Applying Multitask Learning to Targeted Sentiment Analysis using Transformer-Based Models

Fábio Rodrigues Pereira fabior@uio.no
Per Morten Halvorsen pmhalvor@uio.no
Eivind Grønlie Guren eivindgg@ifi.uio.no

Abstract

Whereas sentiment analysis in general attempts to classify the polarity of an arbitrary length text document; targeted sentiment analysis aims to extract and classify the polarity of the target entities referred to in a text. This is a relatively new field for the Norwegian language as the fine-grained annotated dataset has been lacking until recently. In this paper, we explore how a transformer-based architecture (NorBERT) affects performance on the task through a variation of multitask learning architectures. We conclude that task-independent architectures (also referred to as pipeline models) perform best for target sentiment analysis, with a proportional F1-score of around 45% and a binary F1-score of around 58%. This architecture barely outperforms a collapsed transformer-based setup, mainly due to the elevated proportional score as a result of subtask optimization. Ideas for further development are discussed, along with the previous experiments and findings our hypotheses were derived from.

1 Introduction

In this work, we attempt to improve the field of targeted sentiment analysis for Norwegian by applying multitask learning to different transformer based architectures. Sentiment analysis refers to the general task of polarity classification of a text, whether that be at entity-, sentence- or document-level. For this experiment, we limit this polarity classification to a simple binary mapping of either positive or negative, applying it to the targeted entities of the text. This means our models will ultimately be solving two tasks: first find the scope of a text's targeted entities, then decide whether the target is positively or negatively referred to.

Multitask learning (MTL) frameworks have proven useful when a model objective can be broken down into multiple subtasks (Caruana, 1993). Different sharing mechanisms between the parameters of each subtask can be beneficial dependent upon the level of abstraction the subtask at hand requires. Three variations of MTL frameworks will be explored in this project, namely: hard sharing, soft sharing, and no sharing.

Naturally, targeted sentiment analysis research is highly dependent on finely annotated datasets, a challenge which until recently has not been overcome. $\text{NoReC}_{\text{fine}}$, the Fine-Grained Norwegian Review Corpus (Øvrelid et al., 2020), overcame this hurdle, opening the door to development of such models for Norwegian text. A curated version of this dataset will be used in this experiment.

Self-attention based transformers quickly replaced the current golden standard of language models for a range of different tasks (Devlin et al., 2019). Consequently, the recently released Norwegian variation of BERT, NorBERT (Kutuzov et al., 2021), has unlocked much untapped potential in the realm of natural language processing in Norway. Further, Barnes et al. 2021 suggests that jointly predicting target and BIO polarity labels (for English sentiment analysis) improves targeted polarity classification. We will adapt this approach to targeted sentiment analysis for Norwegian and see if the hypothesis holds.

This paper will explore how these state-of-the-art transformer-based language models affect the task of targeted sentiment analysis by comparing different multitask setups against a standard bidirectional LSTM baseline.

2 Dataset

The provided dataset used in this project consists of texts from the Norwegian Review Corpus (NoReC) (Øvrelid et al., 2020), which is a large collection of reviews spanning different categories including literature, restaurants, and consumer products, among

others.

The texts were preemptively split into sentences and presented in a simplified `Conll`-format, which are annotated with BIO-tags and target polarities for each entity. This means there are a total of five different labels: B-targ-positive, I-targ-positive, B-targ-negative, I-targ-negative and O.

The sample space was split into train, dev, and test datasets. Table 1 shows the distributions of each set. As there are two possible polarities for each target, Table 2 shows a more detailed overview of the distribution of the BIO labels.

