randomseed19 at SemEval-2020 Task 10: Emphasis Selection For Written Text in Visual Media

Aleksandr Shatilov **Denis Gordeev** **Alexey Rey**
The Russian Presidential Academy of National Economy and Public Administration (RANEPA), Moscow, Russia
{shatilov-aa,gordeev-di,rey-ai}@ranepa.ru

Abstract

This paper describes our approach to emphasis selection for written text in visual media as a solution for SemEval 2020 Task 10. We used an ensemble of several different Transformer-based models and cast the task as a sequence labeling problem with two tags: 'I' as 'emphasized' and 'O' as 'non-emphasized' for each token in the text. Our approach ranked fourth in this competition.

1 Introduction

The purpose of SemEval 2020 Task 10 is to design automatic methods for emphasis selection, i.e. choosing candidates for emphasis in short written texts or enabling automated design assistance in authoring. The given task is different from related ones in that word emphasis patterns are person- and domain-specific, making different selections valid depending on the audience and the intent. Examples of different emphases are presented in Figure 1.

The dataset for this shared task includes short sentences from the following two datasets:

1. Spark dataset: This dataset is collected from Adobe Spark and is a collection of short texts containing a variety of subjects featured in flyers, posters, advertisements or motivational memes on social media.

2. Quotes dataset: This dataset is a collection of quotes from well-known authors collected from Wisdom Quotes.

The data was labeled by 9 annotators, for further details please refer to the task paper (Shirani et al., 2020).

Figure 1: Two examples with different emphasis

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 1685–1690
Barcelona, Spain (Online), December 12, 2020.

2 System overview

2.1 Dataset

Each token in the dataset is labeled using the BIO notation. The main target of the task is to predict emphasis probability for each token in a given sentence, which is calculated as a share of 'B' and 'I' labels from 9 annotators. An example of training and validation data is presented in Table 1. In the test data only columns 'word_id' and 'word' are available.

word_id	word	BIO_annotations	BIO_frequencies	emphasis_probability	pos_tags
Q_2_0	What	O|O|O|O|O|O|O|O|O	0|0|9	0.0	WP
Q_2_1	's	O|O|O|O|O|O|O|O|O	0|0|9	0.0	VBZ
Q_2_2	dangerous	O|O|B|B|O|O|B|B|B	5|0|4	0.5556	JJ
Q_2_3	is	O|O|O|O|O|O|O|O|O	0|0|9	0.0	VBZ
Q_2_4	not	O|B|B|O|O|O|O|O|O	2|0|7	0.2222	RB
Q_2_5	to	O|I|I|O|O|O|O|O|O	0|2|7	0.2222	TO
Q_2_6	evolve	B|I|I|O|B|B|O|O|O	3|2|4	0.5556	VB
Q_2_7	.	O|O|O|O|I|O|O|O|O	0|1|8	0.1111	.

Table 1: Example of training and validation data.

We approach the task as a sequence labeling problem with two tags: 'I' as 'emphasized' (the original 'B' was transformed to 'I') and 'O' as 'non-emphasized' for each token in the text. Subsequently, the probability of predicted 'I' token was taken as emphasis probability.

We considered two ways to convert the given dataset to a sequence labeling dataset:

1. Separate annotations for each sentence (9 examples for each given sentence)
2. Major vote annotation for each sentence (1 example for each given sentence), i.e. in Table 1 all annotations for token 'evolve' B|I|I|O|B|B|O|O|O were converted to 'I' (emphasized) - 5 'B' and 'I' vs. 4 'O'

Models were trained on these two types of dataset representation (separate annotations or major vote annotation).

2.2 Models

The main approach is quite simple: a pretrained Transformer-based model (Vaswani et al., 2017) with a token classification head on top (a linear layer on top of the hidden-states output).

Given a sequence of tokens, the model is to label each token with its appropriate class ('I' or 'O'). Emphasis probability for each token was calculated as a softmax over logits for a given token (the value of label 'I' was taken as a result). For words that were divided into several tokens, the result was considered to be the average of its parts. The model architecture is shown in Figure 2.

3 types of models were used:

– BERT (Devlin et al., 2018)

– RoBERTa (Liu et al., 2019)

– XLNet (Yang et al., 2019)

2.3 Evaluation

Predictions for the competition were evaluated with $Match_m$ metric.

