 for the details per task); for task11 we used `conlleval.pl`, and for task12 we used an average of accuracy over our converted data.

Task	SINGLE	MULTI	MULTI_FINE
task2-a1	**66.00**	64.67	63.64
task3-1	66.77	66.82	**66.87**
task3-2	74.07	84.82	**85.37**
task4-1	42.59	52.81	**80.00**
task4-2	25.67	**28.86**	27.65
task6-a	31.27	**59.75**	43.08
task6-b	17.05	**21.64**	19.22
task6-c	90.74	89.20	**95.37**
task10	35.10	**37.70**	25.70
task11	**79.86**	75.73	79.52
task12	**96.06**	95.10	95.31
avg.	56.83	61.55	**61.98**

Table 4: Scores of multi-task settings versus the single task baselines for mBERT.

	MULTI	MULTI_FINE
task2-a1	**78.79**	67.38
task3-1	66.85	**66.86**
task3-2	85.41	**85.80**
task4-1	65.57	**71.07**
task4-2	28.77	**29.54**
task6-a	**69.74**	51.66
task6-b	**29.82**	18.77
task6-c	**96.30**	91.05
task10	**47.70**	45.60
task11	80.45	**82.94**
task12	95.67	**96.33**
avg.	**67.73**	64.27

Table 5: Scores of multi-task settings for RemBERT.

4.1 Single task results

On the single task level, we compared for all datasets consisting of multiple languages or sub-datasets whether it is useful to train them as a single task (with one decoder: COMBINED), or as separate tasks (with N decoders: SEPARATE). For computational efficiency, these tests are only done with mBERT.

Table 3 shows that modeling the languages in separate decoders is only beneficial for task 6 c. We hypothesize that this is because this dataset only contains two languages (English and Arabic), which are relatively distant, so sharing the decoder leads to performance drops. For all further experiments, we will use the COMBINED setup.

4.2 Multi-task results

We first evaluate the results with mBERT, as we also have the single-task results with mBERT (due to computational constraints we do not have them for RemBERT). Table 4 shows the scores for the single-task (SINGLE) baseline, the multi-task model (MULTI), and the intermediate multi-task with finetuning per task setup (MULTI_FINE). Interestingly, each of the three models perform best for 3 or 4 different tasks, and it is thus highly dependent on the task which setup is most beneficial. Differences between scores can be huge though, and when looking at the averages it is clear that the multi-task setups are beneficial over single task models and competetive to each other. The smallest and largest datasets (Table 2) score best with the single task model, as well as task 12, which

can be considered an outlier. The multi-task setup without additional finetuning (MULTI) seems to be mostly beneficial for classification tasks. The additional finetuning (MULTI_SEQ) is especially flourishing for task 4-1 and 6c, which are small to medium sized classification tasks, and it is unclear why their trends differ so much compared to task 3-1, 4-2 and 6-a. It should be noted that MULTI is computationally more attractive as well as much smaller to store, as we only need one model for all tasks.

Before the deadline for the SemEval task, we managed to also train the final model with Rem-BERT (Chung et al., 2021) as language model, however, we do not have the single task baselines. Unfortunately, here only the model with separate decoders for each language/dataset fit on our largest GPU (40gb), so we submitted results with these.

Table 5 shows that also for the RemBERT embeddings, there is no clear single best strategy. The best multi-task strategy sometimes differs compared to the mBERT results (Table 4): task4-2, task11 and task12 differ, where the latter two where the tasks where the single task was the best performing for the mBERT embeddings. On average, the MULTI setup performs more than 3 absolute points higher, but this is mainly due to task 6, which has 3 subtasks (and thus weights heavier in the average).

4.3 Test data

We submitted the results of the mBERT single task baseline and the RemBERT MULTI_FINE setup for the official test evaluations. In Table 6 we show the obtained results and rankings for each task.

