UAntwerp at SemEval-2021 Task 5: Spans are Spans, stacking a binary word level approach to toxic span detection

Ben Burtenshaw and **Mike Kestemont**
Antwerp Centre for Digital Humanities and Literary Criticism
University of Antwerp
Prinsstraat 13, 2000
Antwerp (Belgium)
firstname.lastname@uantwerpen.be

Abstract

This paper describes the system developed by the Antwerp Centre for Digital humanities and literary Criticism [UAntwerp] for toxic span detection. We used a stacked generalisation ensemble of five component models, with two distinct interpretations of the task. Two models attempted to predict binary word toxicity based on ngram sequences, whilst 3 categorical span based models were trained to predict toxic token labels based on complete sequence tokens. The five models' predictions were ensembled within an LSTM model. As well as describing the system, we perform error analysis to explore model performance in relation to textual features. The system described in this paper scored **0.6755** and ranked **26th**.

1 Introduction

SemEval 2021 Task 5: Toxic Spans Detection was organised by John Pavlopoulos and colleagues, and described in detail in their task description paper (Pavlopoulos et al., 2021). Competing teams were asked to develop systems capable of detecting spans of toxic text. Predictions were evaluated using a pairwise F1-score of toxic character offset predictions, described in section 5.1.

Initial analysis of the development data revealed that toxic spans were varied in content and not limited to single words. Though most examples contained single toxic words or phrases, others contained longer spans and complete sentences. Figure 1 illustrates this phenomena. With this in mind, we sort a strategy that combined longer span based detection with binary word classification.

Table 1 reveals that toxic spans were on average $\tilde{3}$ times longer in the development set, whilst stop words were $\tilde{4}$ times more frequent. Figures 8 and 9 shows the frequency of these features in relation to model performance.

# 38	... while brutal scum hold sway in our institutions.
# 36	Why even bother to give this woman a time of day. Bury her in the cotton field where she belong.
# 31	other religions aren't carrying out the violent and extremist doctrines as Islamists are.
# 34	I hope he and other car theves read this, I'll shoot you if I catch you in my vehicles stealing then call the police.

Figure 1: Example of possible annotation rationales.

		dev			test		
		mean	total	std	mean	total	std
TOX	words	3.22	25547	9.37	1.24	2488	3.59
	stop_ws	0.98	7802	3.82	0.26	529	1.73
NOT	words	32.52	258165	34.39	31.44	62884	30.87
	stop_ws	16.19	128533	17.55	15.45	30909	15.75
	support		*7939*			*2000*	

Table 1: Macro values for toxic and non-toxic spans for development and test data. Mean average, total number, and standard deviation are shown for all words and stop words. Support is shown from development and test data as the total number of samples

Strategy We combined models that used antithetical contexts, i.e. full sequences, and shorter ngram sequences before and after a given word. This approach is based on the hypothesis that their predictions would have a low correlation, and in turn, they would create ideal ensemble components.

Results The system described in this paper scored **0.6755** and ranked **26th**. We discovered that model correlation did play a factor in the accuracy of an ensemble approach; however, much of this performance increase was lost in transition to test data, where correlation increased on the most frequent type of examples. In section 5.3 we analyse model performance and correlation in relation to textual features.

Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021), pages 898–903
Bangkok, Thailand (online), August 5–6, 2021.

2 Background

Toxic span detection is a development of binary toxicity detection which has garnered recent attention, in the form of shared-tasks and datasets (Wulczyn et al., 2017; Zampieri et al., 2019).

Features Teams were supplied with development data consisting of 7939 text samples in varying lengths up to 1000 characters, and tested on 2000 text samples.

Target Span detection asks systems to detect which specific series of characters are toxic, irrespective of the text's overall toxicity. Figure 2 illustrates the target value for SemEval 2021 Task 5. Unlike Named Entity Recognition, systems were not scored on their performance at negative, beginning, middle, or end token detection. This target definition led to a focus on positive optimisation, where false positives were of more importance than true negatives. In section 5.3 on error analysis we compare model scores using a binary word level representation of toxicity, that scores both positive and negative prediction.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
S	a	i	d		a		v	i	l	l	a	g	e		i	d	i	o	t	.

Figure 2: Illustration of toxic span character offsets.

3 System overview

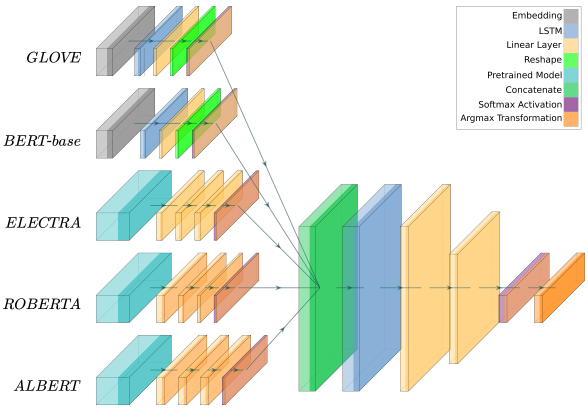

Figure 3: Model Diagram including all component models. Colours represent layer types and arrows represent training pipeline.

Task Interpretations We used two types of component models; binary word level models and categorical span based models, and combined those in an LSTM network (Hochreiter and Schmidhuber, 1997). We used two word based models [GLOV, BERT] and three span based models [ALBE, ROBE, ELEC], the softmax output of all models were concatenated and supplied to an LSTM model [ENSE].

Motivation We intended for the word based models to learn local features in the tokens nearest the target word, and for the span based to learn the overall features that affected sub and multi word toxicity.

3.1 Baselines

To interpret the task we relied on the Spacy implemented baseline shared by the organizers and described in the task description paper (Pavlopoulos et al., 2021; Honnibal et al., 2020). The approach retrained the RoBERTa based `en_core_web_trf` model's `ner`, `trf_wordpiecer`, and `trf_tok2vec` components, producing f1-scores of 0.5630 on the development data and 0.6305 on test data. To Interpret the problem further, we implemented two simple baselines.

Lexical Lookup Using a subset of samples from the development data, we created a toxic words list from all words within toxic spans, except for stop words [1]. On the test data, we then classified words as toxic if they appeared within the aforementioned toxic words list. We then converted word offsets into character offsets. This approach achieved an F1-score of 0.4161 on the test data.

SVM Using Term Frequency to Inverse Document Frequency we created two document vector representations of toxic and non-toxic spans. Using a Support Vector Machine, we predicted the probability that a word vector appeared within a toxic or non-toxic document (Salton and McGill, 1986; Wu et al.). We then used a binary threshold of 0.5 and class weights based on relative label frequency to predict whether a word was toxic. This approach achieved an F1-score of 0.5489 on the test data.

3.2 Component Models

3.2.1 Span Prediction

Span prediction models used the complete sequence of words, up to a maximum length, to pre-

[1] The toxic words list was created from the first 5800 samples of the development data. We used Spacy tokenisation and English stop words list, and we removed space and character offsets from predictions.

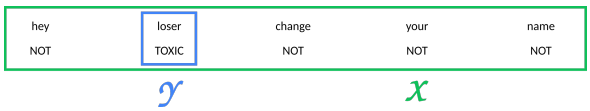

Figure 4: Illustration of toxic span prediction based on complete sequence.

dict toxic character offsets. Sequences were represented as token reference indexes, described in section 4.1. The target sequence was processed from character offsets into categorical arrays for toxic, non-toxic, and padding tokens. 4.1.

Transformer Models We selected three pre-trained transformer models (ALBERT, RoBERTa, ELECTRA) and fine-tuned them for this task with extra linear layers. We performed separate hyper-parameter optimisation for each model, detailed in section 4.2. ALBERT is a lightweight implementation of a BERT model (Lan et al., 2020; Devlin et al., 2019) that uses feature reduction to reduce training time. ELECTRA is a further development of the BERT model that pre-trains as a discriminator rather than a generator (Clark et al., 2020). RoBERTa develops the BERT model approach for robustness, (Liu et al., 2019). During development we found that these three transformer models achieved the highest f1-scores in relation model correlation compared to alternatives. All models used the Adam optimizer (Kingma and Ba, 2017).

