

UCD-CS at W-NUT 2020 Shared Task-3: A Text to Text Approach for COVID-19 Event Extraction on Social Media

Congcong Wang

School of Computer Science
University College Dublin
Dublin, Ireland

`congcong.wang@ucdconnect.ie`

David Lillis

School of Computer Science
University College Dublin
Dublin, Ireland

`david.lillis@ucd.ie`

Abstract

In this paper, we describe our approach in the shared task: *COVID-19 event extraction from Twitter*. The objective of this task is to extract answers from COVID-related tweets to a set of predefined slot-filling questions. Our approach treats the event extraction task as a question answering task by leveraging the transformer-based T5 text-to-text model.

According to the official evaluation scores returned, namely F1, our submitted run achieves competitive performance compared to other participating runs (Top 3). However, we argue that this evaluation may underestimate the actual performance of runs based on text-generation. Although some such runs may answer the slot questions well, they may not be an exact string match for the gold standard answers. To measure the extent of this underestimation, we adopt a simple exact-answer transformation method aiming at converting the well-answered predictions to exactly-matched predictions. The results show that after this transformation our run overall reaches the same level of performance as the best participating run and state-of-the-art F1 scores in three of five COVID-related events. Our code is publicly available to aid reproducibility¹.

1 Introduction

Since the outbreak of COVID-19, a wide variety of research has been conducted to mine insights or gain rapid access to information in relation to the crisis. For example, the TREC-COVID challenge seeks advanced techniques for finding useful information from the COVID-19 corpus comprising hundreds of thousands of COVID-related academic articles (Wang et al., 2020; Roberts et al., 2020). Related research focuses on mining valuable information from social media (Müller et al.,


¹https://github.com/wangcongcong123/ttt/tree/master/covid_event

2020; Dimitrov et al., 2020). This is largely inspired by the fact that social media motivates people to share information or express their opinions quickly. However, this user-generated content is usually noisy and enormous during crises, and thus needs to be condensed and filtered before further processing and analysis.

Motivated by this, the ACL Workshop on Noisy User-generated Text 2020 proposed a shared task (W-NUT Task-3) of *extracting COVID-19 related events from Twitter* (Zong et al., 2020). The major objective of this task is to seek computational linguistic techniques for extracting text spans from a corpus of raw tweets to answer a set of predefined slot questions. The corpus used in the task can be described in two parts. First, the corpus consists of approximately 7,500 tweets categorised into five broad event types: (1) TESTED POSITIVE, (2) TESTED NEGATIVE, (3) CAN NOT TEST, (4) DEATH and (5) CURE AND PREVENTION. As the names indicate, these tweets are a summary of people’s primary concerns about COVID-19 that they likely post about on social media. Secondly, for the tweets in each event, a set of questions or slot-filling types are defined to help gather more fine-grained information about the tweets. The human annotations, (i.e., ground truths) of the corpus are simply the answers to the predefined questions. Figure 1 illustrates an example from this corpus. This example represents a TESTED POSITIVE or TESTED NEGATIVE event, and the associated slot-filling questions relate to the “who” and the “duration” slots, asking who has tested positive or negative and how long it takes to know the test result. The annotation process in part 2 is conducted by annotators who select answers from a drop-down list of candidate choices². This ex-

²The choices are automatically-extracted text spans obtained through a Twitter tagging tool (Ritter et al., 2011) or predefined choices such as “not specified”, “yes”, “no”, etc.

Event type: TESTED POSITIVE or TESTED NEGATIVE

 *Prince Charles tests positive for Corona
Prince William knowing he's the next
in line to the throne: <https://t.co/B1nmlPlj69>.

Slot "who": Who is tested positive (negative)?
label: *Prince Charles

Slot "duration": How long does it take to get to
know the test results?
label: Not Specified

Figure 1: An example of the event extraction task: this shows a tweet represents a TESTED POSITIVE or TESTED NEGATIVE event. The objective is to extract answers to the slot questions concerning the event.

plains why the label for slot “who” in Figure 1 has the symbol * at the beginning.

Using this corpus, we leverage a text-to-text model for generating answers to the slot questions. Given that recent years have witnessed the success of transformers (Vaswani et al., 2017; Devlin et al., 2019; Raffel et al., 2020) in natural language processing (NLP) via transfer learning, we adopt the T5 (Raffel et al., 2020) model architecture and its pre-trained weights. The method is straightforward but effective, easy to adapt to domain-similar tasks, and efficient in data input without needing additional pre-processing and part-of-speech features. In this paper, we report its performance as compared to other participating runs. Our run achieves competitive performance as indicated by F1. However, evaluation is based on exact matches, and thus a prediction is deemed a true positive only when it is a character-by-character match with the ground truth from the candidate answers. Since our method treats the task as a text-generation-based question answering task instead of categorising labels from a fixed list of candidate labels, it potentially generates well-answered predictions that do not exactly match the ground truth answers. Having observed some of these mismatches in our experiment we subsequently transformed these to exact-matched answers using a simple approach based on Levenshtein string edit distance (Levenshtein, 1966). After this transformation, our best run reaches the same level of F1 performance as the best participating run.

For more details, see (Zong et al., 2020).

2 Related Work

The literature has seen much work on processing crisis-related messages on social media. For example, the TREC Incident Streams (McCreadie et al., 2020) track is a research initiative for seeking computational linguistic techniques for finding actionable information from Twitter during crises. With similar motivations to this initiative, many techniques have been applied for crisis messages categorisation and analysis in recent years. For instance, Miyazaki et al. (2019) apply label embedding for crisis tweet classification adopting a bi-directional LSTM model. Wang and Lillis (2020) leverage contextual ELMO embeddings (Peters et al., 2018) and data augmentation. CrisisBERT is proposed for crisis event detection (Liu et al., 2020) through fine-tuning the pre-trained BERT (Devlin et al., 2019).

More recently, the COVID-19 pandemic has motivated research on insights mining, information extraction and classification on social media. Since the pandemic is characterised by being long-lived, unlike more short-term crises such as earthquakes or shootings, the messages on social media are studied to gain a better understanding of the pandemic from multiple perspectives. For example, TweetsCOV19 (Dimitrov et al., 2020) is a corpus of semantically annotated tweets about COVID-19 that can potentially be used to explore public opinion and perception on the pandemic. Müller et al. (2020) introduce COVID-Twitter-BERT, which pre-trains a BERT model on hundreds of millions of COVID-related tweets and then fine-tunes the model to downstream tasks including vaccine sentiment analysis and vaccine stance classification.

Although most recent work involves fine-tuning transformer-based models for COVID-related short message processing, we have chosen to take an alternative approach for this task. Inspired by the text-to-text T5 model, our approach formulates the W-NUT Task 3 as a question-answering problem. The principal idea is that its unified source and target structure can be easily adapted to other domain-similar tasks such as informativeness classification or vaccine sentiment for COVID-related tweets. Regarding fine-tuning T5 for COVID-related language tasks, probably the most relevant work is that Tang et al. (2020) fine-tuned T5 for COVID-related question answering from scientific articles. However, they con-

