NLRG at SemEval-2021 Task 5: Toxic Spans Detection Leveraging BERT-based Token Classification and Span Prediction Techniques

Gunjan Chhablani[*]
Dept. of CS&IS
BITS Pilani, Goa, India
chhablani.gunjan@gmail.com

Abheesht Sharma[*]
Dept. of CS&IS
BITS Pilani, Goa, India
f20171014@goa.bits-pilani.ac.in

Harshit Pandey[*]
Dept. of CS
Pune University, India
hp2pandey1@gmail.com

Yash Bhartia
Dept. of CS&IS
BITS Pilani, Goa, India
f20190151@goa.bits-pilani.ac.in

Shan Suthaharan
Dept. of CS
UNC-Greensboro, NC, USA
s_suthah@uncg.edu

Abstract

Toxicity detection of text has been a popular NLP task in the recent years. In SemEval-2021 Task-5 Toxic Spans Detection, the focus is on detecting toxic spans within English passages. Most state-of-the-art span detection approaches employ various techniques, each of which can be broadly classified into Token Classification or Span Prediction approaches. In our paper, we explore simple versions of both of these approaches and their performance on the task. Specifically, we use BERT-based models - BERT, RoBERTa, and SpanBERT for both approaches. We also combine these approaches and modify them to bring improvements for Toxic Spans prediction. To this end, we investigate results on four hybrid approaches - Multi-Span, Span+Token, LSTM-CRF, and a combination of predicted offsets using union/intersection. Additionally, we perform a thorough ablative analysis and analyze our observed results. Our best submission - a combination of SpanBERT Span Predictor and RoBERTa Token Classifier predictions - achieves an F_1 score of 0.6753 on the test set. Our best post-eval F_1 score is 0.6895 on intersection of predicted offsets from top-3 RoBERTa Token Classification checkpoints. These approaches improve the performance by 3% on average than those of the shared baseline models - RNNSL and SpaCy NER.

1 Introduction

Offensive language can include various categories such as threats, vilification, insults, calumniation, discrimination and swearing (Pavlopoulos et al., 2019). Detection of such language is necessary for ease of moderation of content on social media. Despite their popularity, toxicity detection tasks have focused majorly on sequence classification, rather than sequence tagging. Finding which spans make a comment or document toxic in nature is crucial in explaining the reasons behind their toxicity. Additionally, such attributions would allow for more efficient semi-automated quality-based moderation of content, especially for verbose documents, in comparison to quantitative toxicity scores.

In SemEval-2021 Task-5, Pavlopoulos et al. (2021) provide a dataset of 10k English texts filtered from Civil Comments (Borkan et al., 2019) dataset. Each text is crowd-annotated with character offsets that make the text toxic. The task is to predict these character offsets given the text. The work presented in this paper aims to provide a comprehensive analysis of simple Token Classification (TC) and Span Prediction (SP) methods across multiple BERT-based models - BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and SpanBERT (Joshi et al., 2020). Additionally, we experiment with a few hybrid approaches - Multi-Span (MSP), where the model is trained on multiple spans simultaneously; Span+Token (SP-TC), where the model is trained on both kinds of tasks simultaneously; LSTM-CRF (LC), which uses a LSTM and CRF layer on top of BERT-based models; and a combination of predicted offsets for above techniques using union/intersection. In Section 2, we perform a compendious literature survey. Section 3 elucidates our approach, including the modelling aspect, the various variants of the base model, and the different Hybrid Systems. In Section 4, we describe our experimental setup and hyperparameters used for our methods. Lastly, in Section 5 we analyze our results and perform ablative analysis on our systems.

2 Background

Before the advent in research pertaining to toxic texts, Warner and Hirschberg (2012) modeled hate

[*] Equal contribution. Author ordering determined by coin flip.

Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021), pages 233–242
Bangkok, Thailand (online), August 5–6, 2021.

speech as a word sense disambiguation problem where SVM was used for classification of data. Mehdad and Tetreault (2016) used RNN Language Model with character and token based methods to classify the text. Recently, however, toxic text detection has garnered a lot of attention (Nobata et al., 2016; Park and Fung, 2017; Pavlopoulos et al., 2017; Wulczyn et al., 2017). The increase in offensive language research can partly be credited to various workshops such as Abusive Language Online[1] (Waseem et al., 2017) , as well as other fora, such as GermEval for German texts,[2] or TRAC (Kumar et al., 2018) and Kaggle challenges[3].