Task	Single mBERT	Multi_fine RemBERT	Ranking
task2-a1	—	66.07	NA
task3-1	78.78	86.42	11/21
task3-2	0.6792	-0.164	17/17 (3/17)
task4-1	0.4172	0.4211	56/78
task4-2	0.0772	0.1546	34/49
task6-a	0.3639	0.3187	31/43 & 12/32
task6-b	0.0919	0.0851	3/22
task6-c	0.2400	0.2250	16/16 & 13/13
task10	0.472	0.501	13/22
task11	0.6027	0.6768	18/26
task12	2.67	7.42	—

Table 6: Official test set results. The — indicate results we could not obtain, and the NA is because we trained on data that was not allowed for that task, so we participated without ranking. For some tasks, multiple rankings are given per sub-track, for task3-2, the single mBERT based model would have ranked 3th.

We note that there are some discrepancies between scores on the test data (Table 6) and the previously reported dev scores (Table 4), these are probably the result of differences in implementation for the metric (when the official code was not released), and could sometimes be the result of uploading the data in the wrong format (e.g. task3-2). Perhaps surprisingly, the single task mBERT model sometimes outperforms RemBERT. This leads to the conclusion that we should not always blindly use the latest, larger language model. Furthermore, we see that for most tasks we rank somewhere in the middle. It should be noted that little to no tuning is done (except for MaChAmp task-type for three of our tasks: 10, 11, 12), as our focus was mostly on comparing our own models to each other and answering our research question. Results can be expected to still increase by selecting the architecture (SINGLE, MULTI, MULTI-FINE), selecting language model per task, tuning the multi-task setup (loss weighing, combination of tasks, smoothing α etc.) or the other hyperparameters (learning rate, scheduler, batch size etc.) of MaChAmp.

5 Conclusion

We have compared three setups in this work: SINGLE: single task finetuning of language models, MULTI: multi-task finetuning of language models, MULTI_FINE: using the output of MULTI and finetuning on single target tasks again. Our setup is both multi-lingual and uses pre-defined set of tasks with a large variety in types of tasks. Our results confirm the findings of recent and concurrent work (Phang et al., 2018; Aghajanyan et al., 2021), showing that for some task combinations, we can benefit from an intermediate task-trained model (MULTI_FINE). However, we also show that all three evaluated setups perform well for certain tasks. We hypothesize that this is an effect of using a pre-defined set of tasks. In our setup the differences between the setups are in some cases extremely large (error reductions larger than 40% compared to the single task baseline have been obtained for three tasks), whereas for some other tasks our single task baseline performed best. This leads to a positive answer to our research question, and the conlusion that intermediate finetuning can be beneficial. However, care should be taken, as our results also show that MULTI_FINE does not outperform MULTI nor SINGLE in all situations, which raises the question: how can we predict whether the intermediate model is better or we need to finetune one more time on the target task?

References

Ibrahim Abu Farha, Silviu Oprea, Steven Wilson, and Walid Magdy. 2022. SemEval-2022 Task 6: iSarcasmEval, Intended Sarcasm Detection in English and Arabic. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Rodrigo Agerri, Montse Cuadros, Sean Gaines, and German Rigau. 2013. OpeNER: Open polarity enhanced named entity recognition. In *Sociedad Española para el Procesamiento del Lenguaje Natural*, volume 51, pages 215–218.

Armen Aghajanyan, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. 2021. Muppet: Massive multi-task representations with pre-finetuning. *arXiv preprint arXiv:2101.11038*.

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q. Tran, Dara Bahri, Jianmo Ni, Jai Gupta, Kai Hui, Sebastian Ruder, and Donald Metzler. 2022. Ext5: Towards extreme multi-task scaling for transfer learning. In *International Conference on Learning Representations*.

Jeremy Barnes, Toni Badia, and Patrik Lambert. 2018. MultiBooked: A corpus of Basque and Catalan hotel reviews annotated for aspect-level sentiment classification. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Jeremy Barnes, Andrey Kutuzov, Laura Ana Maria Oberländer, Enrica Troiano, Jan Buchmann, Rodrigo Agerri, Lilja Øvrelid, Erik Velldal, and Stephan Oepen. 2022. SemEval-2022 task 10: Structured sentiment analysis. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, Seattle. Association for Computational Linguistics.