	Train	Dev.	Test	Total	Avg. len.
Sents.	5914	1151	895	7960	16.8
Targets	3339	629	511	4479	2.0

Table 1: Dataset details

Labels	Train	Dev.	Test	Total
B	3339	629	511	4479
I	3453	643	465	4561
O	91686	18336	14425	124447
Positive	4584	869	714	6167
Negative	2208	403	262	2873

Table 2: More detailed dataset specifications

As seen in Table 2, all the data sets contain roughly 30x more 'O' tags than others, and about twice as many positive polarities than negative. These imbalances tell us that accuracy will not be a valid metric when comparing models, since different random (unstratified) splits could grossly affect a model's evaluation. Therefore, an F1-score will instead be used as the main evaluation metric.

3 Architectures

The novelty this experiment brings to the table comes from the combination of multitask architectures with transformers. We will attempt to achieve a new state-of-the-art architecture specifically for targeted sentiment analysis in Norwegian. As a control, a baseline will be defined in order to gauge performance improvements when training and evaluating on our particular dataset.

3.1 Baseline

The baseline approach employs a bidirectional long short-term memory model (BiLSTM) on top of different pre-trained corpora[1], built particularly from Word2Vec, fastText, and Global Vectors algorithm types. We use BiLSTM because of its more complex cell structure than a typical recurrent neuron (Schuster and Paliwal, 1997). In addition, its sequential learning processes of predicting what is relevant and forgetting what is unnecessary make it an appropriate choice for our task of extracting sentiment targets from sentences and classifying their polarity (Huang et al., 2015). For this architecture, a collapsed prediction is made on the labels, meaning that both BIO-tag and polarity are predicted together.

This BiLSTM model receives as input the ids of the tokenized sentences. Unfrozen embeddings are retrieved from the selected pre-trained corpus, meaning the embedder can learn from each new input. Cross-entropy loss between predictions on the development set and that set's real targets help teach the model what it's predicting correctly, and what needs to be changed. Validations are calculated by the so-called Binary F1-score and Proportional F1 score. The former considers if the joint entity and polarity is or is not an exact match with the real classification, while the latter measures the weighted F1-score between predicted and real labels (BIO + polarity).

Hyperparameter searches were performed in order to improve the performance of the model and increase metrics. Specifically, the best embedding corpus, the number of hidden layers, hidden dimension, learning rate, dropout, batches, and epochs were optimized.

3.2 Simple Transformer

The simple Transformer model, also known as a Collapsed models (Hu et al., 2019) combines multi-targets span entities (BIO system) and sentiment polarities (Positive, Negative) into only one label space. The example in Figure 1 reveals the idea behind the predictions.

3.3 Multitask Learning

Moving on from simplistic baselines, multitask setups split complex model objectives into a simpler tasks, allowing sub-models to get really good at predicting different parts of the final expected output. In our experiment, the final output can be broken down into a name-entity recognition task along with a polarity classification task.

Loss is calculated per task, according to the desired parameter sharing mechanism between tasks. The three sharing approaches explored here will

[1] NLPL Repository: http://vectors.nlpl.eu/repository/

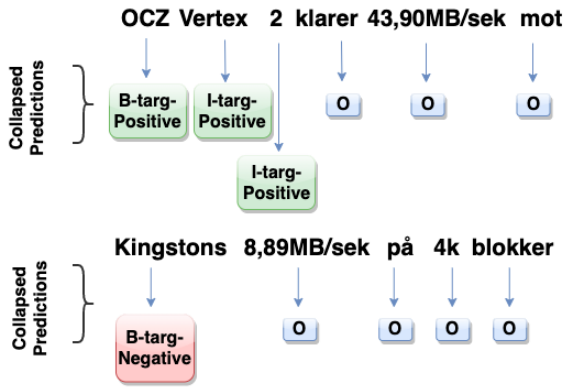

Figure 1: Collapsed predictions

be hard sharing (Caruana, 1993), soft sharing (Liu et al., 2016), and no sharing.