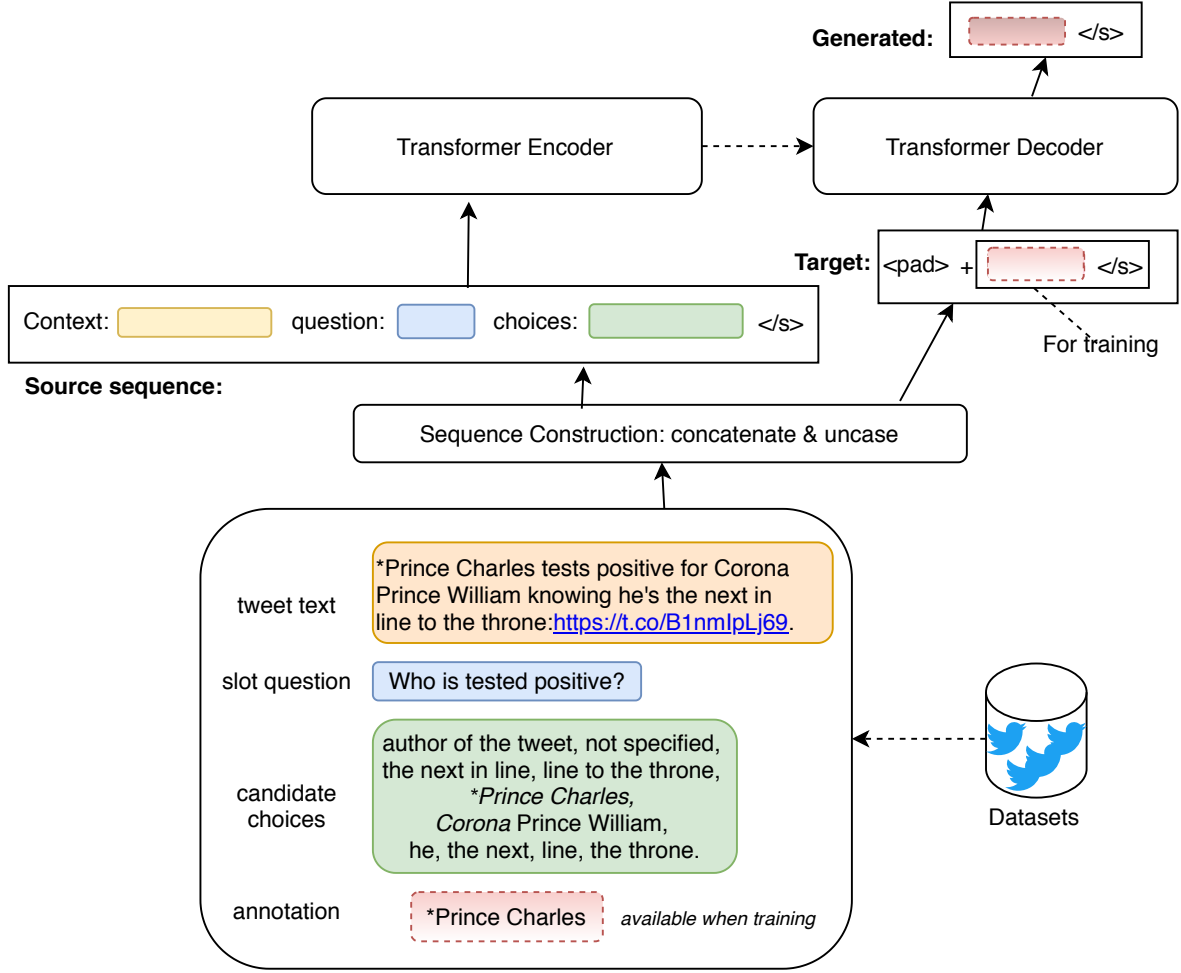


Figure 2: The system architecture of our approach

structured the source sequence by concatenating a document and a query, and the target sequence is a generated token indicating the document’s relevance to the query. Our approach is characterised by being capable of unifying various text classification tasks to a general question answering task. In our approach, the source sequence takes the raw text as the context, a classification description as the question and the classification’s labels as the candidate choices. The target sequence is simply the generated answers that are exact text spans from the source sequence, indicating which label(s) the text belongs to.

3 Method

In this section, we detail our approach in W-NUT Task-3. We firstly describe how a general transformer encoder-decoder model work in a sequence-to-sequence way and then introduce our approach adapting it to the event extraction task. Basically, given a source sequence

$X: \{x_1, x_2, \dots, x_n\}$, the transformer encoder uses it as the input to output its contextualised encoding sequence $\bar{X}: \{\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n\}$. The encoding process can be represented by a mapping function f with learnable parameters θ_e , as follows.

$$f_{\theta_e}: \mathbf{X}_{1:n} \rightarrow \bar{\mathbf{X}}_{1:n}.$$

The encoded source sequence then is used as the input to the transformer decoder that defines the conditional probability distribution of a target sequence $Y: \{y_1, y_2, \dots, y_m\}$ given $\bar{X}_{1:n}$:

$$p_{\theta_d}(\mathbf{Y}_{1:m} | \bar{\mathbf{X}}_{1:n}) = \prod_{i=1}^m p_{\theta_d}(y_i | \mathbf{Y}_{0:i-1}, \bar{\mathbf{X}}_{1:n}).$$

Where p with learnable parameters θ_d is the function to learn the conditional probability distribution by the decoder. The target token y_i is generated conditional on both the source context $\bar{X}_{1:n}$ and its previous $i - 1$ already-generated tokens in an auto-regressive way. The powerful part of the

transformer-based decoder is that it applies a single directional attention mechanism, i.e., masked multi-head attention as proposed by Vaswani et al. (2017) to learn the relations between the target token and its previous tokens. Meanwhile, it applies a bi-directional attention mechanism, i.e., multi-head attention, to learn the relations between the target token and the source context, which is also used in the encoder to learn the relations (self-attentions) between the source tokens.

Inspired by the feature of target generation given a source in a transformer encoder-decoder model, we leverage it for the shared task. Figure 2 presents the system architecture of our approach. The core component of the system is the sequence construction, which converts each tweet in the dataset into a source sequence and target sequence that well fits to train the transformer encoder-decoder model in a text-to-text fashion. As introduced, each tweet in the dataset is annotated in two parts, 1), labels indicating the event types, 2), answers to the slot questions. Our approach uses both parts of the annotations to construct the source and target sequences as follows:

- **Source:** This sequence is constructed from the raw tweets through mapping their major four attributes, i.e., as outlined in Figure 2, the tweet text is mapped to “context”, event type question (part 1) or slot question (part 2)³ to “question”, and the candidate choices to “choices”. This leads to the final source sequence concatenating all these fields in the form: “context: {tweet text} question: {slot/event question} choices: {candidates}”.
- **Target:** The target sequence needs both part 1 and part 2 annotations in training. They are simply mapped from the event labels for part 1 (“yes” or “no” indicating if the tweet falls into the event as specified in the question field of the source sequence) and annotation answers for part 2. Where the annotations are not available in inference, the source sequence and the target sequence with the start decoder token <pad> are fed as input to the model for generating the answers directly.

Although the aforementioned only details the

³The event type questions are manually constructed: taking DEATH as an example: *Does this tweet report death from coronavirus?*

sequence construction process using the event extraction corpus as an example, it follows similar steps when adapting to similar tasks. For example, for a binary informativeness classification task, the question field can be replaced by something like “Is the tweet informative of COVID-19?” and choices can be “yes” or “no” implying INFORMATIVENESS or UNINFORMATIVENESS. We leave this generalisation capability to future work and here continue to focus on the event extraction task.

After the source and target sequences are constructed, the next step is to train a transformer encoder-decoder model. As evidenced in the literature, sequence-to-sequence transformers such as BART (Lewis et al., 2019) or T5 (Raffel et al., 2020) has demonstrated great success in various downstream language tasks via transfer learning. Since T5 has different pre-trained weights available where in particular its small version is easy to handle in the pilot study, we set up the encoder-decoder model following the T5 (Raffel et al., 2020) architecture and train it on this task via fine-tuning its pre-trained weights.

4 Experiments

This section describes the details of experimental process and results.

4.1 Data Preparation

Because the original dataset is released with only Tweets IDs, some tweets had become invalid prior to our retrieval time and consequently 7,149 valid tweets were used in our experiments. Since each training tweet is assigned one event type label and multiple slot annotations, there a total of 34,464 examples used in our experiments after source and target sequence construction. Among these, we sample 10% as the validation set and the rest as the training set.

4.2 Training

Batch size	16
Epoch	12
Learning rate (LR)	5e-05
LR scheduler	Warmup linear decay
Warmup ratio	0.10
Optimizer	Adam

Table 1: Hyper-parameters of model training

In our experiments, we chose t5-small, t5-base, and t5-large for fine-tuning. Ta-

ble 1 shows the hyper-parameter configuration in our experiments. The batch size is selected from the options of {8, 16, 32}, based on evaluation on the validation set and the other hyper-parameters are determined with reference to the literature in a similar domain (Liu et al., 2020). We fine-tune each model with 12 epochs as we observe no further improvements after this. The maximum source and target sequence lengths were set to be 473 and 78 respectively since no examples exceeded these. The training was accelerated by a TPU3-8, taking around 4 hours to complete the `t5-large` fine-tuning. After the model is fine-tuned, we use greedy decoding during inference. This is because the generated texts are usually short answers in our problem and thus top-k or top-p sampling (Holtzman et al., 2019) would give similar results as greedy decoding.

	F1	P	R	Params
small-2	0.5308	0.4308	0.6913	1.0x
base-2	0.6225	0.5449	0.7258	3.7x
large-2	0.6392	0.5800	0.7118	12.8x

(a) Evaluation results of different sizes of models where **large-2** is the officially submitted **run-2** and Params refers to the model’s parameters relative to `t5-small` that has around 60M parameters.

team name	F1	P	R
winners	0.6598*	0.7272	0.6039
HLTRI	0.6476	0.7532*	0.5679
Our (run-2)	0.6392	0.5800	0.7118*
VUB	0.6160	0.6875	0.5580
UPennHLP	0.5237	0.6754	0.4277
Test_Positive	0.5114	0.5377	0.4875

(b) Evaluation results of submitted runs ranked by F1. **Our (run-2)** is named **UCD_CS** officially.

Ours	F1	P	R
run-1	0.6429*	0.5815	0.7188 *
run-2	0.6392	0.5800	0.7118
run-3	0.6367	0.5920*	0.6887

(c) Evaluation results of our `t5-large` runs at different epochs.

Ours	F1	P	R
post-run-1	0.6571*	0.5956	0.7327*
post-run-2	0.6517	0.5921	0.7247
post-run-3	0.6495	0.6050*	0.7012

(d) Evaluation results of our `t5-large` runs after post-processing.

Table 2: F1, precision (P) and recall (R) scores averaged over the five COVID-19 events where * refers to the highest in each column.

4.3 Results and Discussion

We saved the last three checkpoints of the final fine-tuned T5 models ready for evaluation at epochs 10, 11, and 12 (denoted as **run-3**, **run-2**, and **run-1** for `t5-large` respectively). The evaluation was conducted on a test set of 2,500 tweets (500 per event) for part-2 only. Since this shared task only allows one submission from each team and we found trivial difference between **run-2** and **run-1** based on the validation evaluation, we submitted **run-2** for the official evaluation. Table 2b reports the F1, recall (R) and precision (P) scores of the submitted runs. Additionally, we put the evaluation results of all our three `t5-large` runs in Table 2c as well as of different sizes of T5 models in Table 2a for reference⁴ (discussed in Section 4.3.2).

4.3.1 Performance

First, Table 2a indicates that the larger the model is, the better overall performance it can achieve. Interestingly, as compared to small-2, large-2’s advantage over base-2 seems not significant (F1: 0.6392 versus 0.6225) given its 12.8x larger size versus base-2’s 3.7x larger size. Hence, our base-2 can achieve competitive performance with a decent number of parameters. Here we continue to focus on the large runs since we want to explore the limit of our approach’s performance.

Table 2b presents the evaluation results of our `t5-large` based runs at different epochs. It shows that our submitted run (i.e., run-2) achieved competitive performance as compared to other participating runs (F1: our 0.6392 versus the highest 0.6598). Particularly, our run exhibited a significant advantage in recall over other runs (0.7118 versus the second-highest of 0.6039). However, this advantage is combined with a trade-off in terms of precision. This reveals that our run was somewhat “active” at finding the answers to the slot questions. This is consistent with its performance at event type level as well. Table 4 shows the evaluation results of our run-2 at event type level (we will discuss post-run-2 in Section 4.3.2). We find that our run achieves the best recall in every event type but quite behind in precision, especially for `can_not_test`, `death` and `cure`. To further analyse the cause of low precision, we

⁴The small-2 and base-2 corresponds to `t5-small` and `t5-base` respectively that are fine-tuned with the same experimental setup as **run-2**

Example 1	Tweet text	only precautionary steps can protect and prevent us from corona virus. sanitation, mask, alertness, yoga and healthy food.
	Slot question (where)	what is the cure for coronavirus mentioned by the author of the tweet?
	Our raw prediction	mask, alertness, yoga and healthy food
	Post-processed prediction	virus. sanitation, mask, alertness, yoga and healthy food
	Ground truth	virus. sanitation, mask, alertness, yoga and healthy food
Example 2	Tweet text	@bbhuttozardari what is your sindh government doing? stop playing politics at this time and take measures to prevent spread of corona. provide healthcare facilities. sometimes doing something is better than barking tweets.
	Slot question (what)	what is the cure for coronavirus mentioned by the author of the tweet?
	Our raw prediction	provide healthcare facilities
	Post-processed prediction	. provide healthcare facilities
	Ground truth	. provide healthcare facilities
Example 3	Tweet text	@realdonaldtrump they're drinking bleach to cure covid-19?.
	Slot question (what)	what is the cure for coronavirus mentioned by the author of the tweet?
	Our raw prediction	bleach
	Post-processed prediction	@realdonaldtrump they're drinking bleach
	Ground truth	@realdonaldtrump they're drinking bleach
Example 4	Tweet text	@joshua4congress my sister is a vet with an active-duty husband and a 6wk old baby. she has a fever and symptoms but can't get tested bc she lives in a military base in texas. they require exposure to a confirmed positive. she had to give birth without family the day after our grandma died.
	Slot question (where)	where is the can't-be-tested situation reported?
	Our raw prediction	a military base in texas
	Post-processed prediction	a military base
	Ground truth	texas
Example 5	Tweet text	so in a few weeks time 4 year olds will be expected to be back in school but can't get tested. doesn't make sense. #covid19 #dailybriefings
	Slot question (who)	who can not get a test?
	Our raw prediction	4 year olds
	Post-processed prediction	year olds
	Ground truth	a few weeks time 4 year olds

Table 3: Examples of mismatches (uncased), i.e., predictions that are self-evidently correct answers, but do not exactly match the ground truths in **violet text**. The **orange text** refers to the original submitted predictions generated by our approach and the **blue text** refers to the transformed predictions from the raw predictions based on their edit distances to candidate answers.

rethink the metrics used to evaluate runs with a similar approach to ours.

4.3.2 Post Processing

It is worth noting that, the part 2 annotations were originally labeled by providing a fixed list of candidate choices. For example, the candidate choices in Figure 2 are either pre-defined (author of the tweet, not specified) or text spans extracted from the raw tweet via the named entity tagging tool (Ritter et al., 2011). This makes it easy to treat the task as a classification-based slot filling task, i.e, binary categorising if a candidate choice answers a given slot type (Zong et al., 2020). In this sense, the F1, P and R are good metrics for evaluating the performance of a system in this task. However, our approach is based on answer-generation and thus it is likely to generate some correct predictions that are not exact character matches. In situations where a mismatch occurs this is doubly penalised by the metrics. The

prediction made by our system will be considered to be a false positive, because it does not match a label from the gold standard. Simultaneously the presence of a gold standard label that our system does not return results in a false negative being counted also. We argue this can result in the metrics underestimating the effective performance of such systems as it has an adverse effect on precision, recall and F1 score.

Table 2d presents a list of examples from **run-1** where this occurs. In these examples, the raw generated predictions are good answers to the corresponding slot questions, although they do not exactly match the ground truth from the candidate options. In particular, in example 2, the answer “provide healthcare facilities” is taken as a false positive as it misses the dot character at the beginning. In examples 3 and 5, our raw prediction is arguable a better answer than the ground truth. Example 1 is especially interesting in that our raw prediction omits one important word (“sani-

	F1			P			R		
	best	run-2	post-run-2	best	run-2	post-run-2	best	run-2	post-run-2
positive	0.6973	0.6778	0.6989	0.8569	0.7380	0.7620	0.6267	0.6267	0.6454
negative	0.7030	0.7030	0.7047	0.7107	0.6873	0.6890	0.7194	0.7194	0.7212
can_not_test	0.6523	0.5660	0.5667	0.6863	0.4646	0.4656	0.7240	0.7240	0.7240
death	0.6942	0.6048	0.6191	0.7240	0.4917	0.5041	0.7855	0.7855	0.8020
cure	0.6205	0.6078	0.6236	0.8405	0.4961	0.6236	0.7843	0.7843	0.8028

Table 4: The evaluation results of our run-2 and post-run-2 at event type level where best represents the highest score across all participating runs. These in bold stand for new state-of-the-art scores in W-NUT task-3 after applying **TransM**.

tation”) whereas the ground truth includes an additional word that is not part of the most appropriate question answer (“virus”). To alleviate this effect, we apply a simple post-processing method to transform the raw predictions to these that can best be exactly matched with the ground truth labels. The transformation method (which we name “TransM”) is described as follows.

TransM: Given the raw prediction r for an example x and its candidate choices $c : [c_1, c_2, ..c_i., c_n]$ where n is the number of candidates, r is converted to the final prediction t that is the one selected from c with the shortest edit distance to p . The edit distance is simply calculated by the Levenshtein (Levenshtein, 1966) distance divided by the length of c_i .

After this post-processing, we can now see from Table 2d that most of the post-processed predictions have been transformed to exactly-matched predictions. As a result, we re-evaluate our T5 large runs with the transformed predictions and present the results in Table 2d and report run-2 performance at event type level after applying TransM, i.e., post-run-2 in Table 4. First, Table 2d shows that our three runs can achieve higher scores overall as compared to the original predictions-based runs. Also, our best run post-run-1 can actually reach the same level of performance as the best participating run (F1: 0.6571 versus 0.6598). Referring to Table 4, it presents that not only does our post-run-2 achieve noticeable improvements across F1, P, and R in almost every event as compared to run-2 but it hits new state-of-the-art F1 scores in the positive, negative and cure events. Notably, for positive and cure, our run-2 was around 2 points behind the best participating runs in F1 but it reached the new state-of-the-art F1 scores after applying TransM (0.6989 and 0.6236). This reveals that the exact-matched metrics were exerting stronger underestimation in measuring our system’s performance for

answering tested-positive and cure-and-prevention related questions than other types of questions.

Although the post-processing alleviates the underestimation of performance, our run is not the best in precision. Arguably, the last two rows (example 4-5) in Table 2d are good examples showing that in some cases that our raw predictions contain the ground truths but are taken as false positives despite the post-processing: i.e., TransM helps alleviate the underestimation but not completely eliminate it. We present a complete list of unmatched predictions generated by run-2 or post-run-2 to enable the community to examine the system’s performance openly and critically⁵.

5 Conclusion

This paper presents our text-to-text based approach at W-NUT 2020 shared task 3. We show that the principal idea behind the approach is adaptability to other domain-similar tasks such as informativeness classification of COVID-19 tweets. We expect to conduct more work on this adaptability in the future. It is even more interesting to test the idea in zero-shot learning. For example, how well it performs if transferring the model that is trained on the event extraction corpus to do inference in the informativeness task directly without further training. In addition, we empirically present that our system is effective, achieving competitive performance and arguably the state-of-the-art F1 scores in three of five COVID-events in the shared task. Despite the effectiveness, one concern of our approach is the model size. Our best performed model is fine-tuned using the large version of T5 with around 770M parameters (Li et al., 2020). This makes it important to compress the model efficiently in the future.

⁵https://github.com/wangcongcong123/ttt/tree/master/covid_event

Acknowledgments

We would like to thank Google’s TensorFlow Research Cloud (TFRC) team who provided TPUs credits to support this research.

References

- 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.
- Dimitar Dimitrov, Erdal Baran, Pavlos Falafios, Ran Yu, Xiaofei Zhu, Matthäus Zloch, and Stefan Dietze. 2020. TweetsCOV19 – A Knowledge Base of Semantically Annotated Tweets about the COVID-19 Pandemic. *arXiv preprint arXiv:2006.14492*.
- Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *International Conference on Learning Representations*.
- Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.
- Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joseph E. Gonzalez. 2020. [Train large, then compress: Rethinking model size for efficient training and inference of transformers](#).
- Junhua Liu, Trisha Singhal Lucienne Blessing, Kristin L Wood, and Kwan Hui Lim. 2020. CrisisBERT: Robust Transformer for Crisis Classification and Contextual Crisis Embedding. *arXiv preprint arXiv:2005.06627*.
- Richard McCreadie, Cody Buntain, and Ian Soboroff. 2020. Incident Streams 2019: Actionable Insights and How to Find Them. *Proceedings of the International ISCRAM Conference, 2020-May(May)*.
- Taro Miyazaki, Kiminobu Makino, Yuka Takei, Hiroki Okamoto, and Jun Goto. 2019. Label embedding using hierarchical structure of labels for twitter classification. 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 6318–6323.
- Martin Müller, Marcel Salathé, and Per E Kummervold. 2020. COVID-Twitter-BERT: A Natural Language Processing Model to Analyse COVID-19 Content on Twitter. *arXiv preprint arXiv:2005.07503*.
- Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. [Deep contextualized word representations](#). *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.
- Alan Ritter, Sam Clark, Oren Etzioni, et al. 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.
- Kirk Roberts, Tasmee Alam, Steven Bedrick, Dina Demner-Fushman, Kyle Lo, Ian Soboroff, Ellen Voorhees, Lucy Lu Wang, and William R Hersh. 2020. [TREC-COVID: rationale and structure of an information retrieval shared task for COVID-19](#). *Journal of the American Medical Informatics Association*. Ocaa091.
- Raphael Tang, Rodrigo Nogueira, Edwin Zhang, Nikhil Gupta, Phuong Cam, Kyunghyun Cho, and Jimmy Lin. 2020. Rapidly bootstrapping a question answering dataset for covid-19. *arXiv preprint arXiv:2004.11339*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.
- Congcong Wang and David Lillis. 2020. Classification for Crisis-Related Tweets Leveraging Word Embeddings and Data Augmentation. In *Proceedings of the Twenty-Eighth Text REtrieval Conference (TREC 2019)*, Gaithersburg, MD.
- Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darin Eide, Kathryn Funk, Yannis Katsis, Rodney Kinney, Yunyao Li, Ziyang Liu, William Merrill, Paul Mooney, Dewey Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, Brandon Stilson, Alex Wade, Kuansan Wang, Nancy Xin Ru Wang, Chris Wilhelm, Boya Xie, Douglas Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. 2020. [CORD-19: The COVID-19 Open Research Dataset](#).
- Shi Zong, Ashutosh Baheti, Wei Xu, and Alan Ritter. 2020. [Extracting COVID-19 Events from Twitter](#).