# Aspect-Based Sentiment Analysis

**Pavlo Vasylenko**
vasylenko.1880258@studenti.uniroma1.it

## 1 Introduction

In this work, we wiil explore Aspect Based Sentiment Analysis (ABSA) via pre-trained language model RoBERTa as a core of architecture of neural network (NN) models. In particular, we will explore 3 approaches and their variations: end-to-end sequence labeling, splitting aspect and sentiments predictions into two branches trained together and auto-regressive generation via fine-tuning Bart. In addition, aspect category identification (ACI) and aspect category polarity classification (ACPC) are simply presented as sequence classification task with separate classifiers trained together.

## 2 BERT vs RoBERTa

After checking BERT tokenizer, it turned out that it splits words with hyphens without adding extra information which leads to inappropriate merging in a decoding phase. On the other hand, the RoBERTa tokenizer decodes in such way that words with hyphens can be properly reconstructed Pic 1. However, in order to compare BERT and RoBERTA performance, we also trained models with architecture described in (Li et al., 2019) (also in section 3.1) - E2E-ABSA - where we set the encoder to be either BERT or RoBERTa. They were trained 10 times with early stopping and their accuracies, for tokens that are not $O$ or $[PAD]$, were averaged to exclude noise. The obtained results are: BERT - 53% and RoBERTa - 63%. To be mentioned, we showed comparison in terms of accuracy to omit cases where there are words with hyphens predicted correctly but decoded incorrectly by BERT tokenizer thus affecting F1 score. Therefore, selection for RoBERTa as an encoder is also better in terms of performance of the introduced metric, hence we used RoBERTa in all our models.

In addition, from the beginning, we fine-tune RoBERTa encoder and we do not explore models without doing that.

## 3 Methods

### 3.1 E2E-ABSA

We have chosen an approach introduced in (Li et al., 2019) as our first model but, instead of BERT we use RoBERTa. The model consists of the bidirectional encoder that produces contextualized embeddings and the classsifier on top of the encoder that take those embeddings and predicts following classes: B-{POS, NEG, NEU, CON}, I-{POS, NEG, NEU, CON}, E-{POS, NEG, NEU, CON}, S-{POS, NEG, NEU, CON} or O. There are 17 classes in total which is greater than in the original paper since we have one more sentiment class - conflict. As a classifier, we utilized a 1-layer feed-forward neural network (FNN) as described in the paper. Therefore, the model (E2E in our report) consists of one Cross Entropy (CE) loss.

### 3.2 Multi-task E2E-ABSA

It is well known that multi-task learning can improve performance of the original task. Moreover, for quantized problems (classification), using less quantized (continuous) auxiliary tasks might be beneficial (Ruder, 2018). We explored an extra task in which we predict 4-dimensional vector of term frequencies (TF) of sentiments in the sentence: number of occurrences of a particular sentiment divided by the total number of aspects in the sentence. For instance, if we have 3 positive aspects and 1 conflict we will get a vector: $(0.75, 0, 0, 0.25)$. Desired vector is predicted from produced embeddings of the first token $< s >$ of the input sequence. It is fed to 2-layer FNN with the output layer of dimension 4. From this it follows, the model contains 2 losses: CE Loss for original sequence tagging problem and Mean Squared Error (MSE) Loss for predicting a TF vector. The later is continuous prob-

lem which, according to (Ruder, 2018), might be helpful for the original task. We called the model E2E-TF.

