Winners at W-NUT 2020 Shared Task-3: Leveraging Event Specific and Chunk Span features to Extract COVID Events from tweets

Ayush Kaushal and **Tejas Vaidhya**
Indian Institute of Technology, Kharagpur
ayushk4@gmail.com, iamtejasvaidhya@gmail.com

Abstract

Twitter has acted as an important source of information during disasters and pandemic, especially during the times of COVID-19. In this paper, we describe our system entry for *WNUT 2020 Shared Task-3*. The task was aimed at automating the extraction of a variety of COVID-19 related events from Twitter, such as individuals who recently contracted the virus, someone with symptoms who were denied testing and believed remedies against the infection. The system consists of separate multi-task models for slot-filling subtasks and sentence-classification subtasks while leveraging the useful sentence-level information for the corresponding event. The system uses COVID-Twitter-Bert with attention-weighted pooling of candidate slot-chunk features to capture the useful information chunks. The system **ranks 1st** at the leader-board with **F1 of 0.6598**, without using any ensembles or additional datasets. The code and trained models are available at this https url[1].

1 Introduction

The World Health Organization declared COVID-19, a global pandemic on March 11, 2020. As of 2020/09/21, there are over 30 million cases[2] and 900,000 deaths due to the infection. With the imposed lockdown, work from home and physical distancing, social media like twitter saw an increased usage. A large part of the use was posting and consuming information on the novel infection. These information include potential reasons for contraction of the disease, such as via exposure to a family member who tested positive, or someone who is showing COVID symptoms but was denied testing. Accompanying to the pandemic was an infodemic of misinformation about COVID-19, including fake remedies, treatments and prevention-suggestions in social media (Alam et al., 2020).

Zong et al. (2020) show the possibility to automatically extract structured knowledge on COVID-19 events from Twitter and released a dataset of COVID related tweets across 5 event types. We used this dataset in our experiments for the shared-task. These tweets are annotated for whether they belong to an event (we refer to this as the **event-prediction** task in this paper) and their event-specific questions (factual or opinion). We identify these event-specific questions into two types of subtasks, **slot-filling** and **sentence classification**.

Our system consists of separate multi-task models for slot-filling subtasks and sentence-classification subtasks. Our contribution comprises improvement upon the baseline (mentioned in section 2) in three ways:

- We incorporate the event-prediction task as auxiliary subtask and fuse its features for all the event-specific subtasks.
- We perform an attention-weighted pooling over the candidate chunk span enabling the model to attend to subtask specific cues.
- We use the domain-specific Bert of Covid-Twitter Bert (Müller et al., 2020).

2 Related Works

Sentence classification tasks (such as opinion or sentiment mining) as well as slot-filling tasks have greatly progressed with deep learning advancements such as LSTM (Hochreiter and Schmidhuber, 1997), Tree-LSTM (Tai et al., 2015) and transfer learning over pre-trained models (Peters et al., 2018; Howard and Ruder, 2018; Devlin et al., 2019). Among these, CT-Bert outperforms others on COVID related twitter tasks (Müller et al., 2020). Taking inspiration from the same, we use

[1] https://github.com/Ayushk4/extract_covid_entity
[2] https://coronavirus.jhu.edu/map.html

Proceedings of the 2020 EMNLP Workshop W-NUT: The Sixth Workshop on Noisy User-generated Text, pages 522–529
Online, Nov 19, 2020. ©2020 Association for Computational Linguistics

CT-Bert as part of our architecture. A variety of slot-filling approaches have been built on top of these deep learning advancements (Kurata et al., 2016; Qin et al., 2019). The proposed baseline for our task (Zong et al., 2020) modifies Bert model for slot-filling problem inspired by Baldini Soares et al. (2019). Due to the excellent performance offered by Bert (Devlin et al., 2019) and Baldini Soares et al. (2019), we build upon this baseline approach.

Extraction of structured knowledge from tweets pertaining of events (Benson et al., 2011) has been studied for disaster and crises management (Abhik and Toshniwal, 2013; Rudra et al., 2018) and in pandemic scenarios (Al-Garadi et al., 2016). Extracting such entities can be useful for epidemiologists, deciding policies and preventing spread (Al-Garadi et al., 2016; Zong et al., 2020).

Due to the fast-spreading nature of the infection, it is also difficult to manually trace the spread of the pandemic. However, with twitter event-specific entity extraction and Geo-location, one could potentially build a real-time pandemic surveillance system (Lwowski and Najafirad, 2020; Al-Garadi et al., 2020). Bal et al. (2020) show that health-issues related misinformation is prevalent in social media, while Alam et al. (2020) talks about covid-specific misinformation. Such systems for extracting structured knowledge over the tweets talking about potential cures for COVID will help study how users perceive the COVID misinformation.

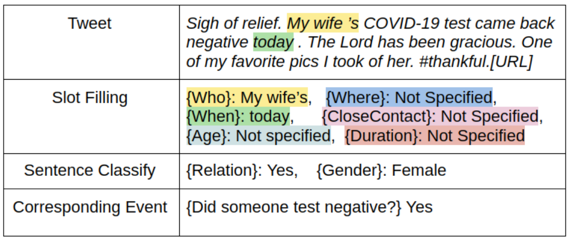
In §3, we describe the dataset and the problem statement. Then in §4, we discuss the details of our two multi-task models followed by experiments, results and conclusion.

3 Dataset and Problem statement

Now, we will briefly go over the dataset. The reader may refer (Zong et al., 2020) for full details. Each of the 7500 tweets in the dataset belongs to one of the 5 event types: tested-positive, tested-negative, can-not-test, death, and cure. The first four events aimed at extracting structured reports of coronavirus related events, such as self-reported cases or news stories about public figures who were exposed to the virus. Each tweet was first annotated for whether it belongs to its respective event (e.g. Is the tweet belonging to the tested-positive event talking about someone who tested positive?). Throughout this paper, we refer to this as the **Event-Prediction** task. The tweets that correspond to its event were then annotated for event-specific questions or **subtasks** about factual information and user's opinions. All annotations are done by multiple Amazon Mechanical Turks with inter-annotation agreement. The event-specific questions or subtasks (e.g. name, age, gender of the person tested positive) varies depending on the event. These subtasks are of two categories: **slot-filling** (e.g., Who tested positive/negative?, Where are they located?, Who is in close contact with person contracting the disease?) and **sentence classification** (e.g. Is author related to infected person?, Does the author experience any symptoms?, Does the author believe a cure method is effective?).

Event	# Tweets
Tested positive	2397
Tested negative	1144
Can Not Test	1128
Death	1231
Cure/Prevention	1244
Total	7144

Table 1: Dataset statistics, scraped during early July.



Tweet	*Sigh of relief. My wife 's COVID-19 test came back negative today . The Lord has been gracious. One of my favorite pics I took of her. #thankful.[URL]*
Slot Filling	{Who}: My wife's, {Where}: Not Specified, {When}: today, {CloseContact}: Not Specified, {Age}: Not specified, {Duration}: Not Specified
Sentence Classify	{Relation}: Yes, {Gender}: Female
Corresponding Event	{Did someone test negative?} Yes

Figure 1: An example tweet from tested negative event.

The dataset released tweet IDs and their annotations. We obtain our text corresponding to tweets using the official Twitter API[3]. Table 1 shows the statistics for the dataset we scrapped in early July.[4] Figure 1 shows an annotated example from the dataset. We identify the event-specific subtasks into two categories shown in Table 2.

We now formally describe the two types of event-specific subtasks:

Slot-filling subtasks: Assume n slot-filling subtasks $\{S_1, S_2 ... S_n\}$. We set up each slot-filling subtask S_i as a supervised binary classification problem. Given the tweet t and the candidate slot s, the model $f(t, s) \rightarrow \{0, 1\}$ predicts whether s answers its designated question. We extract a list of

[3] https://developer.twitter.com/

[4] We get about 350 fewer tweets than the corpus. Some tweets are not obtainable over time as the accounts/tweets get deleted, renamed, banned, or change-visibility etc.

Event	Sentence Classification	Slot-Filling task
Tested positive	gender, relation	who,age,recent-visit,when,where,employer,c.-contact
Tested negative	gender, relation	who,age,when,where,duration,close-contact
Can Not Test	relation, symptoms	who,when,where
Death	relation, symptoms	who,age,when,where
Cure	opinion	what is the cure, who is promoting cure

Table 2: The proposed event-specific subtasks split into two subtask types: slot-filling and sentence classification

candidate slot of all noun chunks and name entities in each of the tweets by using a Twitter tagging tool (Ritter et al., 2011) same as the baseline.

Sentence classification subtasks: Assume m sentence classification subtasks $\{C_1, C_2...C_m,\}$. Given a sentence classification subtask C_i aims to learn a model $g(t) \rightarrow \{l_1, l_2...l_k\}$, where t is a tweet and l_j is a label. Here the number of labels can vary depending on the subtask, for example, gender is labelled with {Male, Female, Others/Not Specified}, Relation with {Yes, No}, Opinion with {effective, no cure, not effective, no opinion} and so on. All these subtasks are 'supervised' classification problems.

The dataset is also annotated with whether a tweet corresponds to its respective event or not. We treat this as an additional **Event-Prediction task**. This is a binary classification task that aims to learn a model $h(t) \rightarrow 0, 1$ where t is a tweet.

4 Approach

In the following subsections §4.1 and §4.2, we describe our multi-task model for slot-filling and sentence-classification respectively.

4.1 Slot-filling

We improve upon the baseline (Zong et al., 2020) by using domain-specific Bert, using attention-weighted pooling over the candidate chunk feature sequence, incorporating auxiliary Event-Prediction task and utilizing its logits for all the slot-filling subtasks. Before describing the approach, we first describe the Bert baseline. Our slot-filling model can be seen in figure 2.

The baseline consists of Bert based classifier. It takes a tweet t as input and encloses the candidate slot s, within the tweet, inside special entity start $< E >$ and end $< /E >$ markers. The Bert hidden representation of token $< E >$ is then processed through a fully connected layer with softmax activation to make the binary prediction for a task (Baldini Soares et al., 2019). Since many slot-filling tasks within an event are semantically related to each other, they jointly trained the final softmax layers of all the subtasks S_i in an event by sharing their Bert model parameters.

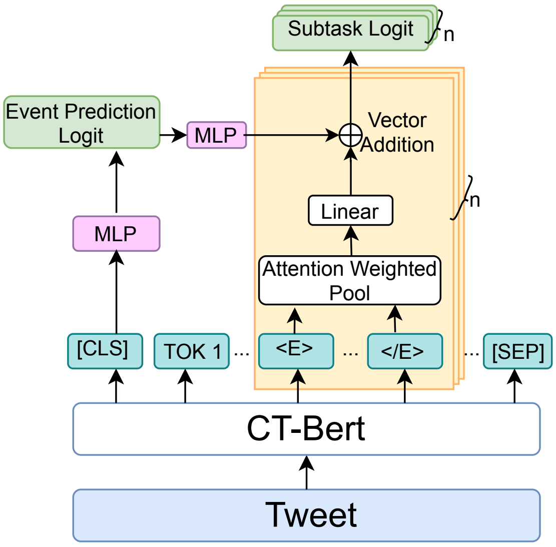


Figure 2: Slot-Filling Model, described in Section §4.1. Here n is the number of slot-filling subtasks.

COVID Twitter Bert (CT-Bert) is a Bert-Large model pretrained on Twitter Corpus on COVID-19 topics, leading to marginal improvements from Bert on tasks based on Twitter datasets(Müller et al., 2020). This motivates us to use CT-Bert instead of Bert from the baseline model.

The baseline, uses the Bert hidden representation of token $< E >$ for classification. Here, however, we use attention-weighted pool of the CT-Bert hidden representation of tokens between $< E >$ and $< /E >$ (both inclusive). Formally, let $\{x_0, ...x_p, ...x_q, ...x_n\}$ be the output vectors from the hidden representation of CT-Bert where p and q are indices of $< E >$ and $< /E >$ respectively, then for any of the slot-filling subtask S_j, we get its pooled vector as follows:

$$\tilde{x}^{S_j} = \sum_{i=p}^{q} \alpha_i^{S_j} x_i \tag{1}$$

$$\alpha_i^{S_j} = Softmax_{p\ to\ q}(x_i^T a^{S_j})$$

where ${x_i}^T$ denotes the transpose of x_i, a^{S_j} is a trainable vector. The motivation for attention weighted pooling is that depending on the task, model can attend to different portions of the candidate slot chunk. Next we obtain the binary classification score vector:

$$h^{S_j} = W^{S_j} \widetilde{x}^{S_j} + b^{S_j} \tag{2}$$

Here W^{S_j} and b^{S_j} are trainable parameters.

We treat the Event-Prediction task as an auxiliary task and then fuse its logits to each of the other slot-filling subtasks. The motivation is that a task-specific entity shall be present in a tweet only if the tweet belongs to its respective event.

To predict the label for Event-Prediction task, we take the CT-Bert features of $[CLS]$ token and pass it through a MultiLayer Perceptron (MLP) to get logits h_{ces}.

We fuse h_{ces} prediction over each subtasks S_j by adding it to h^{S_j} (from (2)) to get the logits $h_f^{S_j}$:

$$h_f^{S_j} = h^{S_j} + MLP^{S_j}(h_{ces}) \tag{3}$$

In practice, we share the parameters of the MLP^{S_j} across all the slot-filling subtasks S_j.

Given a tweet t and slot s, our loss for slot-filling model over n slot-filling subtasks $\{S_1, S_2...S_n\}$ and Event-Prediction task looks like:

$$\begin{gathered} Loss(t, s, y_{ces}, (y_1, y_2...y_n)) \\ = \lambda_1 CE_{Loss}(h_{ces}, y_{ces}) + \sum_{k=1}^{n} CE_{Loss}(h_f^{S_k}, y_k) \end{gathered} \tag{4}$$

where CE_{loss} is softmax cross entropy loss, y_{ces} is ground truth label for Event-Prediction task and $(y_1, y_2...y_n)$ are the labels for the candidate slot s of tweet t for the subtasks $\{S_1, S_2...S_n\}$. We keep λ_1 = 1.

Our preprocessing for this is same as baseline.

4.2 Sentence classification

Our Sentence classification model is shown in figure 3. We use a Bert based sentence classifier and improve it by using CT-Bert, incorporating the auxiliary Event-Prediction task and attention-weighted pooling over the entire sequence.

This model uses CT-Bert instead of Bert and the auxiliary Event-Prediction task for same reason as the slot-filling model.

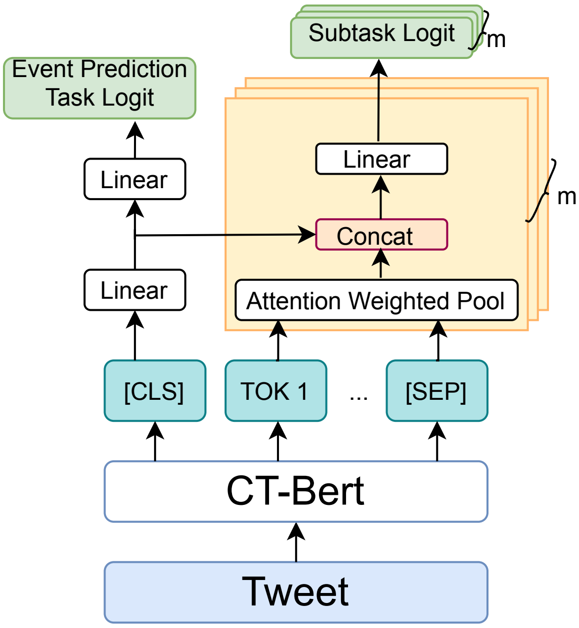


Figure 3: Sentence Classification model, described in section. §4.2. Here m is the number of Sentence Classification subtasks.

An attention-weighted pooling is done over the feature sequences from CT-Bert to extract the most relevant information. Formally, let $\{x_0, x_1, ...x_n\}$ be the output vectors from CT-Bert (here 0 and n are indices of $[CLS]$ and $[SEP]$ respectively). Then for any of the sentence classification subtask C_j, we get its pooled vector $\widetilde{x}^{C_j}$ as follows:

$$\widetilde{x}^{C_j} = \sum_{i=0}^{n} \beta_i^{C_j} x_i \tag{5}$$

$$\beta_i^{C_j} = Softmax_i(x_i^T a^{C_j} + c^{C_j})$$

where a^{C_j}, c^{C_j} are trainable vector and scalar respectively.

For the Event-Prediction task, we take the CT-Bert vector representation of $[CLS]$ token and pass it through a MLP. Assume the MLP's final and hidden states to be v_{ces} and h'_{ces}.

Next, we incorporate information from Event-Prediction task into sentence classification subtask C_j. Since the sentence classification subtasks aren't binary classification, so, unlike the slot-filling model, we cannot merely add the Event-Prediction logits to all tasks. Additionally, we desire sentence-level event specific features for each of the sentence level predictions. Hence, we concatenate the hidden state features from the MLP of Event-Prediction task h'_{ces} to pooled vector $\widetilde{x}^{C_j}$

from 5 to get the logits $h_f^{C_j}$ for each subtask C_j, as follows:

$$h_f^{C_j} = [\widetilde{x}^{C_j}; h'_{ces}]^T W^{C_j} + b^{C_j} \tag{6}$$

Here ^{T} denotes transpose, $[;]$ denotes vector concatenation. W^{C_j} and b^{C_j} are trainable.

Given a tweet t, our loss for sentence classification model over m sentence classification subtasks $\{C_1, C_2 ... C_m\}$ and Event-Prediction task is:

$$Loss(t, y_{ces}, (y_1, y_2 ... y_m))$$

$$= \lambda_2 CE_{Loss}(v_{ces}, y_{ces}) + \sum_{k=1}^{m} CE_{Loss}(h_f^{C_k}, y_k) \tag{7}$$

where CE_{Loss} is softmax cross entropy loss, y_{ces} is ground truth label for Event-Prediction task and $(y_1, y_2 ... y_m)$ are the labels for tweet t for the subtasks $\{C_1, C_2 ... C_m\}$. We keep λ_2 = 1.

Preprocessing for sentence classification is done using ekphrasis library (Baziotis et al., 2017). We remove Emoji, URL, Email, punctuation and normalize text by word segmenting, lower-casing and word decontraction.

5 Experiments

All the experiments were performed using PyTorch (Paszke et al., 2019) and Hugging Face's transformers (Wolf et al., 2019). We use git and wandb (Biewald, 2020) for experiment tracking. Optimization is done using Adam (Kingma and Ba, 2014) with a learning rate of 2e-5. Slot-filling models are trained for 8 epochs and sentence classification model for 10 epochs. Average training time per epoch on Tesla P100 is ≈ 4 minutes for slot-filling, and ≈ 30 second for sentence classification.

We use a 70-30 split for train-valid set. The valid set is used to obtain the best threshold for each of the slot classification tasks over the grid $\{0.1, 0.2, ..., 0.9\}$. We exclude labels with "No consensus" from our data.[5]

All the MLP have 1 hidden layer and 0.1 dropout. MLP_{S_j} has 4 hidden size, LeakyReLU activation (Maas et al., 2013) with 0.1 negative slope, rest of the MLP have 50 hidden size and Tanh activation.

Event	F1	P	R
Tested Positive	.68	.80	.58
Tested Negative	.66	.66	.67
Can Not Test	.65	.67	.64
Death	.69	.72	.67
Cure/Prevention	.63	.75	.53
Overall	**.66**	**.73**	**.60**

Table 3: Micro averaged scores on the held out test set for our final submission.

6 Results

Our performance on the held-out test set is shown in Table 3. Our system **ranks 1st position** in the W-NUT 2020 Shared Task-3 (Zong et al., 2020). We also independently rank 1st for 3 of the 5 events: 'Can Not Test', 'Death', and 'Cure'.

Now we discuss our various experiments.

Slot-filling: We experimented with a variety of architectures for slot-filling model. **Our (SF)** is our Slot-Filling Model from §4.1. **Our (SF) w/o pool** is our slot-filling model that uses the CT-Bert hidden representation of token $< E >$ to classify instead of doing an attention-weighted pooling. **Our (SF) w/o CES** is our slot-filling model without Event-Prediction task. **CT-Bert** and **Bert-large** are baseline models using CT-Bert and Bert-large instead of Bert-base.

Table 4 shows the performance of these models. There is a considerable performance difference by using CT-Bert instead of Bert, demonstrate the benefits of domain specific pre-training. *Our (SF) w/o pool* and *Our (SF) w/o CES* outperform CT-Bert demonstrating the importance of Event-Prediction task and attention-weighted pooling over slot-chunk respectively. *Our (SF)* using CT-Bert with Event-Prediction and attention-weighted pooling performs the best among these models.

Sentence level tasks: We experimented with various architectures for sentence level tasks. **Our (SC)** is our Sentence Classification architecture from §4.2. **Our (SC) w/o CES** is our Sentence Classification without Event-Prediction task. **Bert multitask** model predicts using the $[CLS]$ representation from Bert (Devlin et al., 2019). We also build an LSTM model (Hochreiter and Schmidhuber, 1997) with GloVe embedding (Pennington et al., 2014), and twitter-tokenization using Word-

[5] As per the submission guidelines, some subtasks like opinion had their label classes merged. We incorporate these changes in our model.

Model	Micro F1	Macro F1
Our (SF)	**.684**	**.558**
Our (SF) w/o pool	.678	.557
Our (SF) w/o CES	.665	.552
CT-Bert	.662	.551
Bert (large)	.610	.529
Bert (baseline)	.612	.528

Table 4: Results of slot-filling models on our 70-30 split. We report results on the valid set across *all slot filling subtasks* across the 5 events.

Model	Micro F1	Macro F1
Our (SC)	**.788**	**.767**
Our (SC) w/o CES	.777	.731
CT-Bert multitask	.760	.717
Bert multitask	.715	.612
LSTM multitask	.614	.543

Table 5: Results sentence classification models on our 70-30 split. We report results on the valid set across *all sentence classification subtasks* across the 5 events.

Tokenizers package (Kaushal et al., 2020).

Table 5 shows the performance of these architectures. Our (SC) outperforms others on macro F1 and micro F1, followed by Our (SC) w/o CES. The performance difference between these two, shows the benefits of including the Event-Prediction task. While the performance difference between CT-Bert multitask and Our (SC) w/o CES shows the gains from attention weighted pooling. CT-Bert also outperforms Bert multitask, showing its usefulness in our proposed system over using Bert. Lastly, Bert multitask, and all the models using Bert/CT-Bert outperform LSTM by a very large margin demonstrating the superiority of these pretrained language models.

Separate Sentence classification and slot filling models: Consider **Bert separate**, a simple system treating the two categories of tasks separately. It has the Bert baseline as its slot filling model and a simple Bert sentence classifier using features from $[CLS]$ for sentence prediction. Bert separate does not have the event-prediction auxilliary task or any attention weighted pooling. Table 6 shows the performance of *Bert separate* against the baseline. *Bert separate* outperforms the Bert baseline by a considerable margin, thus showing the importance of treating the two subtasks differently.

Model	Micro F1	Macro F1
Bert Separate	**.631**	**.545**
Bert Baseline	.608	.512

Table 6: Results comparing the systems treating the sentence classification and slot-filling subtasks separately vs those treating it similarly. We report results on the valid set across *all the subtasks* of both categories across the 5 events.

7 Conclusion and Future Work

In this paper, we presented our system that bagged 1st position in the WNUT-2020 Shared Task-3 on Extracting COVID Entities from Twitter. We divided the event-specific subtasks into slot-filling and sentence classification subtasks, building separate architectures for the two. For both architectures, we used COVID-Twitter Bert, weighted-attention pooling over chunk-spans/sentence and fused logits and features from auxiliary Event-Prediction task. Our ablation studies demonstrated the usefulness of each component in our system.

There is a lot of scope of improvement for subtasks with few positive labels. Pretraining on relevant data (such as COVID-misinformation datasets for event cure) is a promising direction.

Another direction would be to reduce the training and inference time of slot-filling model by *not* enclosing the candidate chunk within special start $< E >$ and special end $< /E >$ tokens. We can instead use the attention-weighted pooling over candidate slot chunks. This will reduce the number of Bert forward passes from $O(k)$ to $O(1)$, where k is the number of candidate chunks in a tweet.

Acknowledgments

We are very grateful for the invaluable suggestions given by Nikhil Shah, Dibya Prakash Das and Sayan Sinha. We also thank the organizers of the Shared Task-3 at WNUT, EMNLP-2020.

References

Dhekar Abhik and Durga Toshniwal. 2013. Sub-event detection during natural hazards using features of social media data. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 783–788.

Mohammed Al-Garadi, Muhammad Khan, Kasturi Varathan, Ghulam Mujtaba, and Abdelkodose Abdulla. 2016. Using online social networks to track a pandemic: A systematic review. *Journal of Biomedical Informatics*, 62.

Mohammed Ali Al-Garadi, Yuan-Chi Yang, Sahithi Lakamana, and Abeed Sarker. 2020. Text classification approach for the automatic detection of twitter posts containing self-reported covid-19 symptoms.

Firoj Alam, Fahim Dalvi, Shaden Shaar, Nadir Durrani, Hamdy Mubarak, Alex Nikolov, Giovanni Da San Martino, Ahmed Abdelali, Hassan Sajjad, Kareem Darwish, and Preslav Nakov. 2020. Fighting the covid-19 infodemic in social media: A holistic perspective and a call to arms.

Rakesh Bal, Sayan Sinha, Swastika Dutta, Risabh Joshi, Sayan Ghosh, and Ritam Dutt. 2020. Analysing the extent of misinformation in cancer related tweets. *Proceedings of the International AAAI Conference on Web and Social Media*, 14(1):924–928.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy. Association for Computational Linguistics.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. DataStories at SemEval-2017 task 4: Deep LSTM with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754, Vancouver, Canada. Association for Computational Linguistics.

Edward Benson, Aria Haghighi, and Regina Barzilay. 2011. Event discovery in social media feeds. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 389–398, Portland, Oregon, USA. Association for Computational Linguistics.

Lukas Biewald. 2020. Experiment tracking with weights and biases. Software available from wandb.com.

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.

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

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Ayush Kaushal, Lyndon White, Mike Innes, and Rohit Kumar. 2020. Wordtokenizers.jl: Basic tools for tokenizing natural language in julia. *Journal of Open Source Software*, 5(46):1956.

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

Gakuto Kurata, Bing Xiang, Bowen Zhou, and Mo Yu. 2016. Leveraging sentence-level information with encoder LSTM for semantic slot filling. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2077–2083, Austin, Texas. Association for Computational Linguistics.

Brandon Lwowski and Peyman Najafirad. 2020. Covid-19 surveillance through twitter using self-supervised learning and few shot learning.

Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. 2013. Rectifier nonlinearities improve neural network acoustic models. In *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*.

Martin Müller, Marcel Salathé, and Per E Kummervold. 2020. Covid-twitter-bert: A natural language processing model to analyse covid-19 content on twitter.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8026–8037. Curran Associates, Inc.

Jeffrey Pennington, Richard Socher, and Christopher 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, Doha, Qatar. Association for Computational Linguistics.

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, Volume 1 (Long Papers)*, pages

2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. A stack-propagation framework with token-level intent detection for spoken language understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2078–2087, Hong Kong, China. Association for Computational Linguistics.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Koustav Rudra, Pawan Goyal, Niloy Ganguly, Prasenjit Mitra, and Muhammad Imran. 2018. Identifying sub-events and summarizing disaster-related information from microblogs. In *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '18, page 265–274, New York, NY, USA. Association for Computing Machinery.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566, Beijing, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Shi Zong, Ashutosh Baheti, Wei Xu, and Alan Ritter. 2020. Extracting covid-19 events from twitter.