Ting-Yun Chang and Chi-Jen Lu. 2021. Rethinking why intermediate-task fine-tuning works. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 706–713, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hyung Won Chung, Thibault Fevry, Henry Tsai, Melvin Johnson, and Sebastian Ruder. 2021. Rethinking embedding coupling in pre-trained language models. In *International Conference on Learning Representations*.

Michael Crawshaw. 2020. Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796*.

Viet Dac Lai, Amir Ben Veyseh, Thien Huu Nguyen, and Franck Dernoncourt. 2022. Semeval-2022 task 12: Symlink - linking mathematical symbols to their descriptions. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. 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.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1923–1933, Copenhagen, Denmark. Association for Computational Linguistics.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022. Semeval-2022 task 11: Multilingual complex named entity recognition (multiconer). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Lilja Øvrelid, Petter Mæhlum, Jeremy Barnes, and Erik Velldal. 2020. A fine-grained sentiment dataset for Norwegian. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5025–5033, Marseille, France. European Language Resources Association.

Carla Perez Almendros, Luis Espinosa Anke, and Steven Schockaert. 2020. Don't patronize me! an annotated dataset with patronizing and condescending language towards vulnerable communities. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5891–5902, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Carla Pérez-Almendros, Luis Espinosa-Anke, and Steven Schockaert. 2022. SemEval-2022 Task 4: Patronizing and Condescending Language Detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Jason Phang, Iacer Calixto, Phu Mon Htut, Yada Pruksachatkun, Haokun Liu, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. English intermediate-task training improves zero-shot cross-lingual transfer too. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 557–575, Suzhou, China. Association for Computational Linguistics.

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

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. Intermediate-task transfer learning with pretrained language models: When and why does it work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5231–5247, Online. Association for Computational Linguistics.

Alan Ramponi, Rob van der Goot, Rosario Lombardo, and Barbara Plank. 2020. Biomedical event extraction as sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5357–5367, Online. Association for Computational Linguistics.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon

Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. 2022. Multi-task prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–235, Berlin, Germany. Association for Computational Linguistics.

Michalina Strzyz, David Vilares, and Carlos Gómez-Rodríguez. 2019. Viable dependency parsing as sequence labeling. 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 717–723, Minneapolis, Minnesota. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Marcos Garcia, Carolina Scarton, Marco Idiart, and Aline Villavicencio. 2022. SemEval-2022 Task 2: Multilingual Idiomaticity Detection and Sentence Embedding. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Harish Tayyar Madabushi, Edward Gow-Smith, Carolina Scarton, and Aline Villavicencio. 2021. AStitchInLanguageModels: Dataset and methods for the exploration of idiomaticity in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3464–3477, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Cigdem Toprak, Niklas Jakob, and Iryna Gurevych. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 575–584, Uppsala, Sweden. Association for Computational Linguistics.

Rob van der Goot, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021. Massive choice, ample tasks (MaChAmp): A toolkit for multi-task learning in NLP. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 176–197, Online. Association for Computational Linguistics.

Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. 2020. Exploring and predicting transferability across NLP tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7882–7926, Online. Association for Computational Linguistics.

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R. Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, Shuning Jin, Berlin Chen, Benjamin Van Durme, Edouard Grave, Ellie Pavlick, and Samuel R. Bowman. 2019. Can you tell me how to get past sesame street? sentence-level pretraining beyond language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4465–4476, Florence, Italy. Association for Computational Linguistics.

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.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Roberto Zamparelli, Absar Chowdhury Shammur, Brunato Dominique, Cristiano Chesi, Felice Dell'Orletta, Arid Hasan, and Giulia Venturi. 2022. SemEval-2022 Task3 (PreTENS): Evaluating neural networks on presuppositional semantic knowledge. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.