### 3.3 Cascade models

More advanced solution for ABSA is (Luo et al., 2020) where authors split a model into two branches: the first one (encoder) predicts sequence of B, I, E, O for aspects and the second (decoder) predicts sentiment classes for that aspect tokens. To achieve that, the encoder shares outputs of 9-th hidden layer with the decoder so $K$ (keys) and $V$ (values) for the decoder transformers are outputs of the shared layer, whereas $Q$ (query) is decoder's inputs. In the first layer, $Q$ is a sequence of predicted aspect tokens, while, on the subsequent layers, $Q$ is outputs of previous layers Pic. 5. It must be mentioned that losses from two branches are added and trained together. In the training mode, decoder's inputs are true aspect labels from O, B, I, E - teacher forcing. We explored simplified approach without VAT training and trained models with Gradient Harmonized Loss (Cascade-GHL) and without (Cascade) . For GHL, weights are calculated for each training sample based on its gradient norm which tells if examples are easy or hard (Luo et al., 2020). After that, the weights are used in weighted cross entropy which by authors statements should deal with imbalances in the data. Therefore, our model consists of two cross entropy losses for the Cascade model and for Cascade-GHL - one CE Loss for aspects and GHL for sentiments.

### 3.4 Cascade model with separate binary classifiers

There might be a problem with detecting conflicts and neutral sentiments in aspects, hence (Tan et al., 2019) proposed to train a model with two separate classifiers for positive and negative sentiment and determine neutral and conflicts from combination of positive and negative predictions: (1, 0) - positive, (0, 1) - negative, (0, 0) - neutral, (1, 1) - conflict, where tuple ($y_p$, $y$) are binaries positive and negative predictions respectively. We applied this idea to Cascade model described above having two classifiers instead of one in the sentiment branch. Likewise original solution, in the new model (Cascade-Bin), the loss for aspect classifier and two losses for positive and negative sentiment classifiers were added and trained together.

In addition, we explored aggregation of tokens for sentiment classification. When aspect length is greater than 1, we can encounter a problem when for each token we predicted different sentiments. To eliminate such problem and predict one sentiment for the whole aspect we can aggregate output embeddings of aspect tokens in one vector and then predict sentiment. For this case, we used aggregation based on attention mechanism like in (Hu et al., 2019): we train a vector with parameters that is used as an attention key to find attention weights for each token embeddings to be summed up.

In models Cascade-Bin and Cascade-Bin-Agg, since data is unbalanced we calculate ratio of (1, 0) classes separately for positive and negative classification tasks and use found weights as thresholds in the evaluation phase to obtain binary classes.

### 3.5 Sequence-to-sequence model

Inspired by (Cao et al., 2021), we tried to generate aspects and sentiments auto-regressively. When we do auto-regressive generation via transformers, each token can attend directly other tokens, hence auto-regressive generation might be useful if there is hidden relation between context, aspects and sentiment. We fine-tuned Bart which is trained for sequence-to-sequence tasks, in particular denoising an input sequence (Lewis et al., 2019).

The new constructed dataset consists of original sentences as input, whereas target sentences are modification of original ones with brackets encompass aspects and a vertical pipe followed by aspect's sentiment Pic.2.

Moreover, we explored Bart ability to reconstruct sentences with mask tokens in order to improve performance. We come up with extra task that predicts the same target sequence but the input sequences is a target sequence where aspects are replaced with $<mask>$ tokens Pic.2. Therefore, the idea is that the model tries to predict an aspect based on context and a given sentiment.

### 3.6 ACI + ACPC model

Since category identification is a multi-label problem, the simplest solution is to create a separate classifier for each class. The model for categories and their sentiment prediction is similar to E2E-ABSA but, instead of classifying each token from the input sequence, we utilize the first token to predict classes. The first token ($<s>$ - in RoBERTa, $[CLS]$ - in BERT) is usually used for sentence classification (Devlin et al., 2019). In our model, the output embedding is fed to 5 (number of category classes) binary classifiers with two linear layers.

The first linear layer is also shared with sentiment classifier Pic. 3. We use here two losses: one is Binary CE loss for categories and CE Loss for sentiments which are summed up and trained together.

## 4 Prediction generation

For E2E and Cascade (all except with aggregation) models we do reverse procedure of finding tokens that belongs to B, I, E classes and grouping them together. Sentiment is found by majority rule, for instance, if among predicted tokens there are two positive and one negative we set it to be positive. In Cascade-Bin-Agg, we predict sentiment for the whole aspect straight away.