Hanu and Unitary team (2020) introduced Detoxify, a comment detection library modeled using HuggingFace's transformers (Wolf et al., 2020) to identify inappropriate or harmful text online as a result of participation in three such challenges. In a contemporary work, Pavlopoulos et al. (2020) discuss context requirement for toxicity detection.

In SemEval 2020-Task 11 (Da San Martino et al., 2020), the first sub-task - Span Identification - aims at detecting the beginning and the end offset for the propaganda spans in news articles. This sub-task is similar to SemEval 2021-Task 5. The proposed approaches for the sub-task can be broadly classified into Span Prediction or Token Classification. Most teams use multi-granular transformer-based systems for token classification/sequence tagging (Khosla et al., 2020; Morio et al., 2020; Patil et al., 2020). Inspired by Souza et al. (2019), Jurkiewicz et al. (2020) use RoBERTa-CRF based systems. Li and Xiao (2020) use a variant of SpanBERT span prediction system.

3 Models

3.1 Token Classification Models

3.1.1 Baseline Models

From the models already provided with the dataset, we use RNNSL and SpaCy NER Tagging baselines for token-wise classification.

RNNSL model is a combination of a single Bi-LSTM layer with a randomly initialized embedding layer. It uses a three-label classification task for each word in the sentence. The labels used are: *special token*, *non-toxic word*, and *toxic word*. For each word, the corresponding offsets are added to the predicted spans. A word with containing any toxic offset is marked as toxic during training.

SpaCy NER Tagging model is an NER classifier built on SpaCy Language Models. It is used to predict the entities which are labelled as *TOXIC* in the text using the spans provided.

3.1.2 BERT-based Token Classification Models

These models comprise a BERT-based model and a classification layer over each final token embedding which predicts whether a token is toxic or not. Based on these classifications, we add the offsets for those tokens (not words) which are marked as toxic by the model. Figure 1a represents a Token Classification Model.

3.2 Span Prediction Models

3.2.1 BERT-based Span Prediction Models

We use the BERT-based Span Prediction (Figure 1c) models based on Extractive Question Answering systems similar to work on SQuAD (Rajpurkar et al., 2016) and MRQA (Fisch et al., 2019). In these systems, the output at each token is a start logit and an end logit denoting whether that token is a start token or an end token of the span, depending on the softmax value. Since the Toxic Spans text can have multiple toxic spans, we take different contiguous spans from the given offsets, and make several 'samples' out of the example. Each span becomes an 'answer' for the particular text sample. We use the word *'offense'* as a dummy question. Thus, each contiguous span leads to one 'sample' for every example (Table 1).

Text		**Spans**
...an idiot - just an embarrassingly uninformed, ignorant,...		idiot, ignorant
Question	**Context**	**Answer**
offense	...an idiot - just an embarrassingly uninformed, ignorant,...	idiot
offense	...an idiot - just an embarrassingly uninformed, ignorant,...	ignorant

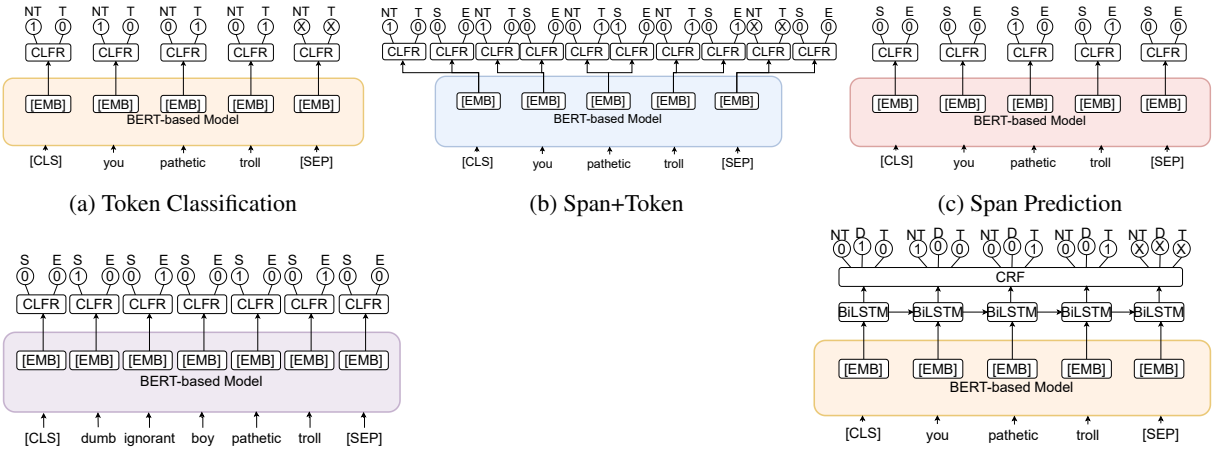
Table 1: Conversion of Toxic Spans example to samples for single-span Span Prediction.