$Match_m$: For each instance X in the test set D_{test}, we select a set $S_m^{(x)}$ of $m \in (1...4)$ words with the top m probabilities according to the ground truth. Analogously, we select a prediction set $\hat{S_m}^{(x)}$ for each m, based on the predicted probabilities.

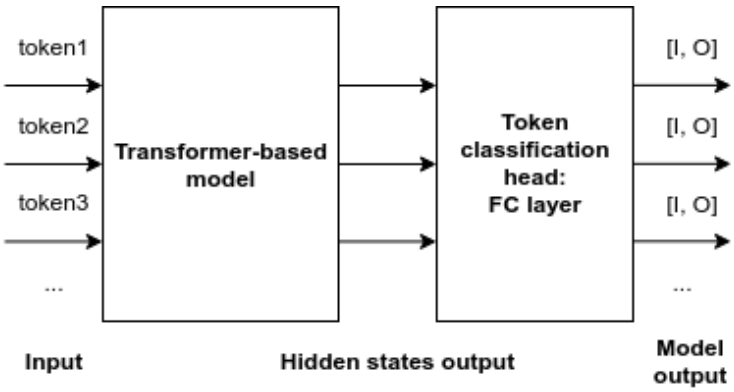


Figure 2: Model architecture

$Match_m$ is defined as follows:

$$Match_m = \frac{\sum_{x \in D_{test}} \left| S_m^{(x)} \bigcap \hat{S}_m^{(x)} \right| / m}{|D_{test}|}$$

The final evaluation metric is the mean of $Match_1$, $Match_2$, $Match_3$, $Match_4$

3 Experimental setup

We used the dataset split scheme provided by the organizers, the data contained 3,877 sentences with 70% training, 10% validation and 20% test sets. All models were trained only on the given train part of the dataset.

The technology stack of frameworks and libraries we used [1]:

– pytorch[2] (Paszke et al., 2019) - as the main framework

– huggingface transformers[3] (Wolf et al., 2019) - we used model architectures, pretrained models and additional functions

– Pytorch Lightning[4] (Falcon, 2019) - as a wrapper for organizing model training code

During training we tried all combinations of the following hyperparameters for each model type:

– Dataset representation (label mode): separate, major vote, described in 2.1

– Batch size[5]: 16, 32, 64, 128, 256

– Learning rate: 1e-05, 5e-05

For each set of hyperparameters the model was trained for up to 10 epochs with 4 validations per epoch. We used linear gradient scheduling with warm-up (fraction of warmup steps was set to 0.05 for all models) and AdamW optimizer (Loshchilov and Hutter, 2017).

For each model we chose two best sets of hyperparameters by:

– loss - Cross Entropy Loss of sequence labeling task

– score - described in 2.3

[1]The implementation is available online: `https://github.com/InstituteForIndustrialEconomics/semeval2020_task10`

[2]ver. 1.4.0, `https://pytorch.org/`

[3]ver. 2.5.1, `https://github.com/huggingface/transformers`

[4]ver. 0.7.1, `https://github.com/PytorchLightning/pytorch-lightning`

[5]to fit big batches on 1080 GPU gradient accumulation was used

4 Results

4.1 Test results

For the test data we submitted results of 4 models:

- Best single model based on validation loss
- Ensemble of best single models of each type based on validation loss
- Best single model based on validation score
- Ensemble of best single models of each type based on validation score

The ensemble is just an average prediction of the models that compose it.

The best test score was achieved by an ensemble of single models chosen by the validation score. The ensemble and its components parameters and results are presented in Table 2.

Model type	Label mode	Batch size	Learning rate	Validation score	Test score
roberta-large	separate	32	0.00001	0.79933	0.79769
xlnet-large-cased	separate	16	0.00001	0.79893	-
bert-large-cased	separate	32	0.00005	0.79296	-
Ensemble	separate	-	-	0.80848	0.80486

Table 2: Parameters and scores of best models selected by validation score

The ensemble of single models chosen by validation loss, its components parameters and results are presented in Table 3.

Model type	Label mode	Batch size	Learning rate	Validation score	Test score
roberta-large	major	256	0.00005	0.77984	0.77035
bert-large-cased	major	64	0.00005	0.75830	-
xlnet-large-cased	major	16	0.00005	0.76718	-
Ensemble	major	-	-	0.78893	0.78916

Table 3: Parameters and scores of best models selected by validation loss

As can be seen from the tables above, the models trained on separate annotations showed better scores than the models trained on major vote annotations.