Sequence-to-sequence model (Seq2seq) generates results inside predicted sequence, thus all we need is just to parse output sequence. For generation, we use Beam Search with number of beams equal to 10 as described in (Cao et al., 2021).

## 5 Settings

For training all models, we merged Laptop and Restaurants datasets in one dataset. We used RoBERTa-base with original hyper-parameters. Each model is trained with AdamW optimizer (suggested for training transformers by $huggingface.co$). Additionally, for hyper-parameters optimization, we use early stopping to stop when specified metric does not improve. The metric to be monitored for all models is Macro F1. Macro F1 is the better metric since it was observed that if we continue training, validation loss worsens whereas Macro F1 improves Pic 7. Hyper-parameters optimization space and models for which its applied are shown in Table 2. Training parameters like learning rate and batch size are taken from original papers and shown in the Table 1. Regarding E2E model, we trained it 10 times with different seed without a scheduler.

In Cascade and Cascade-GHL, first, aspect branch is trained for 3 epochs and, after that, aspect and sentiment branches are trained together. Such training is described in the original paper (Luo et al., 2020) but with 5 epochs in the beginning but we noticed that 3 epochs are enough.

For Seq2Seq model, we used Bart-large and utilized gradient accumulation, since we could manage to train only with small batches.

## 6 Results

Table 4 shows performances for all models that we explored. As it can be noticed, the best model that we obtained is model with extra task of TF vector prediction. The best parameters for this model are: dropout in TF task - 0.1 and no scheduler in optimization. E2E model achieved the same Micro F1 and if we look at performance for each sentiment in Table 3, we can observe that noticeable difference is in conflict column, hence extra TF task might help to detect conflicts. We collected the best F1 scores of conflict sentiment for E2E and E2E-TF and plotted histograms Fig 6 and showed the mean and the standard deviation of their distributions in Table 9. E2E-TF distribution is shifted to the right and its mean is larger than E2E by 13%.

Even though, we expected increase in F1 for conflict and neutral sentiments in Cascade-Bin model, the simplest Cascade performed the best for all sentiments expect negative (Table 3). GHL has not improved results but did the opposite. Moreover, GHL impaired aspect extraction during the second phase when aspect and sentiment branch are trained together Pic 4. The reason might be that we used GHL only in the sentiment branch whereas in (Luo et al., 2020) they applied GHL to both.

The Seq2seq-Multi model showed very good performance on validation dataset during training. Obtained Macro and Micro F1 are 66.5 and 75.2 respectively. Moreover, for hyper-parameters in Table 6 we achieved 88.3 F1 for aspect extraction which is pretty good: for example in (Luo et al., 2020), there are 87.93 for laptops and 85.45 for restaurants. However, it should be mentioned that results in Table 4 are from the training phase, whereas for auto-regressive generation with Beam Search, we could not achieve comparative results Table 7. The problem with generation via Beam Search that it often produces repetitions but we cannot use $no\_repeat\_ngram\_size = 2$ since, then, it cannot generate multiple answers like " $...[...|positive]...[...|positive]$ ", since it forces the second prediction to contain a different word from "positive". However, we can assume that the model works relatively good in sentiment detection, since for aspect extraction of 73.6 it achieves Macro F1 52.2 whereas Cascade-GHL with extraction of 80.3 - only 53.1. Thus, better generation can potentially lead to good results.

Finally, Table 8 shows results for ACI and ACPC. The simplest solution brought Macro F1 of 62.9.

| Model | LR | BS | ES |
|---|---|---|---|
| E2E | $2 \cdot 10^{-5}$ | 25 | Macro F1 |
| E2E-TF | $2 \cdot 10^{-5}$ | 25 | Macro F1 |
| Cascade | $3 \cdot 10^{-5}$ | 32 | Macro F1 |
| Cascade-GHL | $3 \cdot 10^{-5}$ | 32 | Macro F1 |
| Cascade-Bin | $3 \cdot 10^{-5}$ | 32 | Macro F1 |
| Cascade-Bin-Agg | $3 \cdot 10^{-5}$ | 32 | Macro F1 |
| Seq2seq | $5 \cdot 10^{-5}$ | 16 | Macro F1 |
| Seq2seq-Multi | $5 \cdot 10^{-5}$ | 8-10 | Macro F1 |

Table 1: Training parameters: LR - learning rate, BS - batch size, ES - early stopping metric.

## Tables and Figures

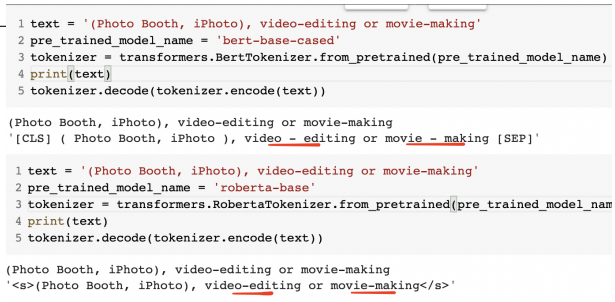


Figure 1: Comparison of BERT (top) and RoBERTa (bottom) tokenizers. It can be clearly seen that BERT tokenizer, after decoding tokens produced by itself, splits words with hyphens in 3 different entities. On the other hand RoBERTa tokenizer works correctly.

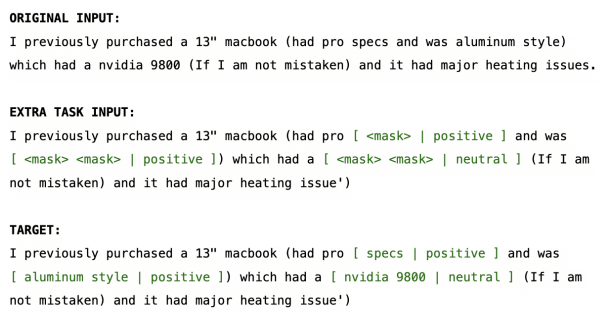


Figure 2: An example of input and output sequences for sequence-to-sequence model. Extra task can be either with multiple masks per one aspect (as shown in the picture) or one mask token for one aspect regardless of its form - we consider this option as hyper-parameter and name it as 'Single mask' in Table 2, hence it has two values: True, False.

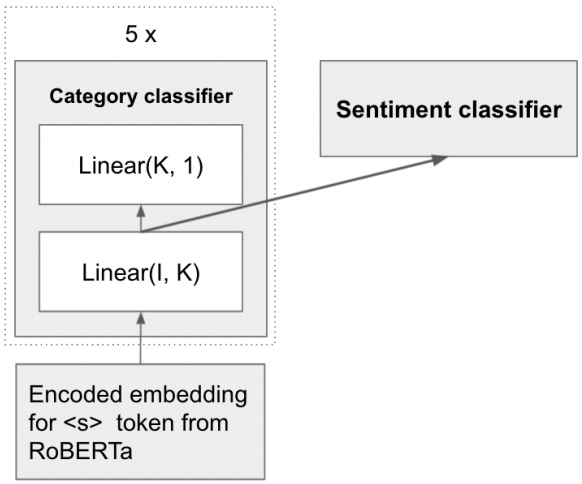


Figure 3: Scheme of category classification. Encoded embedding for the first sequence token is fed to the category classifier (5 for each class) with 2 linear layers. The first linear layer is also shared with a sentiment classifier.

| Model | Space |
|---|---|
| E2E-TFIDF | **TF task dropout:** [0, 0.1, 0.2, 0.3, 0.4] **Scheduler:** [None, Linear - 10 epochs] |
| Cascade | **RoBERTa dropout:** [0.1, 0.2, 0.3] **Sentiment branch dropout:** [0.0, 0.1, 0.2] **Scheduler:** [None, Linear - 10 epochs] |
| Cascade-GHL | *The same as in Cascade* |
| Cascade-Bin | **Sentiment branch dropout:** [0.0, 0.1, 0.2, 0.3] **Sentiment transformer layers:** [2, 3] **RoBERTa layer to share:** [8, 9, 10] |
| Cascade-Bin-Agg | **Sentiment branch dropout:** [0.0, 0.1, 0.2, 0.3] |
| Seq2seq | **Bart dropout:** [0.1, 0.2] **Single mask:** [True, False] **Scheduler:** [None, Linear - 10 epochs] **Weight Decays** [0, 0.001] |
| Seq2seq | **Bart dropout:** [0.1, 0.2, 0.3] **Single mask:** [True, False] **Scheduler:** [None, Linear - 10 epochs] **Weight Decays** [0, 0.001] |
| Seq2seq-Multi | *The same as in Seq2seq + 0.3 dropout* |
| ACI+ACPC | **Dimension in shared layer** [196, 392, 1024] **Scheduler:** [None, Linear - 10 epochs] |