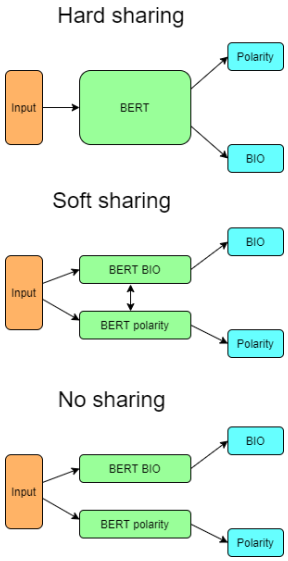

Figure 2: Parameter sharing variations

3.3.1 Hard Sharing

The multitask setup closest to the baseline transformer architecture previously presented consists of shared lower levels for both tasks, with only the final output classification layers kept private per task. This is known as *hard parameter sharing*, and was first presented in Caruana 1993. When the tasks are closely related, hard sharing has proven to be an effective way of increasing baseline performance (Barnes et al., 2020).

This implementation is achieved by maintaining two loss criterion, both of which using a cross entropy algorithm for the same reasons mentioned in Section 3.1. Here the shared parameters plus the private parameters are specified to update for each criterion, meaning the shared layers are updated twice per batch, one time for each task.

3.3.2 Soft Sharing

Another approach to sharing parameters between models which gives more sub-model flexibility than above is a *soft-sharing* technique. These frameworks are basically an individual model built for each task, with similar lower levels so that a constraint can be added to *encourage* similarities without *forcing* them. It has been shown that hard parameter sharing is beneficial for low-level auxiliary tasks, where high level tasks benefit more from a softer sharing mechanism (Sanh et al., 2018). While both name-entity recognition and polarity classification is deemed a low-level task, a soft parameter sharing framework was still tested here to either support or deter these findings.

In our experimenting, we implement an L_2 regularization between the softly-shared parameters, to mitigate extreme differences between parameter values respective to each of the models (Navon, 2019). Total loss then becomes a combination of the individual losses for each task, with this regularized difference between low level parameters. In other words, the sub-models are allowed to focus on their respective tasks, but need to also make sure not let the shared parameters vary too far from the other model's shared parameters.

3.3.3 No Sharing

The most extreme variant of a multitask learning framework tested in this experiment was a no-share setup. This setup goes one step further than soft-sharing, allowing for both tasks to fine-tune individual sub-models for their own respective tasks, without any knowledge of what's going on in the other sub-models. Optimization is still done batch-wise, and the outputs from both sub-task models are concatenated as before, making this setup a multitask learning framework.

3.4 Task-independent models - pipelines

This proposed framework, also called the pipeline model (Hu et al., 2019), is divided into two sub-tasks: Multi-target extraction for distinguishing entities from the input sentence, then, polarity classification for the extracted entities acquired in the previous task. The visualization of the steps is presented in Figure 3.

The first task has its own standalone network, which receives input ids from a given sentence,

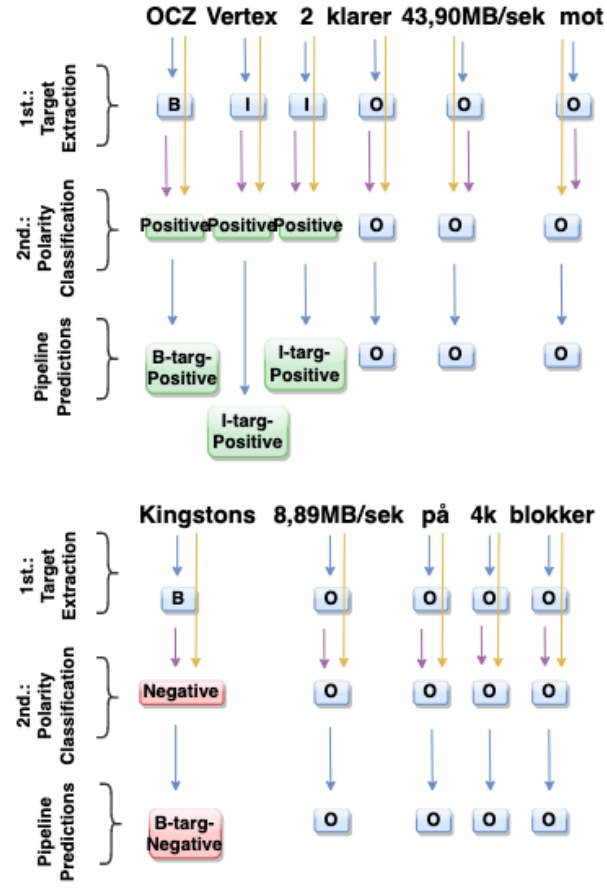

Figure 3: Arrows blue and yellow show the completely independent multi-tasks. Arrows yellow and purple show the mean or last multi-tasks.

tokenized by NorBERT. The model is fitted and optimized solely focusing on extracting BIO tags, i.e. polarity is completely disregarded for this first step. When the BIO model is fully optimized, there exist two different approaches for the next task:

1. The first is to train the second task utterly detached from the first task, taking the tokenized input ids from the provided sentences and feeding them directly to the transformer polarity model to predict polarities (Neutral, Positive, Negative) for each token.

2. The second takes the last or mean of the generated embeddings by the hidden output states from the first task, and they are passed to the polarity model as input embeddings to predict polarities (Neutral, Positive, Negative).

For this second scenario, we utilized two ways of stacking the hidden representations from the BIO submodel before feeding them to the polarity model:

1. '**mean**', where a simple average is broadcasted among all the output hidden states' at its layers' dimension;

2. '**last**', where only the final output hidden states' tensor is utilized.

Thus, for all approaches, the polarity classification is performed to provide contextual sentence vectors. In the end, since both models are individually trained, they are coupled as a pipeline for inferences.

The difference between this setup and the no-share setup mentioned in Section 3.3.3 is when the submodels were optimized. Here, each independent model is optimized as a standalone model, whereas in the no-share setup, each subtask is optimized batch-wise. Since little to no variation was observed between the two variations, only this task independent setup will be commented on in the results.

4 Hypothesis

The introduction of self-attention transformers to NLP came with multiple state-of-the-art architectures for a range of different tasks (Devlin et al., 2019), including targeted sentiment analysis. This ultimately replaced previously reigning architecture of BiLSTMs with a CRF output layer (Huang et al., 2015), which tells us that even **our simplest transformer based architecture should outperform the simplistic BiLSTM model**[H1].

Further, it has been shown that modeling subspaces of a main task explicitly as auxiliary tasks can help improve final performance of that main task (Caruana 1993, Ruder et al. 2019, Barnes et al. 2020). We hypothesize that **all of our multitask setups which incorporate BERT for at least one task will outperform the simple transformer-based architecture**[H2].

5 Experimenting

Even though our hypotheses were drawn from previously documented experiments, none of those experiments were run on our specific dataset, for our specific task (or even in the language which we are focusing on). This meant that a new baseline was necessary to provide a reliable estimate of how much performance actually improved from our architectural adjustments.

The skeleton code provided alongside the dataset gave a simplistic example of a BiLSTM model, but

left much room for optimization. As mentioned in Section 3.1, hyperparameter searches were executed to find the configuration of our BiLSTM setup that gave the best performance.

Once this was found, a similar study was applied to our simple transformer model. This was to highlight the ground-breaking improvements BERT models brought to natural language processing (Devlin et al., 2019), giving a proper gauge of any true improvements in performance our novel architectures presented. The model is built from a BertForTokenClassification object combined with NorBERT pre-trained corpus to encode a given sentence. Then, the forward pass is fed with a batch of filtered and padded input ids and attention masks. Later, the framework computes the loss and updates the learning weights (hidden states) at the end of the backward step. Validations and inferences are additionally applied in the experiments, everything with independent and unseen development and test datasets from the learning process.