3.2.2 Binary Word Prediction

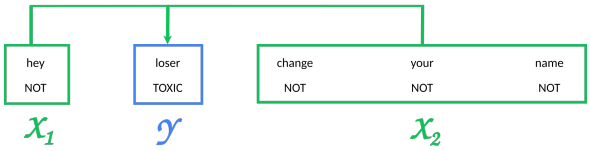

Figure 5: Illustration of toxic word prediction based on sequence before and after target word.

The binary word level models treated the task as word toxicity prediction based on a sequences of words before and after the target word. Figure 5 illustrates this approach. The target word toxicity was represented as a binary value. The sequence length before and after the target word was optimised for each model, and described in section 4.2.

Siamese-LSTM with Glove Word Embeddings A Siamese LSTM model used two networks based on separate glove embeddings of the sequence of words before and after the target words (Bao et al., 2018; Baziotis et al., 2017).

Task Representation	
input_text	"There are still morons"
target_labels	17, 18, 19, 20, 21, 22

BERT Representation					
input_tokens	there	are	still	mor	##ons
word_ids	0	1	2	3	3
target_labels	0	0	0	1	1

Figure 6: Input features and target labels for an example sequence, comparing a BERT specific token representation with the character offset representation defined by organisers (Pavlopoulos et al., 2021).

LSTM Finetuning BERT-base An LSTM model was trained based on the output of a BERT-base model. The words before and after the target word were used as model features, and the target word toxicity was represented as a binary value (Devlin et al., 2019).

3.3 Ensemble Model

A Bidirectional LSTM model was used to predict token toxicity based on tokenised word features and component model predictions. The model used transformer style feature representations to predict a sequence of categorical representations for token toxicity, as described in section 4.1. The ensemble model relied on five fold cross validation, as described in section 4.2.

3.3.1 Component model Predictions

Component model predictions were concatenated together as categorical representations of labels (not toxic, toxic, padding : 0,1,2). Each model's 3 dimensional output (number of samples, sequence length, number of labels) was permuted into a 4 dimensional matrix (number of samples, sequence length, number of labels, number of models).

4 Experimental setup

4.1 Pre-Processing

Tokenisation Text sequences were tokenised into character sequences using a BERT tokenizer and excess characters were replaced with a # character, as shown in Figure 6 (Devlin et al., 2019). Sequences were padded and truncated for uniformity to a length of 200 tokens. Longer sequences were handled separately, and predictions were combined in post-processing, described in section 4.4.

Target Label Representation To best suit the component models, we used a target representation based on the character sequences from the BERT tokenizer. Each word-like sequence was given a label based on its `word-id`, and converted into categorical binary arrays, or one-hot vectors. This is illustrated in Figure 6.

4.2 Training and Optimisation

Cross Validation We used stratified k fold validation of the development data to train all models. After optimisation, each component model's predictions on the *test* portion of fold k were added to the *train* portion of the other folds. Producing unseen training features for the ensemble model. This process avoids overfitting in component models, and facilitates training an ensemble model on the complete development data (Fushiki, 2011; Pedregosa et al., 2011).

Hyper-Parameter Optimisation Model parameters were optimised for each fold of the development data and the best models were used by the ensemble model. Table 2 shows the optimum parameters for each model used on the test data. We used Bayesian optimization for each fold of the development data to find optimum parameters (Snoek et al.). Component models were selected based on their f1-score and prediction correlation to other models. The ensemble model was trained on the predictions of the optimum model for each fold of the development data, expanded on in Section 4.3.

	span-based			word-based		
method	ELEC	ROBE	ALBE	GLOV	BERT	ENSE
dropout	0.05	0.40	0.23	0.4	0.3	0.23
epochs	4	4	4	20	6	4
layers	2	2	3	3	3	3
nodes	9	3	6	20	3	6
neg_weight	1.00	0.92	1.12	0.6	1.0	1.0
pos_weight	1.00	1.24	0.94	6.0	1.0	1.0
dev_F1	0.665	0.663	0.682	0.647	0.656	0.702
test_F1	0.673	0.662	0.672	0.637	0.634	0.675

Table 2: Table of the best model parameters. Pairwise F1 scores are shown for all span based models.

4.3 Prediction

To predict spans for submission, a version of each component model optimised for each fold of the development data was supplied the test data and their outputs were averaged. The ensemble model was then supplied component model predictions and tokenised text sequences.

4.4 Post-processing

Model output was converted from 2 dimensional token-level categorical arrays (n tokens, n labels) into character offsets. The character offsets of each positively labeled token was then added to a list, as illustrated in Figure 6. The predictions of sequences that had been truncated during preprocessing, were combined and duplicates were removed.

5 Results

Table 3 reveals that the ensemble model achieved a similar score on both development and test data, while the ALBERT, ELECTRA, and baseline models improved in testing. Crucially, the $\widetilde{5}\%$ increase in f1-score from component models to ensemble, that we see on the development data, was not transferred to the test data.

	dev			test		
	F1	P	R	F1	P	R
ENSE	0.6736	0.6664	0.7000	**0.6755**	0.6538	0.7182
ALBE	0.6284	0.6966	0.6677	0.6684	0.6695	0.6995
ELEC	0.6390	0.6975	0.6936	0.6668	0.6459	0.7296
ROBE	0.6418	0.7047	0.6908	0.6192	0.5771	0.7386
BERT	0.6568	0.6209	0.6260	0.5568	0.4209	0.5260
GLOV	0.6378	0.5850	0.5547	0.4378	0.4850	0.5547
BASE	0.5523	0.6247	0.5630	0.6305	0.5969	0.6548

Table 3: Scores on development and test data. The final submitted system predictions [ENSE] are shown in bold and component models are shown in italic.

5.1 Task Specific Evaluation Metrics

Systems are evaluated with an F1 score of character offsets (Pavlopoulos et al., 2021) . In cases where predicted spans are empty, 1 is given when true spans are empty and 0 is given if there are any true spans.

5.2 Model Correlation

Figure 7 reveals that the ensemble and ALBERT models have a high correlation, a logical outcome of their shared base layers; whilst word based models [BERT, GLOV] have a low correlation, reflecting their diverse interpretations.

Figure 7: Model Correlation calculated using a macro average f1-score

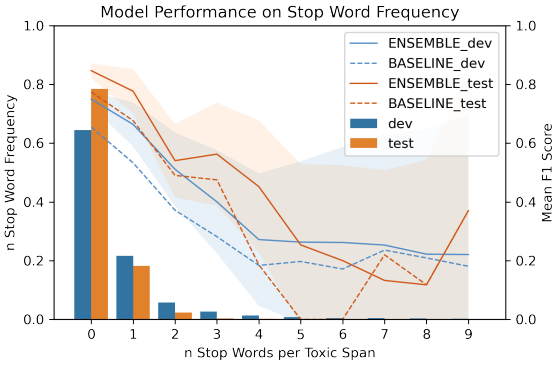

Figure 9: Model F1 score at n stop words per toxic span, and n stop word frequency.

5.3 Error Analysis

We performed error analysis to interpret the hypothesis that there are multiple annotation rationales; single toxic words, and longer offensive sentences, illustrated in Figure 1.

Toxic Span Length Figure 8 reveals that the length of toxic spans had an impact on model performance. Models were less accurate at detecting longer spans on both development and test data. Furthermore, the impact of this effect on test data was decreased as there were fewer longer toxic spans.

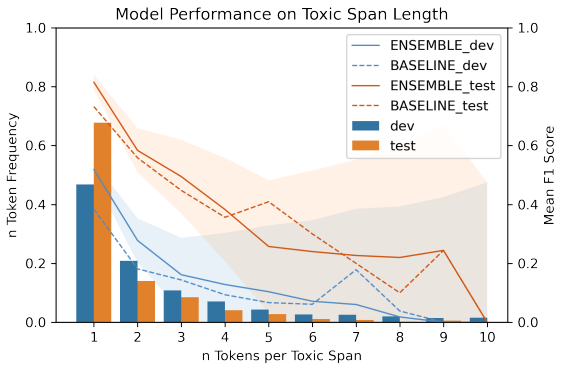

Figure 8: Model F1 score at n tokens per toxic span. Bars show the frequency of n tokens in development and test data. Shaded areas shows standard deviation of the f1-score for the ensemble model.

