Tëxtmarkers at SemEval-2020 Task 10: Emphasis Selection with Agreement Dependent Crowd Layers

Kevin Glocker and **Stefanos Andreas Markianos Wright**
Department of Linguistics
University of Tübingen
{kevin.glocker,stefanos-andreas.markianos-wright}@student.uni-tuebingen.de

Abstract

In visual communication, the ability of a short piece of text to catch someone's eye in a single glance or from a distance is of paramount importance. In our approach to the SemEval-2020 task "Emphasis Selection For Written Text in Visual Media", we use contextualized word representations from a pretrained model of the state-of-the-art BERT architecture together with a stacked bidirectional GRU network to predict token-level emphasis probabilities. For tackling low inter-annotator agreement in the dataset, we attempt to model multiple annotators jointly by introducing initialization with agreement dependent noise to a crowd layer architecture. We found our approach to both perform substantially better than initialization with identities for this purpose and to outperform a baseline trained with token level majority voting. Our submission system reaches substantially higher Match_m on the development set than the task baseline (0.779), but only slightly outperforms the test set baseline (0.754) using a three model ensemble.

1 Introduction

Emphasis selection is the task of choosing individual words or phrases of short written texts to emphasize. While emphasis selection has also been used in previous work for more natural articulation in text-to-speech systems (Mass et al., 2018), in the SemEval shared task emphasis annotations are intended for use in visual communication such as in posters or advertisements (Shirani et al., 2020).

Word emphasis helps readers conveniently scan for important information in a text. Capital letters, bold letters, italics and underlines can help emphasize certain facts and drive the message home; however, overuse of these features can lead to confusion and create an unattractive aesthetic. Usually advertisers focus on one or two key words or phrases that will pique the readers' interest (Shirani et al., 2019).

As with many tasks based on sentence modeling, substantial improvements have been gained in emphasis selection by leveraging recent breakthroughs in language modeling such as the ELMo architecture (Peters et al., 2018; Shirani et al., 2019). Following this trend, our submission uses contextualized word representations from a pretrained model of the state-of-the-art BERT architecture (Devlin et al., 2019) together with a stacked bidirectional GRU network for learning task specific information.

A major challenge when developing a system for automatic emphasis selection are very low agreements between annotators about which parts of a sentence should be emphasized which impacts how models are trained and evaluated (Mass et al., 2018; Shirani et al., 2019). To tackle annotator subjectivity, both majority voting (Mass et al., 2018) and regression with a KL divergence objective (Shirani et al., 2019) have been used for training in previous work. In our approach, we apply a crowd layer architecture adapted from Rodrigues and Pereira (2018) and improve its effectiveness for the task by introducing agreement dependent noise during initialization.

2 Data

A combination of two English datasets was provided for this task (Shirani et al., 2020). In total, the training and development sets comprise of 3,106 short text instances (sentences).

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 1698–1703
Barcelona, Spain (Online), December 12, 2020.

Both datasets consist of tokenized sentences with annotations by nine different annotators for each token using an inside–outside–beginning (IOB) tagging scheme. This is a scheme whereby tokens are tagged as either I when inside an emphasized chunk, B when at the beginning of a chunk or O when outside of a chunk (Ramshaw and Marcus, 1995). The frequencies of the IOB tags are then counted and the sum of the *B* and *I* tags is divided by the number of annotators for each token. This then indicates the emphasis probability which is used for evaluation (Shirani et al., 2019; Shirani et al., 2020). The tokens with the highest probability score are more likely to be the appropriate candidates for emphasis.

The first dataset is from Adobe Spark, and contains 960 short text instances from flyers, posters, advertisements or motivational memes on social media. It contains 5,940 tokens and the average number of tokens per instance is 6.18, ranging from 1 to 27 tokens. The average emphasis probability per token is 0.38. The second dataset is a collection of quotes from well-known authors collected from Wisdom Quotes containing 2,174 short texts. It contains 30,845 tokens and the average number of tokens per instance is 14.18, ranging from 3 to 38 tokens. The average emphasis probability per token is 0.26.

The Light's Kappa (Light and Smith, 1971) between all raters in the task training set is 0.23. In comparison, the kappa agreement between the four professional labelers in the speech emphasis dataset used by Mass et al. (2018) for an expressive text-to-speech system is only slightly higher at 0.35. This suggests that emphasis annotations are very subjective and that this is a common issue across domains which is important to tackle in order to achieve good system performance.

3 System

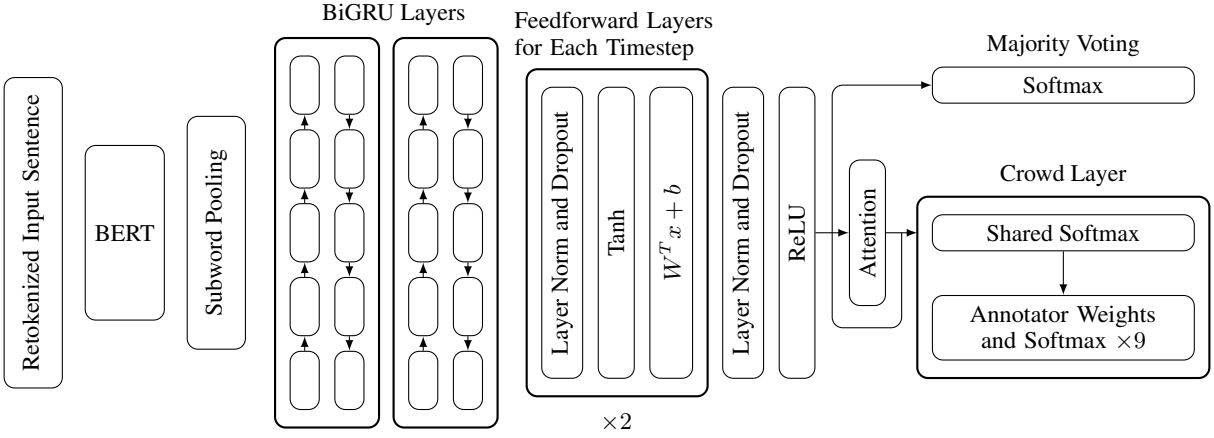

Figure 1: Schematic of our architecture with its three variants differing in the final layers: majority voting, crowd layer and an optional attention layer before the crowd layer. In the variants using a crowd layer, the output of the shared softmax is used directly at inference time.

Our neural network architectures make use of state-of-the-art language modeling of the input sentences by using pretrained BERT embeddings (Devlin et al., 2019) followed by a bi-directional GRU layer (Cho et al., 2014) with a feedforward network for each time step for modeling task specific information. The architecture and its variants are illustrated in Figure 1.

Since the BERT model we use was trained on WordPiece tokens (Devlin et al., 2019; Wu et al., 2016), token boundaries in the pre-tokenized training data are very different in some cases. While token boundaries are generally similar for highly frequent words, WordPiece splits up words which are rare or were not present in the training data into smaller subword units (Wu et al., 2016). To accommodate for this, we construct word vectors for the original tokens in the data by further tokenizing each token separately into its subword units following the same WordPiece segmentation rules as in the pretrained model. The subword units for each of the original tokens are then concatenated to form the entire sentence and passed into the BERT embedding model where contextualized word representations are extracted from the final hidden states. Finally, we compute a single vector representation for the original tokens

by applying max pooling over the subword vectors which we found to work slightly better than mean pooling in preliminary experiments.

For modeling task specific information we then feed the resulting vectors into a component-level dropout layer (Srivastava et al., 2014) for regularization after which they are passed into a bidirectional RNN layer. While LSTMs are the preferred RNN cells in previous work (Rodrigues and Pereira, 2018; Shirani et al., 2019), we achieved slightly higher scores using GRU cells instead which also have the added benefit of being faster to train due to their simplified gating mechanism (Cho et al., 2014). We use a stack of two bidirectional GRU (BiGRU) layers, apply dropout between layers as regularization and use learned matrices as the forward and backward initial states instead of using zero states for minor performance improvements.