Table 2: Spaces of hyper-parameters optimization. TF task dropout - dropout between layers in 2-layer FNN predicting TF vector. Scheduler: None - no scheduler in optimizer, "Linear - 10 epochs" - linear scheduler with warm-up. Sentiment branch dropout - dropout for transformers layers in FNN part. Sentiment transformer layers - number of transformers layers stacked in the sentiment branch. RoBERTa layer to share - RoBERTa layer index that shares its output with the sentiment branch. Bart dropout - dropout of FFN block in transformers. Single mask - described in Fig. 2

| Model | Pos. | Neg. | Neu. | Conf. |
|---|---|---|---|---|
| E2E | 74 | 67.7 | 52.8 | 41.5 |
| E2E-TF | 74.9 | 65.6 | 54.6 | 50 |
| Cascade | 72.6 | 63.7 | 49 | 40.8 |
| Cascade-GHL | 71.6 | 62.9 | 39 | 39.2 |
| Cascade-Bin | 72 | 62 | 49.3 | 35.5 |
| Cascade-Bin-Agg | 71.9 | 64.1 | 45 | 32.7 |
| Seq2seq | 81.8 | 73.7 | 61.8 | 48.7 |
| Seq2seq-Multi | 81.9 | 71.4 | 57.6 | 43.5 |

Table 3: Models performance for each sentiment class (F1)

| Model | Micro F1 | Macro F1 |
|---|---|---|
| E2E | 67.4 | 59 |
| E2E-TF | **67.4** | **61.3** |
| Cascade | 65.3 | 56.5 |
| Cascade-GHL | 63.05 | 53.1 |
| Cascade-Bin | 63.7 | 54.7 |
| Cascade-Bin-Agg | 63.7 | 51.7 |
| Seq2seq | 73.4* | 63.6* |
| Seq2seq-Multi | 75.2* | 66.5* |

Table 4: Performance of models in terms of Macro F1. For Seq2seq and Seq2seq-Multi, we showed results during training phase, but not generation phase.

| Model | F1 |
|---|---|
| E2E | 84.9 |
| E2E-TF | 84.7 |
| Cascade | 85.7 |
| Cascade-GHL | 82 |
| Cascade-Bin | 85.6 |
| Cascade-Bin-Agg | 82.7 |
| Seq2seq | 86.6 |
| Seq2seq-Multi | 88.3 |

Table 5: Performance of models for aspect extraction task. Shown F1 is the maximum value encountered during the training. Therefore, the best models based on Macro F1 can have different F1 for aspect extraction.

| Parameter | Value |
|---|---|
| Bart dropout | 0.2 |
| Weight decay | 0 |
| Scheduler | None |
| Single mask | False |

Table 6: The best hyper-parameters for Seq2seq that result in the best aspect extraction performance of 88.3

| F1 | Value |
|---|---|
| Macro | 52.2 |
| Micro | 61.6 |
| Extraction | 73.6 |

Table 7: Performance of Seq2seq model measured after auto-regressive generation via Beam Search with number of beams 10.

| F1 | Value |
|---|---|
| Sentim. Macro | 62.9 |
| Sentim. Micro | 70.4 |
| Positive | 78.2 |
| Negative | 68.9 |
| Neutral | 53.3 |
| Conflict | 51.3 |
| Extract. Macro | 86.5 |
| Extract. Micro | 87.6 |
| Miscellaneous | 83.7 |
| Price | 88.9 |
| Food | 93.7 |
| Ambience | 76.7 |
| Service | 89.2 |

Table 8: Performance of the model for aspect category identification and aspect category polarity classification

| Model | Mean | Std |
|---|---|---|
| E2E | 37.1 | 5.6 |
| E2E-TF | 42.2 | 5.1 |

Table 9: Mean and standard deviation for distributions in Fig. 6

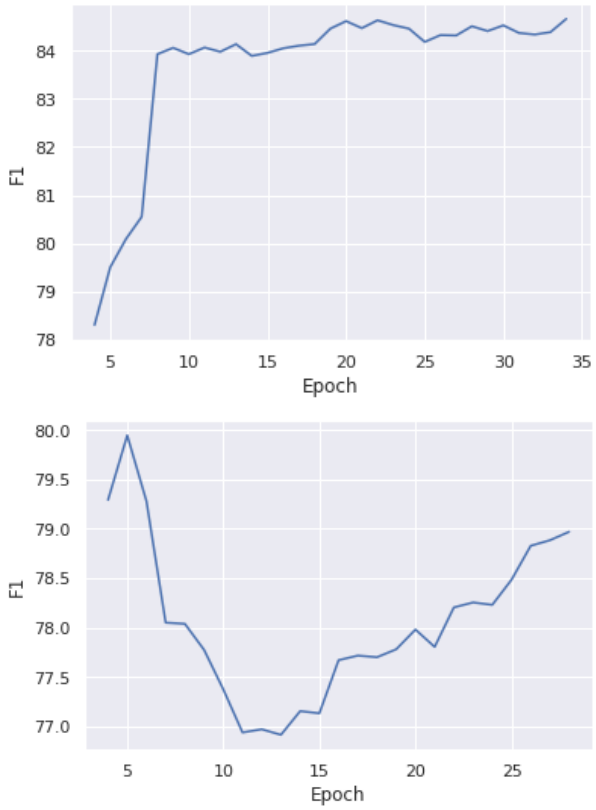


Figure 4: Aspect extraction F1 of Cascade (Top) and Cascade-GHL (Bottom) models (the plot is moving average with window=5). The models are trained with the same hyper-parameters. We can observe that after, the first epochs where model is trained only for aspect extraction (no GHL there), GHL impair aspect extraction for the whole model.

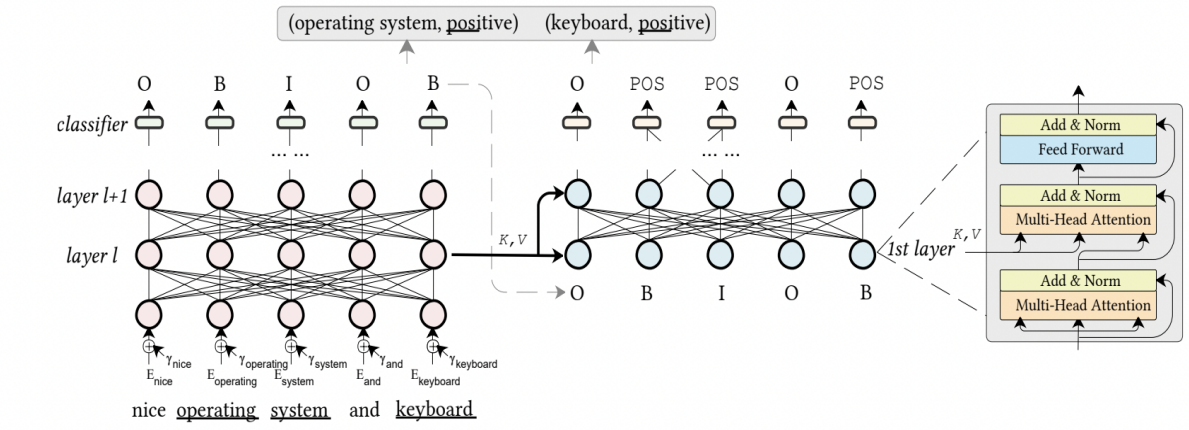


Figure 5: Cascade model

## References


Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Minghao Hu, Yuxing Peng, Zhen Huang, Dongsheng Li, and Yiwei Lv. 2019. Open-domain targeted sentiment analysis via span-based extraction and classification.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Xin Li, Lidong Bing, Wenxuan Zhang, and Wai Lam. 2019. Exploiting bert for end-to-end aspect-based sentiment analysis.

Huaishao Luo, Lei Ji, Tianrui Li, Daxin Jiang, and Nan Duan. 2020. GRACE: Gradient harmonized and cascaded labeling for aspect-based sentiment analysis. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 54–64, Online. Association for Computational Linguistics.

Sebastian Ruder. 2018. An overview of multi-task learning for deep learning.

Xingwei Tan, Yi Cai, and Changxi Zhu. 2019. Recognizing conflict opinions in aspect-level sentiment classification with dual attention networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3426–3431, Hong Kong, China. Association for Computational Linguistics.


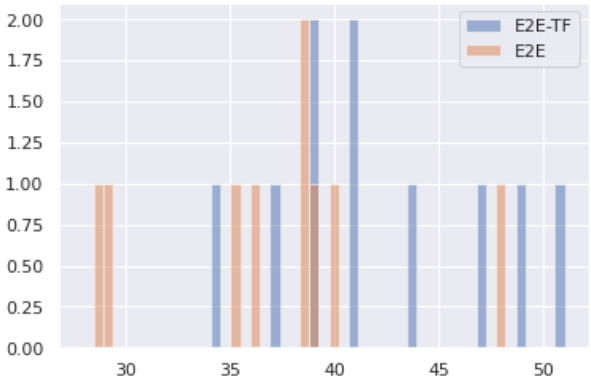


Figure 6: Histograms of F1 score of the conflict sentiment of E2E and E2E-TF models. We trained the models 10 times with different seeds and took the best F1 for conflict in each try. Mean and standard deviation are shown in Table. 9

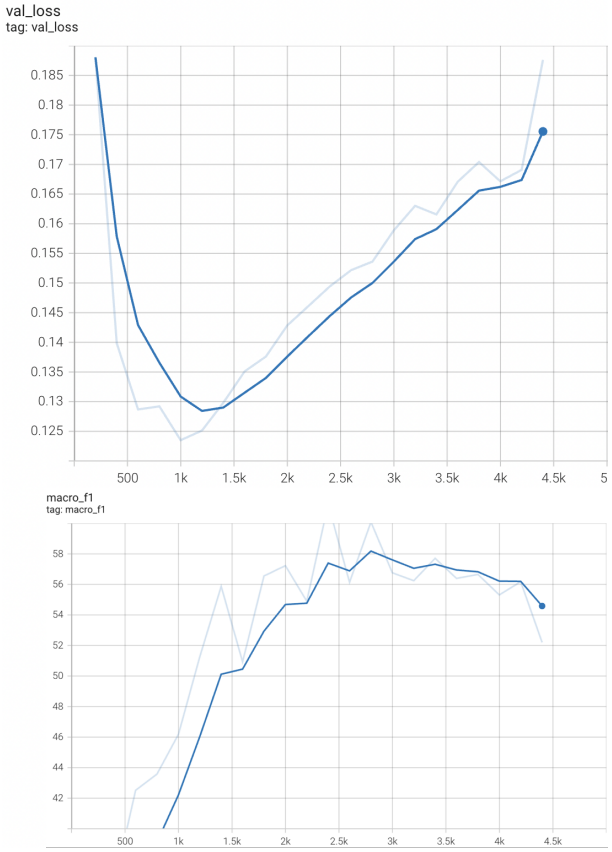


Figure 7: Comparison of validation loss (top) and macro F1 metrics during training. We can observe that in the region [1.2k, 2.7k] validation loss is worsening but macro F1 is improving.