Stop Words in Toxic Spans The frequency of stop words in toxic spans also affected model performance. Figure 9 reveals that, where present, spans with more stop words caused lower model accuracy.

Binary Token Level Evaluation By using token level scoring we are able to reveal how the models perform on both positive and negative tokens. Here, the target labels are represented as binary arrays; 1 for toxic tokens and 0 for non-toxic. We can not expect these calculations to align with character offsets, due to variance in tokenisation and parsing.

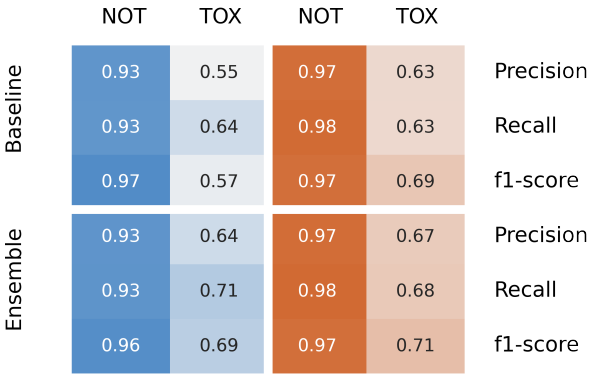

Figure 10: Binary token level scores for precision, recall, and f1-score.

6 Conclusion

Our initial hypothesis, that combining word based and span based approaches would yield a significant performance boost, did not stand up. We measured a $\widetilde{5}\%$ increase in f1-score on development data, but this was not transferred to test data. In future work, we would look to a strategy that incorporated model transferability in component model selection, with the intention of better handling fluctuations in annotation rationale. Drawing on recent work (Fortuna et al., 2021).

References

W. Bao, W. Bao, J. Du, Y. Yang, and X. Zhao. 2018. Attentive Siamese LSTM Network for Semantic Textual Similarity Measure. In 2018 International Conference on Asian Language Processing (IALP), pages 312–317.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. DataStories at SemEval-2017 Task 6: Siamese LSTM with Attention for Humorous Text Comparison. In Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), pages 390–395, Vancouver, Canada. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, and Quoc V Le. 2020. ELECTRA: PRE-TRAINING TEXT ENCODERS AS DISCRIMINATORS RATHER THAN GENERATORS. page 18.

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.

Paula Fortuna, Juan Soler-Company, and Leo Wanner. 2021. How well do hate speech, toxicity, abusive and offensive language classification models generalize across datasets? Information Processing & Management, 58(3):102524.

Tadayoshi Fushiki. 2011. Estimation of prediction error by using K-fold cross-validation. Statistics and Computing, 21(2):137–146.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. Neural computation, 9(8):1735–1780.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength natural language processing in python.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs].

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. arXiv:1909.11942 [cs].

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. arXiv:1907.11692 [cs].

John Pavlopoulos, Léo Laugier, Jeffrey Sorensen, and Ion Androutsopoulos. 2021. Semeval-2021 task 5: Toxic spans detection (to appear). In Proceedings of the 15th International Workshop on Semantic Evaluation.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. the Journal of machine Learning research, 12:2825–2830.

Gerard Salton and Michael J McGill. 1986. Introduction to modern information retrieval.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical Bayesian Optimization of Machine Learning Algorithms. page 9.

Ting-Fan Wu, Chih-Jen Lin, and Ruby C Weng. Probability Estimates for Multi-class Classification by Pairwise Coupling. page 31.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex Machina: Personal Attacks Seen at Scale. arXiv:1610.08914 [cs].

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. SemEval-2019 Task 6: Identifying and Categorizing Offensive Language in Social Media (OffensEval). In arXiv:1903.08983 [Cs].