LAST at SemEval-2020 Task 10: Finding tokens to emphasise in short written texts with precomputed embedding models and LightGBM

Yves Bestgen
Laboratoire d'analyse statistique des textes - LAST
Institut de recherche en sciences psychologiques
Université catholique de Louvain
Place Cardinal Mercier, 10 1348 Louvain-la-Neuve, Belgium
yves.bestgen@uclouvain.be

Abstract

To select tokens to be emphasised in short texts, a system mainly based on precomputed embedding models, such as BERT and ELMo, and LightGBM is proposed. Its performance is low. Additional analyzes suggest that its effectiveness is poor at predicting the highest emphasis scores while they are the most important for the challenge and that it is very sensitive to the specific instances provided during learning.

1 Introduction

This paper reports on the participation of the Laboratoire d'analyse statistique des textes (LAST) in the SemEval-2020 Task 10 *Emphasis Selection For Written Text in Visual Media*. This task requires automatic systems to select tokens[1] for emphasis in English famous quotes (e.g., *No matter how hard you work, someone else is working harder.*[2]) and other short written text such the ones used in flyers, advertisements or social media posts (e.g., *stay positive*). Its importance is growing continuously given the explosion in the number of slogans that web users want to display on smaller and smaller screens of more and more devices. Writing brief, attention-grabbing and engaging message has become paramount. One of the characteristics of this task which makes it particularly complex resides in the brevity of the texts which means that almost all words, with the notable exception of grammatical words, are important otherwise they would have been omitted. So, it is more about identifying the most important words on a scale than making a binary decision between important vs. not important.

This task has not attracted attention until last year when Shirani *et al.* (2019) simultaneously highlighted its interest, collected from Adobe Spark[3] a dataset annotated by human judges, and proposed an automatic system for performing it. This system uses a deep sequence labeling network composed of two stacked bidirectional LSTM layers and relies on ELMo embeddings. Shirani *et al.* (2019) observed that it outperformed a Conditional Random Fields model (Lafferty *et al.*, 2001) with hand-crafted features. However, they did not evaluate other classical supervised learning techniques, such as gradient boosting decision tree (Friedman, 2000), or the combination of these techniques with precomputed embedding models. The main objective of this study is to try to determine whether the use of such procedures can be competitive for the emphasis selection task.

The remainder of this report describes the datasets made available for this challenge and the evaluation criteria, the systems developed, and the results obtained as well as a few analyses performed to get a better idea of the factors that affect the system performance.

[1] The term *token* is preferred to *word* because the highlighted elements can also be numbers and symbols (e.g.,*$ 35* or) and even punctuation marks (e.g., *!* or *;*).

[2] A quote that applies perfectly to this paper.

[3] spark.adobe.com, an application specialized in automated design assistance in authoring.

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 1671–1677
Barcelona, Spain (Online), December 12, 2020.

2 Data and Evaluation Settings

The materials for this task consisted of 3,876 brief pieces of text (or *items*) from 2 to 38 words [4]. The challenge organizers recruited raters, through Amazon Mechanical Turk, for annotating these items. For each item, nine of them had to decide whether each token should be emphasized, a difficult decision for human judges since Shirani *et al.* (2019) reported an agreement (Fleiss' kappa) below 0.65. These items were randomly split up into three sets (see Shirani *et al.* (2020) for details): the training set (Train: 2,741 items and 32,399 tokens), the development set (Dev: 392 items and 4,385 tokens) and the test set (Test: 743 items and 8,192 tokens). The true labels for the Train and Dev sets were available from the beginning of the training phase while those for the Test set still remain hidden.

Systems proposed to this shared task had to submit for each token of each item a predicted score of emphasis. On the basis of Shirani *et al.* (2019), the task organizers decided to use for assessing performances an ad hoc measure called *Match_m*, roughly[5] defined in Shirani *et al.* (2019), but implemented in a python function. It assesses the accuracy with which the tokens with the highest emphasis scores according to the submission are also those with the highest emphasis scores in the gold standard. This calculation is carried out successively for the first, first two, first three and first four tokens with the highest scores in the gold standard. The final *Match_m* for a system is the unweighted average of the average of the four *Match_m* scores over all of the items. Being an average of accuracy scores, its maximum value is 1. The minimum value for a given item depends on both its length in tokens and the number of ties among the highest scores[6]. An important property of this evaluation measure is that it is calculated by item on the ranked data. It follows that it treats in the same way two tokens ranked first in their respective item, but one selected by all the annotators and the other by less than half while it can be considered that identifying the former is more important.

3 System Overview

The proposed approach treats the prediction of the token emphasis scores as a regression problem and uses LightGBM (Ke *et al.*, 2017) to perform this prediction. The first analyzes carried out indicated that using classical features in text categorization such as token, lemma and POStag n-grams were far from achieving an effectiveness comparable to that of the approach proposed by Shirani *et al.* (2019), which was chosen by the task organizers as the baseline. One potential explanation is that the majority of tokens in the materials appear only once. To take this difficulty into account, I chose to represent each token by means of precomputed embedding models and to use the corresponding vectors as features for LightGBM. It was expected that the embeddings will make rare tokens similar to more frequent ones in the training materials. The first analyzes carried out with this approach showed that adding to these embedding features the n-grams features tested first was not useful. So, I decided to focus on the embeddings. However, a limitation of the approach is that no information is extracted from the context (i.e., the whole item). An extended version of the system was therefore developed. It was based on the emphasis score predicted by the base system to which contextual features were added, some of which being obtained after processing the items by means of the Stanford CoreNLP (Manning *et al.*, 2014).

3.1 Implementation Details

3.1.1 Procedure to Build the Systems

With the exception of obtaining the embeddings themselves and the Stanford CoreNLP, all the processing steps were performed by means of a series of custom SAS programs running in SAS University (freely available for research at www.sas.com/en_us/software/university-edition.html). All the predictive models

[4] All the numbers given in the paper do not take into account an item composed of a single word (i.e., *S546: ADVENTURE*), which cannot be used in the results because the challenge measure is not calculated for an item made up of a single word.

[5] For example, this definition does not state how equally-ranked tokens are treated, both in the gold standard and in the solution submitted, or that the maximum number of highest emphasis scores taken into account in a item is strictly less than its number of tokens.

[6] If the minimum value is usually zero, it is, for instance, 0.27 for an item of 5 words without a tie and 1 for an item whose all tokens have the same emphasis score in the gold standard, a case that occurs 30 times in the combined Train and Dev sets.

were built by means of LightGBM. For developing the systems and fine-tuning the parameters, a 7-fold cross-validation procedure (CV) based on the combined Train and Dev sets was used.

3.1.2 Pretrained Embedding Models and Features

Two pretrained embedding models were used. The first was the 24 times 1024-dim uncased BERT embeddings (Devlin *et al.*, 2019), more precisely wwm_uncased_L-24_H-1024_A-16, obtained by means of *bert-as-service* (Xiao, 2018). The tests carried out with the pretokenized version having shown that a large proportion of the tokens are unknow of BERT, I led BERT tokenized the items, adding the embeddings of several BERT tokens when necessary (e.g., *un + ##lea + ##rn + ##ing* to get the layers for *unlearning*). The second was the pretrained ELMo model (Peters *et al.*, 2018) applied without learning and item by item to the tokens provided in the original materials, giving rise to the 1024-dim ELMo embeddings.

The analyzes carried out during system development led to first selecting the 22nd layer of BERT (B22) to which were added the 1024 vectors of ELMo, the 12th layer of BERT (B12) and 1024 features obtained by adding the values of each vector in layer 22 of the preceding token and the target token. A binary feature was used to indicate whether the BERT layers were computed by adding several BERT tokens, that is, as explained just above, when it was necessary to add the embeddings of several BERT tokens to get the embeddings for an original token.

3.1.3 Contextual Features

Two sets of contextual features were used. The first set (called LR) was obtained by encoding for each item all lemmas, POStag, dependency governor and entity code from the Sanford CoreNLP of all the tokens to the left and to right of the target token with a weight proportional to the distance between the target token and each of these tokens.

The second set of contextual features (called REP) has its origin in the fact that a significant number of items contain repeated lemmas and that this seems to affect the position of the emphasis. For example, in the item *Positive mind. Positive vibes. Positive life.*, the emphasis seems to increase with repetitions. This type of information has been encoded by applying the following procedure to items in which at least one lemma is repeated:

```
For each repeated lemma pairs
 for each token in the item
  the repeated lemma with the following cases is one-hot encoded:
   Is it before the first repeated lemma?
   Is it the first repeated lemma?
   Is it between the first repeated lemma and the second one?
   Is it the second repeated lemma?
   Is it after the second repeated lemma?
```

3.1.4 LightGBM Parameters

For the base system, the LightGBM parameters have been left at their default values except the followings: *num_iterations: 6000, max_bin: 510, bagging_freq: 5, bagging_fraction: 0.38, boost_from_average:false, feature_fraction: 0.05, learning_rate: 0.0095, max_depth: 6, min_data_in_leaf: 40, num_leaves: 25.*

For the extended system, the same parameters were used except for: *feature_fraction: 0.09, max_depth: 9, min_data_in_leaf: 10, num_leaves: 27.*

4 Results

As a reminder, the base system uses only traits from precomputed embedding models and the extended system uses the prediction from the base system (one feature) to which contextual features are added. These two systems were submitted for the challenge using the combination of Train and Dev sets for learning (3,133 items and 36,784 tokens). The best of the two was the extended system, which achieved a mean *Match_m* of 0.756 (see the Full model in Table 2), just barely 0.003 more than the base system (see the Full model in Table 1). It ranked twenty-third out of 31 participants, very far from the best team,

	Train + Dev					Train				
Value of m	1	2	3	4	Mean	1	2	3	4	Mean
B22	.595	.735	.808	.849	.747	.598	.735	.809	.847	.747
B22 + ELMo	.602	.751	.808	.851	.753	.612	.749	.807	.853	.755
B22 + ELMo + B12	.612	.746	.813	.848	.755	.598	.746	.806	.849	.750
Full	.607	.748	.809	.850	.753	.608	.749	.807	.845	.752
Def. 6,000	.581	.720	.800	.846	.739	.565	.716	.793	.842	.729
Def. 100	.542	.713	.792	.842	.722	.545	.721	.794	.839	.725

Table 1: *Match_m* for various sets of features of the base system on the Test set.

	Train + Dev					Train				
Value of m	1	2	3	4	Mean	1	2	3	4	Mean
Pred. M1	.600	.747	.812	.852	.753	.599	.743	.810	.855	.752
Pred. M1 + LR	.604	.749	.809	.851	.753	.606	.751	.810	.851	.754
Pred. M1 + REP	.602	.747	.812	.854	.754	.606	.747	.811	.852	.754
Full	.610	.749	.812	.853	.756	.595	.748	.808	.852	.751

Table 2: *Match_m* for various sets of features of the extended system on the Test set.

which obtained a mean *Match_m* of 0.823, and with only 0.006 more than the task organizers' Baseline system[7].

In order to assess the usefulness of the different feature sets, an ablation procedure was used. It must be noted that all the compared models include the binary feature which codes whether the embeddings of several BERT tokens were added to get the embeddings for an original token. The values obtained on the official test set are given in Table 1 for the base system and in Table 2 for the extended system. At the request of the challenge organizers, these tables also present the performances obtained when the training was carried out only on the Train set. In Table 1, the performance of the full model, but using the default parameters of LightGBM are also given. There are two cases: all the default parameters including the number of iterations, which is then fixed at 100 (Def. 100) and the same model, but by fixing this number at 6,000 as in the other cases (Def. 6,000)[8].

Def. 100 < Def. 6,000 < B22 = B22_ELMo = B22_ELMo_B12 = FullM1 < FullM2

Figure 1: Statistically significant differences between the main models for the 7-fold CV

It is clearly the parameter optimization which improves the performance of the base system. The differences between the other conditions are very small and variable depending on which training set is used. Such small differences between the various versions of the systems raise the question whether they are not just the result of random fluctuations.

In order to determine if the observed differences were statistically significant, a Monte-Carlo permutation test for related samples (Howell, 2008, Chap. 18) was used to compare selected conditions at a threshold level of 0.01, using 1,000 random permutations. These analyzes were performed by means of

[7] For this reason, but also because the system is based on the SAS statistical software (BASE part), rarely used by researchers in the field, it was not considered useful to provide the code in a GitHub repository.

[8] These conditions have not been evaluated for the extended model because they rely on the output of the base model and it is not obvious to determine whether to use the version with the parameters optimized or not for this base model.

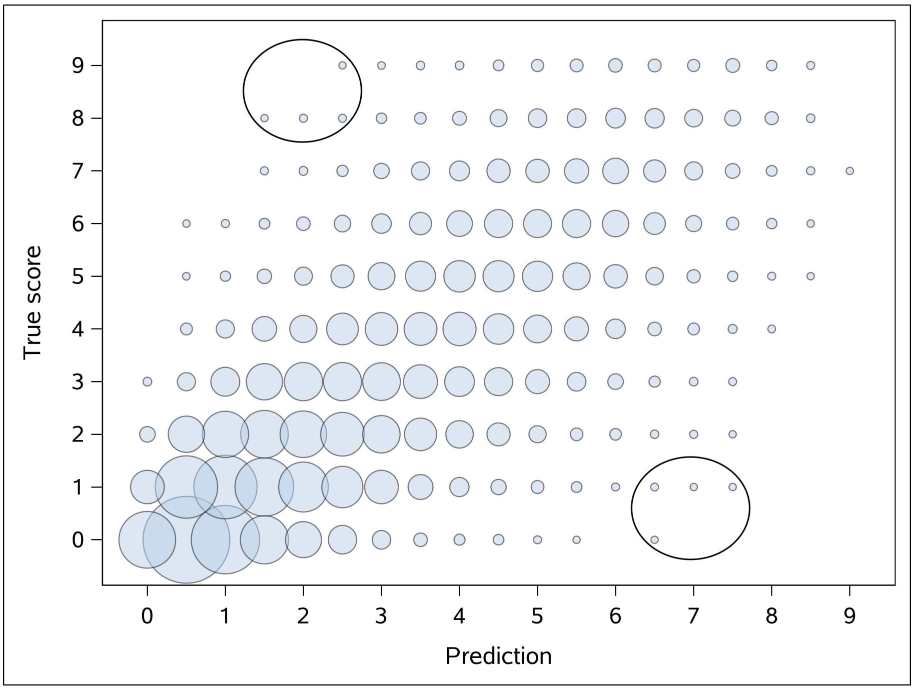

Figure 2: Bubble plot of true and predicted emphasis scores for the 7-fold CV

a cross-validation procedure since the gold standard for the test materials is not available. As Bestgen (2020) observed that performing a single k-fold cross-validation provided sufficient information to assess the reproducibility of observed differences, the 7-fold distribution used during the development of the system was employed. Figure 1 shows that the default parameters are significantly less efficient than the optimized ones while the extended model is significantly better than the other models.

To get a better idea of the successes and failures of the extended system, Figure 2 presents a bubble plot of the true and predicted scores of all tokens in the 7-fold CV, the size of a bubble being proportionally related to the number of observations in this area. It clearly shows that the system effectiveness is low at predicting the highest scores (see the large variability of the predicted scores for true scores of 8 and 9), which are the most important for obtaining a high *Match_m* in the challenge.

In this figure, three areas deserve special attention. The bottom left corner contains a large number of effective predictions. These are very largely punctuation marks and grammatical words correctly predicted by the model as having to obtain (in general) low scores. For example, out of the 289 *that* in the combined Train and Dev sets, 96.5% have an emphasis score less than or equal to 2 and none have a score higher than 5.

	You	're	**never**	a	loser	until	you	quit	trying	.
True score for Q835	1	1	**1**	0	4	1	1	8	4	3
Prediction for Q835	1	2	**8**	4	7	1	1	7	4	2
True score for Q1057	3	3	**9**	7	7	2	2	7	6	3
Prediction for Q1057	1	1	**2**	1	5	1	1	7	4	2

Table 3: True and predicted scores for the item which occurs twice in the Train set.

The two areas highlighted in Figure 2 contain tokens for which the errors are the greatest. They correspond among others to grammatical words considered by the annotators to be emphasized because they are part of a chunk such as *university* **of** *life*. There is also a somewhat strange case (see Table 3): the same item occurring twice in the Train set on which the raters who evaluated these two versions strongly disagreed. The model is mistaken each time on the word *never*, but once it gives it a score that is far too low and once a score far too high. The explanation for this contrasting behavior is that the two statements were not in the same CV test set. It thus appears that the model is very sensitive to the specific instances provided during learning.

5 Conclusion

The system proposed for Task 10 of SemEval-2020 to select tokens to be highlighted in short texts was mainly based on precomputed embedding models and LightGBM. Its performance was mediocre since it did barely better than the baseline. It would not be correct to draw an argument from the fact that the human annotators themselves do not agree with each other in this task since other teams have proposed systems capable of achieving a much better performance. The performance of the extended system might have been improved if a weighting function for the features had been used such as the bi-normal separation feature scaling (Forman, 2008) or BM25 which has proved useful in the VarDial challenge (Bestgen, 2017). Yet, it seems to me that the major limitation of the proposed system is to take into account very little contextual information.

Acknowledgements

The author is a Research Associate of the Fonds de la Recherche Scientifique - FNRS (Fédération Wallonie Bruxelles de Belgique). Computational resources have been provided by the supercomputing facilities of the Université Catholique de Louvain (CISM/UCL) and the Consortium des Equipements de Calcul Intensif en Fédération Wallonie Bruxelles (CECI).

References

Yves Bestgen. 2017. Improving the character ngram model for the DSL task with BM25 weighting and less frequently used feature sets. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pages 115–123, Valencia, Spain, April. Association for Computational Linguistics.

Yves Bestgen. 2020. Reproducing monolingual, multilingual and cross-lingual CEFR predictions. In *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, pages 5597–5604, Marseille, France, May. European Language Resources Association (ELRA).

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, June. Association for Computational Linguistics.

George Forman. 2008. BNS feature scaling: an improved representation over tf-idf for svm text classification. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 263–270, New York, NY, USA. ACM.

Jerome H. Friedman. 2001. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232.

David Howell. 2008. *Méthodes statistiques en sciences humaines*. De Boeck Université, Bruxelles.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3146–3154. Curran Associates, Inc.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2227–2237, New Orleans, Louisiana, June. Association for Computational Linguistics.

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*.

Han Xiao. 2018. Bert-as-service. `https://github.com/hanxiao/bert-as-service`.