A part of the competition leaderboard for the evaluation phase is presented in Table 4. Our model took 4th place. [6]

Column 'Mean Score' is the column with final evaluation metric - mean of $Match_n, n \in (1...4)$. In parentheses is the place of the model for each evaluation metric.

Our model showed good results for $Match_3$ and $Match_4$ evaluation metrics (2nd place) and worse for $Match_1$ and $Match_2$ (8th and 4th place). It means the model doesn't predict one or two most emphasized tokens very well, but works satisfactorily at predicting three and four.

#	Team name	Mean Score	$Match_1$	$Match_2$	$Match_3$	$Match_4$
1	ERNIE	0.823 (1)	0.724 (1)	0.819 (1)	0.862 (1)	0.887 (1)
4	randomseed19	0.805 (4)	0.677 (8)	0.803 (4)	0.858 (2)	0.881 (2)
26	teamTest - baseline	0.750 (26)	0.608 (21)	0.737 (23)	0.807 (24)	0.849 (23)

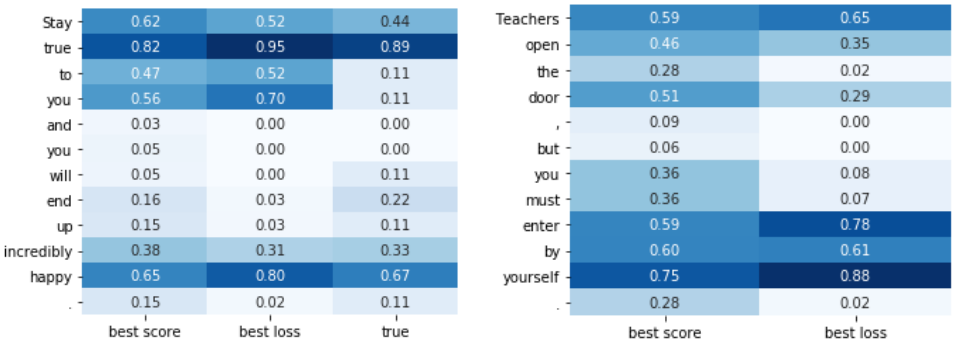
Table 4: Part of the leaderboard

[6] Link to leaderboard: `https://competitions.codalab.org/competitions/20815#results`

The model performed better than the baseline DL-BiLSTM+ELMo model provided by organizers (Shirani et al., 2019).

4.2 Prediction examples

Below are examples of prediction for two types of model checkpoints - chosen by the best validation score and the best validation loss. Models selected by validation loss predict more extreme values (zeros for unlikely emphasized tokens and higher values for likely emphasized tokens) compared to models selected by validation score.



(a) Validation dataset

(b) Test dataset

Figure 3: Examples of predictions

4.3 Future Improvements

We didn't consider the following things that might improve the model:

- Other transformer-based models, i.e. ERNIE (Sun et al., 2019)
- Sentence source: Quotes dataset or Spark dataset
- Part of speech tags for tokens, can be obtained using one of the existing models and libraries: nltk, spacy, etc.

5 Conclusion

In this paper we presented the system we used in SemEval-2020 Emphasis Selection For Written Text in Visual Media competition. The proposed approach is based on Transformer-based models and sequence labeling task.

The final model we used was an ensemble of 3 models (BERT, RoBERTa, XLNet) each trained with best hyperparameters according to validation scores. The model showed good results and took 4th place.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

W.A. et al. Falcon. 2019. Pytorch lightning. `https://github.com/PytorchLightning/pytorch-lightning`.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Amirreza Shirani, Franck Dernoncourt, Paul Asente, Nedim Lipka, Seokhwan Kim, Jose Echevarria, and Thamar Solorio. 2019. Learning emphasis selection for written text in visual media from crowd-sourced label distributions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1167–1172, Florence, Italy, July. Association for Computational Linguistics.

Amirreza Shirani, Franck Dernoncourt, Nedim Lipka, Paul Asente, Jose Echevarria, and Thamar Solorio. 2020. Semeval-2020 task 10: Emphasis selection for written text in visual media. In *Proceedings of the 14th International Workshop on Semantic Evaluation*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2019. Ernie 2.0: A continual pre-training framework for language understanding.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding.