imec-ETRO-VUB at W-NUT 2020 Shared Task-3: A multilabel BERT-based system for predicting COVID-19 events

Xiangyu Yang **Giannis Bekoulis** **Nikos Deligiannis**
Department of Electronics and Informatics (ETRO), Vrije Universiteit Brussel, 1050 Brussels, Belgium
imec, Kapeldreef 75, 3001 Leuven, Belgium
{xyanga,gbekouli,ndeligia}@etrovub.be

Abstract

In this paper, we present our system designed to address the W-NUT 2020 shared task for COVID-19 Event Extraction from Twitter. To mitigate the noisy nature of the Twitter stream, our system makes use of the COVID-Twitter-BERT (CT-BERT), which is a language model pre-trained on a large corpus of COVID-19 related Twitter messages. Our system is trained on the COVID-19 Twitter Event Corpus and is able to identify relevant text spans that answer pre-defined questions (i.e., slot types) for five COVID-19 related events (i.e., TESTED POSITIVE, TESTED NEGATIVE, CAN-NOT-TEST, DEATH and CURE & PREVENTION). We have experimented with different architectures; our best performing model relies on a multilabel classifier on top of the CT-BERT model that jointly trains all the slot types for a single event. Our experimental results indicate that our Multilabel-CT-BERT system outperforms the baseline methods by 7 percentage points in terms of micro average F_1 score. Our model ranked as 4^{th} in the shared task leaderboard.

1 Introduction

COVID-19, a highly infectious disease, has dramatically influenced the world. According to official COVID-19 related data from World Health Organization (WHO), the number of confirmed cases has surpassed 29 million, and the death toll rises to 922,252 as of 15 September 2020. Many countries have taken necessary measures, such as social distancing and mask wearing to prevent the spreading of the virus. Because of these measures, the communication among people has been changed. As a result, people tend to share their opinions on social media while also obtaining useful information from other users. Twitter, a popular social media platform, allows its users to share views through short Twitter messages (tweets). With a large number of COVID-19 related tweets shared in a daily basis, Twitter is a valuable source for one to find relevant piece of information about COVID-19.

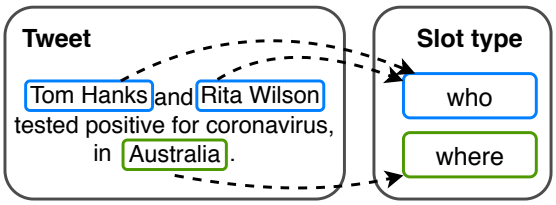

Figure 1: An example of an annotated tweet. That tweet reports a TESTED POSITIVE case. The goal of the task is to identify that the text spans "Tom Hanks" and "Rita Wilson" are slots of type "who", and "Australia" is a slot of type "where".

The "W-NUT 2020 shared task 3: COVID-19 event extraction from Twitter" is the task of finding useful information from COVID-19 related tweets. In this work, we focus on the provided dataset COVID-19 Twitter Event Corpus to build a system that can identify text spans (slots) that answer pre-defined questions related to COVID-19 events (e.g., "Who is tested positive (negative)?"). This task (i.e., identifying text spans that answer pre-defined questions) has been framed as a slot filling problem by the organizers of the competition and a detailed description of that is available on the work of Zong et al. (2020). