LZ1904 at SemEval-2021 Task 5: Bi-LSTM-CRF for Toxic Span Detection using Pretrained Word Embedding

Liang Zou
Department of Mathematics
New York University
lz1904@nyu.edu

Wen Li
Department of Linguistics
Indiana University
wl9@indiana.edu

Abstract

Recurrent Neural Networks (RNN) have been widely used in various Natural Language Processing (NLP) tasks such as text classification, sequence tagging, and machine translation. Long Short Term Memory (LSTM), a special unit of RNN, has the advantage of memorizing past and even future information in a sentence (especially for bidirectional LSTM). In the shared task of detecting toxic spans in texts, we first apply pretrained word embedding (GloVe) to generate the word vectors after tokenization. Then we construct Bidirectional Long Short Term Memory-Conditional Random Field (Bi-LSTM-CRF) model by Baidu research to predict whether each word in the sentence is toxic or not. We tune hyperparameters of dropout rate, number of LSTM units, embedding size with 10 epochs and choose the epoch with best validation recall. Our model achieves an F1 score of 66.99% on test dataset.

1 Introduction

Detecting toxic words plays a critical role in social media to ensure healthy online discussions. In previous study, some tasks (Liu et al., 2019; Borkan et al., 2019a) only identify offensive language based on the whole sentence or post. Most of them do not detect specific spans of words that make the sentence or post offensive.

In SemEval-2021 Task 5: Toxic Spans Detection (Pavlopoulos et al., 2021), the data was collected from civil comments (Borkan et al., 2019b). Each post is in string format, and a word is marked as toxic span in the form of its characters' offsets in the string. The goal of the task is to classify whether each word in a sentence is toxic or not. If so, the indices of characters in the word should be returned. The task is evaluated by F1 score based on the character offsets among all posts.

The challenges of this task include:

- The small dataset makes it very difficult to train complicated models like deep neural networks, since it may cause overfitting.
- We need to predict which word or phrase is toxic given a text (many-to-many) rather than whether the entire sentence is offensive or not (many-to-one). This creates restrictions on feature engineering and modeling:
  - Feature Engineering: We cannot delete or add words in the sentence.
  - Modeling: Models need to be specific on each word instead of sentiment classification on whole sentence.
- Most of the words and phrases in sentences are not toxic. This indicates our dataset is imbalanced.

The models we explore in this task include word-based Conditional Random Field (CRF), word-based Bidirectional Long Short Term Memory (Bi-LSTM) with and without pretrained word embedding, Bidirectional LSTM-CRF with pretrained word embedding. We choose Bi-LSTM-CRF as final submission, since it performs the best during our experiments.

The structure of this paper is organized as follows:

- In section 2, we review related work of applications of different models in Sequence Tagging, Name Entity Recognition, and Sentiment Analysis.
- In section 3, we present the summary statistics of data and how we build models with performance evaluation.
- In section 4, we discuss our model results on validation dataset with key findings.

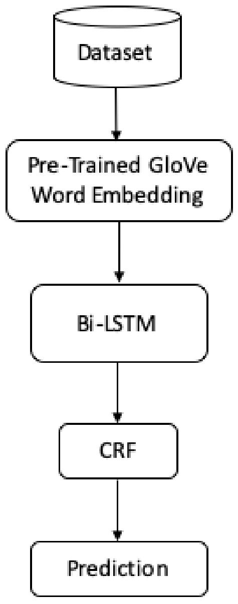

Figure 1: System Flowchart

- In section 5 and 6, we present our conclusion based on experiment results and future work.

2 Related Work

Toxic Span Detection is a type of sequence labeling tasks and is similar to name entity recognition tasks with only two categories. (Xing et al., 2010) surveyed various sequence labeling tasks in terms of methodologies and applications. They also reviewed a few extensions of sequence classification including early classification and semi-supervised learning on sequences. (Nguyen and Guo, 2007) compared different learning algorithms such as Conditional Random Fields (CRF), Support Vector Machine (SVM), and Perceptron for sequence labeling tasks. (Akbik et al., 2018) proposed a new type of embedding called contextual string embedding for sequence labeling tasks. A Comparison between multiple word embedding methods were conducted by (Lauren et al., 2018) for sentiment classification and sequence labeling tasks.

For name entity recognition (NER), (Mansouri et al., 2008) presented a machine learning based approach called Fuzzy Support Vector Machine. Recent advanced deep learning models were summarized by (Yadav and Bethard, 2019) in various shared tasks.

Currently, models that are widely used in various NLP tasks include CRF, LSTM, RNN, and transformers. Also, word embedding such as Word2Vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), and ELMo (Peters et al., 2018) are preferred before model training.

Statistics	Train	Trial
count	7939	690
mean	43	41
std	41	40
min	1	1
25%	16	15
50%	29	28
75%	54	53
max	240	217

Table 1: Distribution of word count

Type	Train	Trial
Toxic	24135	1814
Non toxic	1881225	163786

Table 2: Count of toxic and non toxic words in data

3 Data and Methodology

3.1 Data Description

Toxic Spans Detection Dataset includes trial and training data. The training data contains 7939 records, and trial data contains 690 records. The sentences and indices of characters in toxic span are provided separately. To obtain if a word is toxic or not after tokenization, we split the text by space and punctuation and map the indices of toxicity to corresponding words. As a result, the word sequences in text will be marked as toxic (1) or non-toxic (0).

The distribution of the length of text (in word count) is summarized in Table 1. It shows that training and trial data follow a similar distribution in percentile, mean, and standard deviation (std).

The count of toxic and non-toxic words in texts are concluded in Table 2. It shows the dataset is highly imbalanced that most of words are non toxic.

3.2 Methodology

CRF Conditional Random Field (CRF) was developed in 2001 (Lafferty et al., 2001) for sequence prediction. Given the observable sequence X and labeling sequence Y, the objective of CRF is to construct model for conditional probability $P(Y \mid X)$. An advantage of using CRF compared with other sequential models such as Hidden Markov Model (Rabiner and Juang, 1986) is that it does not rely on the assumption of label independence.

Model	Precision	Recall	F1
CRF	0.686	0.341	0.4556
Bi-LSTM	0.6277	0.3664	0.4627
Bi-LSTM (glove-twitter-50)	0.8571	0.4884	0.6222
Bi-LSTM (glove-twitter-100)	0.8113	0.5	0.6187
Bi-LSTM-CRF (glove-twitter-100)	0.8333	0.5233	0.6429

Table 3: Model evaluations on validation data

Before fitting the model, for each word we create binary features to check whether the word is uppercase, lowercase, titlecase, and digit. We also append the same features of previous and next words. Next, we use "crfsuite" package to build CRF model. We choose "lbfgs" as optimization algorithm, and we set c1 and c2 equal to 0.1, max iteration equal to 100.

Bi-LSTM Long Short Term Memory (LSTM) is one of the most commonly used recurrent neural networks in many natural language processing tasks (Hochreiter and Schmidhuber, 1997). It consists of input gate, output gate, forget gate, and cell. Its gate structure enables the model to memorize long-term dependency and to prevent gradient vanishing issues.

In the experiments, we use Bidirectional LSTM (Bi-LSTM) in tensorflow.keras as second baseline. We configure number of LSTM units to be 200, embedding size equal to 50, and max sequence length to be 240, which is the max sentence length in training dataset. Thus, sentences with length of less than 240 will be padded. To reduce overfitting of neural networks, we set dropout rate as 0.2. Our final output layer uses sigmoid activation function with adam optimizer for gradient descent (Kingma and Ba, 2014).

We set number of epochs equal to 10 and record checkpoints. Since tensorflow package does not contain built-in F1 score, the final model parameters are loaded from the checkpoint with highest validation recall.

Bi-LSTM with pretrained word embedding To further improve model performance, we adopt pretrained word embedding to generate word representation before training Bi-LSTM. The word embedding we use includes glove-twitter-50 and glove-twitter-100 in gensim (Řehůřek and Sojka, 2010). This means we need to modify our embedding size to 50 and 100 respectively. All other hyperparameters are consistent with Bi-LSTM above.

Bi-LSTM-CRF with pretrained word embedding Our final model is Bidirectional LSTM-CRF created by Baidu Research (Huang et al., 2015). Compared with previous Bi-LSTM architecture, we add an extra output layer of CRF to make final predictions (as shown in Figure 1). Accordingly, we replace the loss function of binary cross entropy by CRF loss. We use glove-twitter-100 as pretrained embedding layer. All other hyperparameters of LSTM remain the same.

4 Experiment Results

We split proportion of training and validation dataset into 9:1. The evaluation results on validation set are summarized in Table 3. For each model discussed above, we list its precision, recall, and F1 score with default threshold. Since the dataset is highly imbalanced, we only focus on the evaluation metrics of toxic words. There are several key findings:

- Models with pretrained word embedding perform better than those without pretrained word embedding, since it produces higher precision and recall (thus higher F1 score).
- The performances of pretrained word embedding are close to each other regardless of embedding size. We do not want to further increase the embedding size, since it will increase the training time but not boost the performance significantly.
- As a final output layer, CRF can further improve recall for Bi-LSTM while keeping the precision in the same level. Therefore, it can increase F1 score.

Based on the model evaluation table, we choose Bi-LSTM-CRF with pretrained glove-twitter-100 embedding as our final model. The model achieves an F1 score of 0.6699 in final submission.

The confusion matrix for test data can be found in Table 5. We first flatten the sequence to a list of

Texts	**Predictions**	**Error Type**
Chris Birch is a mean, self-centered, contrary <u>ass</u>. ... always <u>sucks</u> up to Big Oil.	[ass, sucks]	False Positive
I wish this <u>moron</u> would have been shot to death by the US soldier instead of the other way around.	[moron]	False Positive
Lord have Mercy on us, Trump is running **amok**.	[]	False Negative
... They're **vandals**, **thieves**, and bullies.	[]	False Negative

Table 4: Examples of False Positive / Negative

	Actual Pos	**Actual Neg**
Pred Pos	1747	1504
Pred Neg	736	73892

Table 5: Confusion matrix on test data

words before calculation. We define toxic words as positive (Pos) examples and non toxic words as negative (Neg) examples in the matrix. The table shows there are a lot of false positives (1504) in our model, this implies our model may be oversensitive to the toxic words.

To deep dive into the model performance with specific examples, we collect a few sentences from test data in Table 4. In false positive examples marked as underline, the words "ass", "sucks", and "moron" are predicted as toxic words where there exists no toxicity in these sentences. In false negative examples marked as bold, the model fails to identify toxic words like "amok", "vandals", and "thieves". The errors may come from the following reasons:

- Incorrect labels by ground-truth spans. These errors are unavoidable from the model due to human mistake.

- The pre-trained word embedding from GloVe does not reflect sentiment for those words. In other words, these words are not marked as positive or negative but neutral in word embedding.

- The position of words in sentence was not detected as toxicity by our Bi-LSTM-CRF model. For example, one word could be marked as toxic spans when it is in the beginning of the sentence but not the case when it is at the end. This will cause difficulty for model training to detect toxicity.

5 Conclusion

Detecting toxic words in texts is critical to furnish a healthy environment on social media. Sequence labeling task for finding specific offensive words is more difficult than sentiment classification on sentence level, since it requires models to locate the positions or indices of words in sentences. In addition, the task also places restrictions on feature engineering, because we cannot delete or add words in sentences.

Our experiment shows pretrained word embedding can improve model performance compared with randomized embedding weights. This verifies the concept of transfer learning where we can borrow the outputs from other resources and use them as inputs to achieve specific goals. Another finding is the benefit of model stacking where we add an extra layer of CRF after Bi-LSTM that further enhances predictability. In such case, when a single model does not work well in NLP tasks, combining different models with pretrained word embedding can be a good option to explore. However, there are still a lot of false positive examples in test set where the model predicts toxic words that in fact are not toxic.

6 Future Work

Further improvements can focus on feature engineering and model implementation. For feature engineering, we can conduct data augmentation for false negative examples: We first collect the words that are predicted as non-toxic but actually toxic, and reconstruct sentences using those toxic words as more training samples. This method can increase the weights of words that were originally omitted by the model, so that it may return better results. Similarly, we can also collect false positive examples and perform data augmentation to reduce false positive rates.

In addition to data augmentation, one can perform word-level text normalization to transform words of different tenses to the same, even though each word cannot be deleted or added in a sentence.

From the model perspective, we may consider using more advanced classifiers with complicated structures. Due to resource limitations, we cannot design any large neural networks models such as deep neural networks (DNN) or transformers. Most of our experiments are done locally or via Google Colab. Training large neural networks will be very time-consuming and expensive when using tremendous amount of computing resources including multiple GPUs, TPUs.

If we have more time and available resources, we can experiment with more complex models such as BERT (Devlin et al., 2018), GPT (Radford et al., 2018), and so forth. In addition, we can deploy larger LSTM-related architecture including Bi-LSTM-CNN-CRF for sequence labeling (Ma and Hovy, 2016).

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th international conference on computational linguistics*, pages 1638–1649.

Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2019a. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion proceedings of the 2019 world wide web conference*, pages 491–500.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural computation*, 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

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

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data.

Paula Lauren, Guangzhi Qu, Jucheng Yang, Paul Watta, Guang-Bin Huang, and Amaury Lendasse. 2018. Generating word embeddings from an extreme learning machine for sentiment analysis and sequence labeling tasks. *Cognitive Computation*, 10(4):625–638.

Ping Liu, Wen Li, and Liang Zou. 2019. NULI at SemEval-2019 task 6: Transfer learning for offensive language detection using bidirectional transformers. In *Proceedings of the 13th international workshop on semantic evaluation*, pages 87–91.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via Bi-directional LSTM-CNNs-CRF. *arXiv preprint arXiv:1603.01354*.

Alireza Mansouri, Lilly Suriani Affendey, and Ali Mamat. 2008. Named entity recognition approaches. *International Journal of Computer Science and Network Security*, 8(2):339–344.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Nam Nguyen and Yunsong Guo. 2007. Comparisons of sequence labeling algorithms and extensions. In *Proceedings of the 24th international conference on Machine learning*, pages 681–688.

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*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

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*.

Lawrence Rabiner and Biinghwang Juang. 1986. An introduction to hidden Markov models. *ieee assp magazine*, 3(1):4–16.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA.

Zhengzheng Xing, Jian Pei, and Eamonn Keogh. 2010. A brief survey on sequence classification. *ACM Sigkdd Explorations Newsletter*, 12(1):40–48.

Vikas Yadav and Steven Bethard. 2019. A survey on recent advances in named entity recognition from deep learning models. *arXiv preprint arXiv:1910.11470*.