The next step was then to test the different forms of parameter sharing in multitask setups. The natural first setup to test was the hard sharing model, since this setup most resembled the simple transformer; the only difference between the two being singular linear outputs versus dual linear outputs. In order to evaluate in the same manner as before, these dual outputs needed to be joined to resemble the joint labels of the original dataset.

When calculating loss on these multitask setups, it's important to consider how large of a learning rate is necessary for robust performance. If this rate is too high, the model focuses only on learning the easiest thing to classify, i.e. O's. Running evaluations on predictions from these models gives both binary and proportional F1 scores at around 0, since O's are not included in the scoring. A learning rate that is too low means that the number of epochs needed to achieve any useful results skyrockets. Given our computational limitations, redundant jobs should be avoided as much as possible. The possibility of fine-tuning learning rates per task was also possible, since each task has it's own loss criterion. However, not much time was left over for hyperparameter tuning here.

Moving on to the soft-share model, the largest adaptation from the hard-share setup needed was the addition a new BERT-object to the network, so that the BIO-labeling task and polarity classification tasks each had their own. For simplicity, only linear outputs were used for both models (although, in future experiments, different variations of outputs can be tested). Again, two loss criterion were used, but now only with parameters respective to each model. The soft-sharing was implemented using an L_2 normalization between all the parameters of the BERT models, meaning the linear outputs were excluded here. This regularization encourages similarity between these analogous parameters simply by treating the difference between the two respective parameters as a loss. The greater the difference, the higher the loss, and the more the model is penalized.

The no-share model very closely resembled the soft model explained above, just without the L_2 regularization. Otherwise, outputs and evaluations were executed in the same fashion.

Finally, the task-independent model was built, completely isolating the two tasks from each other. Here, each loss criterion were optimized individually, obviously due to the nature of this model.

6 Results

Our fully optimized baseline model gave a safe foundation for our experimenting, with the lowest performance of any model tested. Also, as expected given the results presented in BERT's introductory white-paper (Devlin et al., 2019), the out of the box transformer model increased main task performance from the baseline by a significant amount, from 22.6% to 43.9%. Both of these models underwent hyperparameter searches, so it can be assumed that these scores are as good as they can get.

Not all of the multitask learning frameworks were fully optimized due to computing limitations. For example, providing different learning rates for each subtask could give the model more room to learn what each task requires. This lack of optimization is probably the reason why both hard and soft parameter sharing underperformed compared to our expectations. Scoring worse than the simplistic BiLSTM model, we see that a more thorough fine-tuning process should be run on these models to unlock this architecture's full potential. This poor performance is portrayed through the results presented in Table 4.

Given the conclusions drawn from Barnes et al. 2020, the hard-share setup should have performed better than the soft-share. This is due to the simplicity of these tasks, i.e. they both do not heavily

depend on the structure of the sentence. Rather, the immediate context of a label's scope is usually enough for the mode to predict correctly (Sanh et al., 2018).

The pipeline framework gave the best performance when considering proportional F1-scores as the evaluation metric, i.e. task-wise class labeling is better than joint and collapsed class labeling. Comparing against our baselines, pipeline beat Simple BERT by about 1%, with 45.0% versus 43.9% respectively using this metric. When considering Binary F1, however, the pipeline model scored 58.2% compared to the 58.3% of the simple BERT. This means that the main task of labeling both BIO tags and polarity saw little to no change in performance on split setups versus joined predictions. For reproducibility, the parameters obtaining these results are presented in Table 3. Additionally, a full overview of all the results can be found in the GitHub-repository, in the output directory.

Parameter	BIO	polarity
Epochs	1	9
Learn rate	0.0001	0.00001
Momentum	0.9	0.9
Loss function	cross-entropy	cross-entropy
Optimizer	AdamW	AdamW
Random state	1	1

Table 3: Optimal Pipeline Hyperparameters

The benefits of using pipeline models are relatively small compared to the other structures. Besides, it infers a weak correlation between the tasks (target extraction and polarity classification), confirming the inferences from (Hu et al., 2019). Conversely, we observe that both tasks still have a relationship with the input sentence. It reasonably explains the higher metrics than joint and collapsed models.

A full overview of how the models compare can be found in Table 4

7 Conclusion

Our experimenting found that a pipeline multitask framework got the highest score for our targeted sentiment analysis task in Norwegian, even though performance was very close to that of the simple transformer model. Understandably, we observed improvements in proportional F1 scores when that tasks were split, since those setups had models (or even just a few layers) that were optimized for predicting the individual tags corresponding to their layer's subtask.

Model	Prop. F1	Binary F1
Baseline	0.226	0.345
Simple BERT	0.439	**0.583**
Hard-share MTL	0.411	0.523
Soft-share MTL	0.398	0.490
No-share MTL	0.401	0.527
Pipeline MTL	**0.450**	**0.582**
Mean MTL	0.382	0.579
Last MTL	0.380	0.582

Table 4: Final performance on Test Data

Our hypothesis that the simple BERT model would outperform our BiLSTM baseline (**H1** in Sec. 4) held true, supporting the findings of Devlin et al. 2019.

The second hypothesis that all of our multitask setups which contained at least one BERT object should outperform the simple transformer model (**H2** in Sec. 4) was wrong, for the most part. The split setup (pipeline) model did show some improvements from this model proportion-wise, but barely failed in comparison for exact matches. In theory, our hard sharing model should have beaten the simple transformer, due to the complexity of BIO sequence labeling and polarity classification at token level.

8 Further development

There were a few areas of optimization that would be interesting to dig further into. As mentioned, the learning rate of the different sharing models would have been interesting to run detailed studies on. Also testing non-linear heads of all of the models could help increase performance, as many groups found in Obligatory 3.

The data used in this experiment were a curated subset of the larger NoReC set (Velldal et al., 2018), where full documents are given polarity classifications. Another similar dataset exists where polarity is annotated per sentence (Mæhlum et al., 2019). Combining the overlapping datapoints of these three sets, a potential future experiment could explore how performance of our architectures changes with this higher-level information incorporated.

Acknowledgments

We thank the Language Technology Group for their great work on the NoRec fine dataset. Without their contribution to the field, this project would not be possible.

References

Jeremy Barnes, Lilja Øvrelid, and Erik Velldal. 2021. If you've got it, flaunt it: Making the most of fine-grained sentiment annotations. *ArXiv*, abs/2102.00299.

Jeremy Barnes, Erik Velldal, and Lilja Øvrelid. 2020. Improving sentiment analysis with multi-task learning of negation. *Natural Language Engineering*, 27(2):249–269.

R. Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *ICML*.

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.

Minghao Hu, Yuxing Peng, Zhen Huang, Dongsheng Li, and Yiwei Lv. 2019. Open-domain targeted sentiment analysis via span-based extraction and classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 537–546, Florence, Italy. Association for Computational Linguistics.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Andrey Kutuzov, Jeremy Barnes, Erik Velldal, Lilja Øvrelid, and Stephan Oepen. 2021. Large-scale contextualised language modelling for norwegian.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning.

Petter Mæhlum, Jeremy Barnes, Lilja Øvrelid, and Erik Velldal. 2019. Annotating evaluative sentences for sentiment analysis: a dataset for Norwegian. In *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 121–130, Turku, Finland. Linköping University Electronic Press.

Aviv Navon. 2019. Parameter sharing in deep learning.

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.

Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, Minneapolis, Minnesota. Association for Computational Linguistics.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2018. A hierarchical multi-task approach for learning embeddings from semantic tasks.

Mike Schuster and Kuldip Paliwal. 1997. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, 45:2673 – 2681.

Erik Velldal, Lilja Øvrelid, Eivind Alexander Bergem, Cathrine Stadsnes, Samia Touileb, and Fredrik Jørgensen. 2018. NoReC: The Norwegian Review Corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).