The forward and backward hidden states of the last GRU layer are then concatenated into single vectors for each token in a sentence and passed into a feedforward network. Our feedforward layers are pre-activated by the tanh function. For regularization, all feedforward layers are preceded by layer normalization followed by dropout inspired by IC-layers (Chen et al., 2019). The final feedforward layers are additionally followed by another pair of layer normalization and dropout and a ReLU activation.

Similarly to Shirani et al. (2019), we also tested an architectural variant which includes an attention layer over final feedforward layer outputs where scores are computed via Bahdanau-style attention (Bahdanau et al., 2014) between each output vector and the mean of all output vectors in the sentence:

$$\alpha_i = \text{softmax}\left(v^T \tanh\left(W f_i + U \frac{1}{N} \sum_{j=0}^{N} f_j\right)\right) \tag{1}$$

$$c_i = f_i \alpha_i \tag{2}$$

Here, f_i is the output of the feedforward network at time step i, N is the number of words in a sentence, W and U are learned weights of single vectors and the mean of all vectors in the sentence respectively, and c_i is the product of an output vector f_i with its corresponding attention weight α_i.

We primarily experimented with two techniques for dealing with labels from different annotators. As a baseline, we used a single softmax activated projection layer trained on labels obtained via token level majority voting and optimized with a cross-entropy objective. In our second approach, we attempt to model consensus between annotators in the dataset explicitly in a shared representation by parameterizing the annotators using a crowd layer (Rodrigues and Pereira, 2018).

3.1 Crowd Layer

Our version of the crowd layer is based on the work of Rodrigues and Pereira (2018) who introduce learned weight matrices for each annotator applied to a shared softmax-activated projection layer to model reliability and biases in the context of learning from crowdsourced datasets.

Our version of the crowd layer uses a shared projection layer activated with a log softmax for numerical stability which is replaced by a regular softmax at inference time for retrieving emphasis probabilities. During training, we add a weight matrix for each annotator which transforms the shared log probabilities to account for e.g. biases before applying a second softmax to turn the weighted probabilities into valid distributions. We then use a cross-entropy objective for training the annotator level projection layers on the raw IOB labels in the training set.

The primary difference between our version and the original crowd layer is our initialization of the annotator weight matrices. Rodrigues and Pereira (2018) found that in the context of crowdsourced datasets, initializing weights to identity matrices worked best. The very low agreement between annotators in the task data indicates that cases where transformations of the shared probabilities are ideally close to identities for most or all annotators might be much less likely. Motivated by this assumption, we experimented with adding normal noise to identity matrices for initialization instead. Empirically, we found that the best performance can be achieved by using a mean of 0 and a standard deviation set to $1 - \kappa$ where κ indicates the average Light's Kappa (Light and Smith, 1971) between annotators in the training set. As a result, initialization is close to the original identity initialization in datasets with high agreement

but closer to the standard normal with low agreement as is the case in the task dataset. Conceptually, this can be interpreted as making the assumption that more biases or noise have to be accounted for with increasing disagreement in the data for which noisy initializations are better starting points.

4 Evaluation

Our models were implemented in Python with the PyTorch framework[1] (Paszke et al., 2019). We used the pre-trained cased BERT model ("Bert-cased") from Devlin et al. (2019) as implemented in Python in the "transformers" library[2] (Wolf et al., 2019) with which we compute 768-dimensional contextualized word embeddings together with our pooling approach described in Section 3. In all models we used two BiGRU layers with 250 neurons each and 2 feedforward layers with 150 neurons each.

For the crowd layer we tested three different initializations of the annotator weights which vary by the standard deviation (σ) of normal noise added to identity matrices to investigate the effectiveness of our approach. We trained models using identities without noise ($\sigma = 0$) as in Rodrigues and Pereira (2018), a standard normal distribution ($\sigma = 1$) and our agreement dependent approach ($\sigma = 0.77$).

For regularization, we used dropout rates of 0.1 on the input and 0.2 for all other dropout layers between biGRU and feedforward layers. Our networks were trained using the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.001 and training was stopped after two epochs of no improvement on development set loss.

In addition to single models, we also evaluated some simple ensembles of different architectural variants. Scores from ensembles were obtained by averaging output probabilities of the constituent models for each token. We compare scores using the Match$_m$ metric with $m \in \{1 \dots 4\}$ which was used as the evaluation metric for the shared task (Shirani et al., 2019; Shirani et al., 2020).

5 Results

Model	Match$_m$ scores									
	Development Set					Test Set				
	1	2	3	4	Average	1	2	3	4	Average
Task Baseline (Shirani et al., 2019)	0.59	0.75	0.8	0.82	0.7424	**0.61**	0.74	0.81	0.85	0.75
BERT Models										
Majority Voting	0.63	0.75	0.8	0.83	0.7545	0.59	0.72	0.79	0.84	0.7352
Crowd Layer ($\sigma = 0$)	0.57	0.73	0.81	0.84	0.7385	0.55	0.72	0.8	0.84	0.7262
Crowd Layer ($\sigma = 1$)	0.56	0.72	0.79	0.82	0.7233	0.54	0.71	0.8	0.83	0.7192
Crowd Layer ($\sigma = 0.77$)	0.64	0.77	0.81	0.84	0.7652	0.57	0.75	0.81	0.84	0.7429
Crowd Layer + Attention	0.61	0.76	0.82	0.84	0.7584	0.59	0.74	0.81	0.85	0.7459
BERT Ensembles										
Majority, Crowd	0.65	0.77	0.81	**0.85**	0.7724	0.6	0.74	0.8	0.85	0.748
Majority, Crowd, Attention	**0.67**	**0.78**	**0.83**	0.84	**0.7786**	0.6	**0.75**	**0.81**	**0.85**	**0.7535**

Table 1: Evaluation results for our single models and ensembles on the development and test sets using the Match$_m$ metric. The highest scores in each column are marked in bold and the name of our submission model in italics.

Table 1 shows Match$_m$ for all values of m as well as their averages on the development and the test splits of the task dataset. The model using our version of the Crowd Layer with agreement dependent initialization ($\sigma = 0.77$) and without the attention mechanism substantially outperformed both the task baseline and our majority voting model on the development set by a margin of ~0.01. While still

[1] https://github.com/pytorch/pytorch/releases/tag/v1.4.0

[2] https://github.com/huggingface/transformers/releases/tag/v2.5.1

outperforming both baselines on the development set, adding an attention mechanism to our architecture decreased performance compared to our Crowd model without attention. However, on the test set it slightly improved performance making it our highest scoring single model in this context while still reaching a slightly lower score than the baseline by a margin of ~0.0041.

Both identity and standard normal initializations of the crowd layer reach much lower scores than agreement based initialization and are even outperformed by our majority voting baseline. This highlights the importance of selecting a good starting point when training models with this architecture. We found that improvements over identity initialization could be gained with standard deviations until $\sigma = 0.77$ which corresponds to the disagreement between annotators. Even more noisy initializations decreased performance rapidly below that of identity initialization.

Ensembling our Crowd and Majority models lead to a substantially higher score both on the development and test set. With the further addition of the Attention model we gained another comparatively large score improvement on both subsets. While most of our models reach higher scores than the baseline on the development set, only one of our ensemble models outperforms the test set by a much smaller margin leading to our model ranking 25th in the shared task (Shirani et al., 2020).

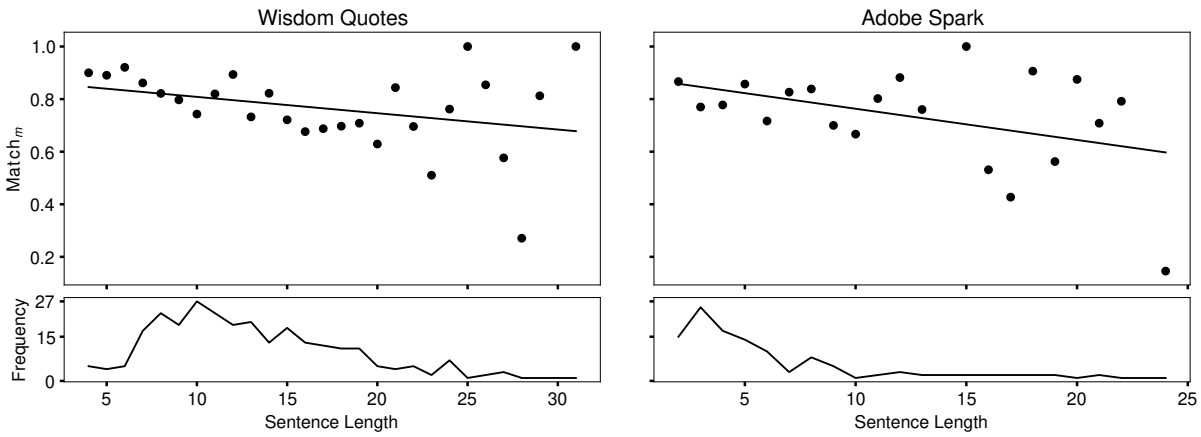

Figure 2: Medians of average Match$_m$ scores for sentences of the same lengths in both subsets of the validation set with linear regression lines. In the bottom row the corresponding sentence length frequency distributions are shown for reference.

We further analyzed performance of our ensemble model on both subsets of the development set and found that our model scores considerably better on the Adobe Spark subset (0.7842) than on the Wisdom Quotes subset (0.7755). Investigating scores of sentences of different lengths for both subsets (Figure 2), we found that while scores and sentence lengths are not very highly correlated in both the Adobe Spark ($R^2 = 0.13$) and Wisdom Quotes ($R^2 = 0.28$) subsets in part due to high variance where fewer long sentences are available, there is a slight overall trend of sentences becoming considerably more difficult to model for our system with increasing length. Since Adobe Spark primarily consists of very short phrases with 70% containing fewer than eight words this is a possible explanation of the better scores of our model on the subset. Further analysis of other factors which potentially lead to significant deviations from the gold standard emphasis probabilities such as different syntactic constructions and part-of-speech tags didn't result in any significant patterns.

6 Conclusion

We presented a GRU-based recurrent architecture for emphasis selection using state-of-the-art language modeling with BERT and introduced agreement dependent noise initialization to adapt a crowd layer to modeling annotators for a dataset with very low inter-annotator agreement which we showed to outperform initializations with only identities or addition of standard normal noise. While our architecture outperformed the task baseline substantially on the development set only one of our ensemble systems

outperforms it on the test set by a small margin.

Acknowledgements

We would like to thank Çağrı Çöltekin for his advice and support with completing this task.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Guangyong Chen, Pengfei Chen, Yujun Shi, Chang-Yu Hsieh, Benben Liao, and Shengyu Zhang. 2019. Rethinking the usage of batch normalization and dropout in the training of deep neural networks. *arXiv preprint arXiv:1905.05928*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

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

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Richard Light and Paul Smith. 1971. Accumulating evidence: Procedures for resolving contradictions among different research studies. *Harvard educational review*, 41(4):429–471.

Yosi Mass, Slava Shechtman, Moran Mordechay, Ron Hoory, Oren Sar Shalom, Guy Lev, and David Konopnicki. 2018. Word emphasis prediction for expressive text to speech. In *INTERSPEECH*.

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 *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Lance A. Ramshaw and Mitchell P. Marcus. 1995. Text chunking using transformation-based learning. *CoRR*, cmp-lg/9505040.

Filipe Rodrigues and Francisco C. Pereira. 2018. Deep learning from crowds. In *AAAI*.

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.

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*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

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.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.