We store the start index of the text, similar to the SQuAD (Rajpurkar et al., 2016) dataset, and process the data to provide start and end token positions during training. The classifier layer on top of the encoder embeddings performs a binary classification task for start and end positions. A

[1] https://sites.google.com/site/abusivelanguageworkshop2017/

[2] https://projects.fzai.h-da.de/iggsa/

[3] Jigsaw Toxic Comment Classification Challenge



(a) Token Classification (b) Span+Token (c) Span Prediction

(d) Multi-Spans (e) LSTM-CRF

Figure 1: BERT-based Approaches*

*CLFR = Classifier, [EMB] = Token Embedding, NT = Non-Toxic, T = Toxic, D = Dummy, X = Don't Care, S = Start, E = End.

span is scored using the sum of predicted start and end logits. From top-K start and end logits, valid predicted answer spans[4] are chosen during post-processing. A union of all the corresponding offsets is taken to give the final prediction for the example. A threshold is learned on the span scores using the resulting dev set F_1 score on offsets, which is then used for test set prediction. All spans with score above threshold are considered to be toxic spans.

3.3 Hybrid Systems

3.3.1 Multi-Spans

In Section 3.2, we allow each context to have multiple single-span answers during training. This is counter-intuitive, as the model is only trained to handle a single-span at a time, and expected to predict multiple single-spans during prediction. Two toxic spans in text are equally important to predict, and thus, should not be shown at different times during training. To mitigate this issue, we try an approach which we refer to as the 'Multi-Spans' (MSP) approach. Here, we take all the ground start and end token positions during training, and use Binary Cross Entropy on each of the start/end logits. This essentially treats the task as a multi-label classification problem. Hence, during training, all the ground spans are used in the same iteration with the example, and only one 'sample' per example is generated. Figure 1d depicts a representation of the system. Note that two tokens - *dumb* and *pathetic* are marked as the start token. Similarly, both *ignorant* and *troll* are marked as the end token.

[4]Valid spans are those which have end index greater than start index, and length less than a maximum span length.

3.3.2 LSTM-CRF

A recently popular approach in Named-Entity Recognition tasks has been to use Conditional Random Fields (CRF) with BERT-based models. Inspired by the CRF-based approaches (Souza et al., 2019; Jurkiewicz et al., 2020), we use BERT-based models with a single BiLSTM layer and a CRF layer. During training, the CRF loss is used and during prediction, Viterbi Decoding is performed. Though CRF is generally used for word-level classification, we do not mask inner and end tokens for a word as it degrades dev set performance for our systems. Hence, all the tokens of a word are considered for classification.

3.3.3 Spans+Token

For this system, we use a combination of the two tasks - Token Classification and single-span Span Prediction. We use two classification layers on the token-wise embeddings - one for start and end prediction, and the other for token classification. Training is done simultaneously on both tasks, and the cross-entropy loss for each classifier is weighted. The overall loss is given as:

$$L(\hat{s}, \hat{e}, \hat{p}, s, e, p) = -\sum_t \hat{p}_t \log p_t - \frac{(\sum_t \hat{s}_t \log s_t + \sum_t \hat{e}_t \log e_t)}{2}$$

where s_t,e_t, and p_t are labels for start, end and token classifiers for token t, while $\hat{s_t}$,$\hat{e_t}$ and $\hat{p_t}$ are predictions. This is done to equally scale both SP and TC task losses. During prediction, we consider top-K start and end scores. From the valid spans,

the score is calculated as the average of start and end logit scores, as well as the mean of toxicity logits over the span under consideration. The score is given as:

$$S(i_s, i_e) = \frac{\hat{s}_{i_s} + \hat{e}_{i_e}}{2} + \frac{\sum_{k=i_s}^{i_e} \hat{t}_k}{e - s + 1}$$

where i_s and i_e are start and end indices, $\hat{s}_{i_s}$ and $\hat{e}_{i_e}$ are start and end logits at those indices, and $\hat{t}_k$ is toxicity logit at index k. A threshold, similar to Section 3.2 is tuned on the dev set. The predicted offsets taken from the predicted spans are considered to be toxic.

3.3.4 Combination of Offset Predictions

Chen et al. (2017) proposed using the predictions from top few checkpoints and averaging the results to achieve better classification scores. Based on a similar line of thought, we also combine the predicted spans for various checkpoints of a model, as well as across different models using union or intersection.

4 Experimental Setup[5,6]

4.1 Hardware Requirements

The training and the evaluation of systems was performed on Google Colab's free GPU (NVIDIA K80/P100). The training time varies with the models. For each model, it is around 4-6 hours, which is well-within the 12 hour limit of Colab.

4.2 Models & Hyperparameters

For RNNSL, a Keras-based BiLSTM model is provided. We use a max length of 192, batch size of 32 and a dropout of 0.1. The training is done using Adam Optimizer with early stopping ($patience\_period = 3$), which in our case halts at 5 epochs. The embedding/hidden_state size used is 200. A threshold is used to classify a word as toxic on the predicted toxic word probability. This threshold is tuned on the trial dataset. For SpaCy, the *en_core_web_sm* model is used with 30 iterations.

For all BERT-based models, we use HuggingFace's transformers (Wolf et al., 2020) in PyTorch. For CRF, we use the pytorch-crf (Kurniawan, 2018) library. We use a batch size of 4, train for 3 epochs, use a linear learning rate decay, and an AdamW optimizer with a weight decay of 0.01. The initial learning rate is $2e{-}5$. During tokenization, the maximum length allowed is 384, with the exception of RoBERTa Span+Token where it is 512. We use *LARGE* models for all - BERT, RoBERTa and SpanBERT, unless otherwise specified.

For Token Classification, we add a label for the *[CLS]* token if the percentage of toxic offsets in text is greater than 30% in order to provide a proxy text classification objective for the system. For span-based models, the K used for top-K start and top-K end logit selection is 20, and the maximum allowed answer length is 30 tokens. For LSTM-CRF systems, a dummy label is used for the *[CLS]* token, while the prediction mask for other special tokens is set to 0. A dropout of 0.2 is used. For Span Prediction systems, the overlapping stride is set to 128.

The training dataset used is *tsd_train.csv* and the dev set used is *tsd_trial.csv* file, unless otherwise specified. For all systems, we evaluate the F_1 scores using the provided script on the checkpoints which give the lowest dev set loss.

5 Results and Analysis

In favor of brevity, for this section, we use the following abbreviations: BT=BERT, RBTa=RoBERTa, SBT=SpanBERT, SP=Span Prediction, TC=Token Classification, MSP=Multi-Span, LC=LSTM-CRF, B=Base, TBT=ToxicBERT, TRBTa=ToxicRoBERTa, TT=Trained on Train+Trial, (x,∩)=Intersection of offsets from x-best checkpoints, (x,∪)=Union of offsets from x-best checkpoints.

In Table 2, we mention scores for our approaches. The scores are evaluated are performed after the evaluation phase, using the hyperparameters mentioned in Section 4.2. We observe that the highest score is obtained by SBT-TC (0.6856). The baseline scores (RNNSL/SpaCy) are good (≈0.65) considering that these models are not pre-trained. Notably, SP systems perform worse than their TC counterparts. A good reason could be the self-attention used in BERT-based models. Since the interaction is between tokens, and not spans, it is expected that each token is well represented and less consideration will be given to the span representation around a single token. The reason why SBT-TC performs best out of all the LARGE models could be the random-spans Masked Language

[5] Our code can be found at: https://github.com/gchhablani/toxic-spans-detection.

[6] We also use Integrated Gradients to understand what the models focus on. For discussion, see Appendix B.

Model	Train F_1	Trial F_1	Test F_1
RNNSL	0.5904	0.5904	0.6514
SpaCy	0.6282	0.5729	0.6573
BT-TC	0.6944	0.6942	0.6781
RBTa-TC	0.6791	0.6769	0.6834
SBT-TC	0.6873	0.6789	**0.6856**
BT-SP	0.6639	0.6465	0.6663
RBTa-SP	0.6401	0.6386	0.6665
SBT-SP	0.6432	0.6212	0.6561
BT-MSP	0.5218	0.4941	0.5406
RBTa-MSP	0.5056	0.4886	0.5244
SBT-MSP	0.5190	0.5004	0.5084
BT-SP-TC	0.6676	0.6214	0.6186
RBTa-SP-TC	0.6395	0.6101	0.5901
SBT-SP-TC	0.6608	0.6491	0.5959
BT-LC	0.6887	0.6843	0.6835
RBTa-LC	0.7236	0.6861	0.6787
SBT-LC	0.7200	0.6982	0.6801

Table 2: F_1 scores for our approaches (Post-Eval).

Modeling used in its pre-training. However, BERT and RoBERTa take over for other approaches.
LSTM-CRF approaches perform as good as Token Classification approaches, and BT-LC achieves the second highest score (0.6835). MSP performs poorly, in contrast to what is expected. Multi-Span Extraction is still an active problem in Deep NLP with only a few recent works (Segal et al., 2020; Yang et al., 2020) on it, which still incorporate sequence tagging approaches. Spans+Token approaches perform better than Multi-Span, but are worse than both TC and SP approaches across all BERT-based models.

Lastly, from combined checkpoint predictions

Combination	Test F_1
RBTa-TC(3,∪)	0.6765
RBTa-TC(3,∩)	**0.6895**
SBT-SP(3,∪)	0.5879
SBT-SP(3,∩)	0.6585
RBTa-TC(3,∪)∪SBT-SP	0.6573
RBTa-TC(3,∪)∩SBT-SP	0.6765
RBTa-TC∪ SBT-SP(3,∪)	0.5840
RBTa-TC∩SBT-SP(3,∪)	0.6883

Table 3: F_1 scores for combined predictions.

(Table 3), we get out best scoring system - RBTa-TC(3,∩) - which achieves a score of 0.6895. However, our best official submission[7] was a variant of the third best combination - RBTa-TC(3,∪)∩SBT-SP (0.6765). It is also observed that intersection approaches perform better than corresponding union and single checkpoints approaches, while union approaches perform worse than single checkpoints. This means that the individual checkpoints are predicting some extra offsets to be toxic.

5.1 Ablative Analysis

Model	Train F_1	Trial F_1	Test F_1
TBT-TC	0.6753	0.6628	0.6792
TRBTa-TC	0.7244	0.6954	0.6773
TBT-SP	0.6638	0.6560	0.6584
TRBTa-SP	0.6475	0.6358	0.6746
BT-B-TC	0.6966	0.6746	**0.6881**
RBTa-B-TC	0.6641	0.6482	0.6834
BT-B-SP	0.6605	0.6434	0.6611
RBTa-B-SP	0.6481	0.6464	0.6661
RNNSL-TT	0.6844	0.6882	0.6259
RBTa-TC-TT	0.7707	0.7788	0.6823
SBT-SP-TT	0.7116	0.7092	0.6669

Table 4: F_1 scores for ablative approaches.[8]

In Table 4, we present results on TBT[8] and TRBTa[9] for TC and SP approaches. These are *BASE* models fine-tuned on the Civil Comments Dataset. Since the Toxic Spans dataset has similar text data, we expect these models to perform better than *BASE* models. We observe that TBT-TC and TRBTa-SP perform slightly better than BT-TC and RBTa-SP, despite being *BASE* models. Also, BT-SP and RBTa-TC are only slightly better than their 'Toxic counterparts.
Yet, in comparison, *BASE* models - BT-B and RBTa-B, without any multi-stage pre-training perform better than their 'Toxic' counterparts, and are comparable, if not better than their *LARGE* counterparts. This means that there not enough data for *LARGE* models, and hence, they tend to overfit. However, the reasons behind worse performance of 'Toxic' systems is unclear.
We also evaluate scores for a few systems on the test set after 3 epochs of training on both train and trial data (-TT). We observe that the performance on both train and trial datasets increases significantly (≈7-10%), showing that these datasets have similar distribution. However, the performance on test decreases for RBTa-TC-TT and RNNSL-TT in comparison to the Table 2, which shows that test set distribution might be slightly different for TC task. For SBT-SP-TT, we see a slight increase, showing

[7] The most significant of our official submission scores are present in Appendix A.

[8] https://huggingface.co/unitary/toxic-bert

[9] https://huggingface.co/unitary/unbiased-toxic-roberta

scope of improvement for SP systems with more data.

Lastly, we evaluate the token-based predictions and span-based predictions for SBT SP-TC separately. Surprisingly, token predictions achieve a F_1 score of 0.6522 on the test set, which is much better than using both token and spans (0.5959). However, for span-based predictions, we only achieve an F_1 score of 0.1510. This means that the system is focusing heavily on token-based-predictions. Hence, we need to re-evaluate our architectural decisions in order to successfully incorporate both token and spans together.

6 Conclusion

Based on our results and analysis, we conclude that Token Classification systems have an edge over Span Prediction methods on this task. *BASE* models perform better than *LARGE* models in either of the approaches, which could imply need for more data to train *LARGE* models. Our Multi-Span approach performs poorly, but Span+Token approach shows some promise and we need to re-evaluate our architectural choices. The reason why ToxicBERT/ToxicRoBERTa perform worse than *BASE* models is also an avenue for further analysis. Finally, our individual BERT-based models tend to predict extra offsets for the task. While checkpoint ensembling using intersection is a good way to address this issue, we will explore other remedies in a future work.

Acknowledgments

We would like to acknowledge the help and moral support provided to us by Rajaswa Patil[10] and Somesh Singh[11]. We would also like to express our gratitude to our colleagues at the Language Research Group (LRG)[12], who have been with us at every stepping stone.

References

Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2019. Nuanced metrics for measuring unintended bias with real data for text classification. *CoRR*, abs/1903.04561.

Hugh Chen, Scott Lundberg, and Su-In Lee. 2017. Checkpoint ensembles: Ensemble methods from a single training process. *CoRR*, abs/1710.03282.

Giovanni Da San Martino, Alberto Barrón-Cedeño, Henning Wachsmuth, Rostislav Petrov, and Preslav Nakov. 2020. SemEval-2020 task 11: Detection of propaganda techniques in news articles. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1377–1414, Barcelona (online). International Committee for Computational Linguistics.

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.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13, Hong Kong, China. Association for Computational Linguistics.

Laura Hanu and Unitary team. 2020. Detoxify. Github. https://github.com/unitaryai/detoxify.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans.

Dawid Jurkiewicz, Łukasz Borchmann, Izabela Kosmala, and Filip Graliński. 2020. ApplicaAI at SemEval-2020 task 11: On RoBERTa-CRF, span CLS and whether self-training helps them. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1415–1424, Barcelona (online). International Committee for Computational Linguistics.

Sopan Khosla, Rishabh Joshi, Ritam Dutt, Alan W Black, and Yulia Tsvetkov. 2020. LTIatCMU at SemEval-2020 task 11: Incorporating multi-level features for multi-granular propaganda span identification. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1756–1763, Barcelona (online). International Committee for Computational Linguistics.

Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. 2020. Captum: A unified and generic model interpretability library for pytorch.

Ritesh Kumar, Atul Kr. Ojha, Shervin Malmasi, and Marcos Zampieri. 2018. Benchmarking aggression identification in social media. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pages 1–11, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

[10] https://rajaswa.github.io/

[11] https://someshsingh22.github.io/

[12] https://lrg.saidl.in/

Kemal Kurniawan. 2018. Pytorch-crf. https://github.com/kmkurn/pytorch-crf.

Jinfen Li and Lu Xiao. 2020. syrapropa at SemEval-2020 task 11: BERT-based models design for propagandistic technique and span detection. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1808–1816, Barcelona (online). International Committee for Computational Linguistics.

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. *CoRR*, abs/1907.11692.

Yashar Mehdad and Joel Tetreault. 2016. Do characters abuse more than words? In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 299–303, Los Angeles. Association for Computational Linguistics.

Gaku Morio, Terufumi Morishita, Hiroaki Ozaki, and Toshinori Miyoshi. 2020. Hitachi at SemEval-2020 task 11: An empirical study of pre-trained transformer family for propaganda detection. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1739–1748, Barcelona (online). International Committee for Computational Linguistics.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive language detection in online user content. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, pages 145—-153, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Ji Ho Park and Pascale Fung. 2017. One-step and two-step classification for abusive language detection on Twitter. In *Proceedings of the First Workshop on Abusive Language Online*, pages 41–45, Vancouver, BC, Canada. Association for Computational Linguistics.

Rajaswa Patil, Somesh Singh, and Swati Agarwal. 2020. BPGC at SemEval-2020 task 11: Propaganda detection in news articles with multi-granularity knowledge sharing and linguistic features based ensemble learning. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1722–1731, Barcelona (online). International Committee for Computational Linguistics.

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*.

John Pavlopoulos, Prodromos Malakasiotis, Juli Bakagianni, and Ion Androutsopoulos. 2017. Improved abusive comment moderation with user embeddings. In *Proceedings of the 2017 EMNLP Workshop: Natural Language Processing meets Journalism*, pages 51–55, Copenhagen, Denmark. Association for Computational Linguistics.

John Pavlopoulos, Jeffrey Sorensen, Lucas Dixon, Nithum Thain, and Ion Androutsopoulos. 2020. Toxicity detection: Does context really matter? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4296–4305, Online. Association for Computational Linguistics.

John Pavlopoulos, Nithum Thain, Lucas Dixon, and Ion Androutsopoulos. 2019. ConvAI at SemEval-2019 task 6: Offensive language identification and categorization with perspective and BERT. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 571–576, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Sahana Ramnath, Preksha Nema, Deep Sahni, and Mitesh M. Khapra. 2020. Towards interpreting BERT for reading comprehension based QA. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3236–3242, Online. Association for Computational Linguistics.

Elad Segal, Avia Efrat, Mor Shoham, Amir Globerson, and Jonathan Berant. 2020. A simple and effective model for answering multi-span questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3074–3080, Online. Association for Computational Linguistics.

Fábio Souza, Rodrigo Frassetto Nogueira, and Roberto de Alencar Lotufo. 2019. Portuguese named entity recognition using BERT-CRF. *CoRR*, abs/1909.10649.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 3319—-3328. JMLR.org.

William Warner and Julia Hirschberg. 2012. Detecting hate speech on the world wide web. In *Proceedings of the Second Workshop on Language in Social Media*, pages 19–26, Montréal, Canada. Association for Computational Linguistics.

Zeerak Waseem, Wendy Hui Kyong Chung, Dirk Hovy, and Joel Tetreault, editors. 2017. *Proceedings of the First Workshop on Abusive Language Online*. Association for Computational Linguistics, Vancouver, BC, Canada.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen,

Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex machina: Personal attacks seen at scale.

Junjie Yang, Zhuosheng Zhang, and Hai Zhao. 2020. Multi-span style extraction for generative reading comprehension.

A Official Submissions

During the evaluation period, we performed a 'cleaning' of the data by removing starting/trailing whitespace and punctuation characters in spans. Additionally, we include those partial words in spans which had more than half the number of characters in the span, and discard remaining partial words from spans. We considered this version of the *tsd_train.csv* and *tsd_trial.csv* to be 'clean train' and 'clean trial', respectively. During the post-eval period, we found out potential issues with the cleaning, and thus, we use original files. Additionally, since the distribution of *tsd_test.csv* is expected to be similar to *tsd_train.csv* and *tsd_trial.csv*, the scores are much better for models trained on *tsd_train.csv* file instead of *clean_train.csv*. However, some of our official submissions were from systems trained on the 'clean train' data. Keeping that in mind, we report our official scores for our top-few approaches in Table 5.

Model	**Trained On**	**Test** F_1
RNNSL	Train+Trial	0.6446
SpaCy	Train+Trial	0.6470
RNNSL ∪ Spacy	Train+Trail	0.6510
RBTa-TC	Clean Train	0.6270
RBTa-TC(3,∪)	Clean Train	0.6469
SBT-SP	Train	0.6631
RBTa-TC(3,∪) ∩ SBT-SP	Clean Train, Train	**0.6753**

Table 5: Official Submission Scores

B Integrated Gradients

We use Integrated Gradients(Sundararajan et al., 2017) from the Captum(Kokhlikyan et al., 2020) library for qualitative analysis of predictions for the SpanBERT-SP, and the RoBERTa-TC models. We calculate Integrated Gradients of the targets with respect to the embedding layer outputs. The Riemann Right numerical approximation method is used, with *n_steps=50*. Following Ramnath et al. (2020), we calculate token-wise importance distributions and word-wise distributions for a few examples. We refer the paper to the reader for more details.

For the Token Classification model, the targets are softmax outputs of toxicity logits of those tokens which the model predicts to be toxic, with a score greater than 0.5. For all such toxicity logits as targets, we calculate attributions with respect to the embedding layer outputs for all the tokens, and average them to get token-wise importance scores. For the Span Prediction model, we find start and end indices for all the predicted spans, and calculate respective attributions, add them, and then average them to get token-wise importance scores.

Text: offense See a shrink you pathetic troll .

Ground Spans: ['pathetic troll']
Predicted Spans: ['pathetic troll']

(a) SpanBERT Span Prediction

Text: See a shrink you pathetic troll.

Ground Spans: ['pathetic troll']
Predicted Spans: ['pathetic','troll']

(b) RoBERTa Token Classification

Figure 2: Qualitative Example of Attributions - Example 1

Text: offense Stupid is as stupid does Gump was right

Ground Spans: ['Stupid','stupid']
Predicted Spans: ['Stupid is as stupid']

(a) SpanBERT Span Prediction

Text: Stupid is as stupid does Gump was right

Ground Spans: ['Stupid','stupid']
Predicted Spans: ['Stupid','stupid']

(b) RoBERTa Token Classification

Figure 3: Qualitative Example of Attributions - Example 2

We observe in Figure 2a that the Span Prediction model performs correct prediction. However, on average, the word *'shrink'* gets higher importance

Text : Why does this author think she can demand, or is owed anything from either of these two people? One guy is a goon, the other is illiterate. They aren't law makers, teachers, or in any kind moral authority position. They are entertainers who get punched for her pleasure, and will likely live out their days mentally debilitated from the repeated blows to the head.
Do we get to comb deeply through this authors personal history and determine all the groups she owes apologies or explanations to? Why not? As an opinion maker in a national news paper and instructor of young people, she has far, far more influence on Canadians than two ignorant punchies. The arrogance of these pseudo-intellectual academics is astounding. Since they are so enlightened and pure, YOU owe THEM an explanation and an apology as to why you're so dumb and ignorant.
Ground Spans: [dumb]

BT-B-SP	[]
BT-B-TC	[dumb, ignorant]
BT-LC	[dumb, ignorant]
BT-MSP	[dumb]
BT-SP	[]
BT-TC	[dumb, ignorant]
BT-SP-TC	[dumb and ignorant]
RBTa-TC(3,∩)	[dumb, ignorant]
RBTa-TC∩SBT-SP(3,∪)	[dumb, ignorant]
SBT-SP(3,∩)	[]
RBTa-TC(3,∪)∩SBT-SP	[]
RBTa-TC(3,∪)	[go, dumb, ignorant]
RBTa-TC∪ SBT-SP(3,∪)	[dumb and ignorant]
SBT-SP(3,∪)	[dumb and ignorant]
RBTa-TC(3,∪)∪SBT-SP	[go, dumb, ignorant]
RNNSL	[ignorant, dumb, ignorant]
RNNSL-TT	[goon, ignorant, dumb, ignorant]
RBTa-B-SP	[]
RBTa-B-TC	[dumb]
RBTa-LC	[on, ignorant, dumb, ignorant]
RBTa-MSP	[]
RBTa-SP	[]
RBTa-TC	[dumb, ignorant]
RBTa-SP-TC	[ignorant, dumb and ignorant]
RBTa-TC-TT	[dumb, ignorant]
SpaCy	[ignorant]
SBT-LC	[ignorant, dumb, ignorant]
SBT-MSP	[dumb and ignorant]
SBT-SP	[]
SBT-SP-TT	[dumb and ignorant]
SBT-TC	[ignorant, dumb, ignorant]
SBT-SP-TC	[ignorant, dumb and ignorant]
TBT-SP	[]
TBT-TC	[ignorant]
TRBTa-SP	[]
TRBTa-TC	[dumb, ignorant]

Table 6: The prediction output of the models for an example in the test set.

than *'pathetic troll'*. This is in contrast with Figure 2b where the Token Detection model misses out on space (because it only considers tokens) and focuses more on the words *'pathetic', 'troll'*. However, the word 'shrink' seems to be important in both cases. This means that while Token Classification models perform better, there are cases which are missed by these approaches. Additionally, some words outside of the span may contribute to toxicity of a particular span. We will be analyzing such words in a future work.

C Model Predictions

The predictions of the various systems for one example that is present in the test set, are listed in Table 6. The examples provide the following intuition about the data and the systems:

- The spaces in between the words are, predictably, ignored by the the token based models. Moreover, the conjunctives like 'and' are ignored as well. This means that additional post-processing of the data will lead to improvements in performance of token classification systems.
- Sometimes, random words like 'go' and 'on' are selected to be toxic, which means that these types of prepositions and verbs can be removed by exact matching in the string, unless they form parts of larger spans.
- The best checkpoints of the span-based models tend to predict empty spans for the selected example. However, when using checkpoint ensembling, we see that union models return accurate spans.
- The ground spans are not entirely correct and are ambiguous. For example, it is not clear whether the word 'ignorant' should be considered to be toxic. The models, based on other examples, predict 'ignorant' to be toxic, but it is not present in the ground spans. This means that finding the toxic spans is not a trivial task for humans, and annotation can not be performed easily by crowd-workers.
- In some cases, one of the occurrences of the word 'ignorant' is considered to be toxic, while the other is predicted to be benign. The first instance of 'ignorant' does not seem to be as toxic as the second instance and therefore, more analysis needs to be done to determine the 'degree' of toxicity of the spans. This can be a good direction for future research.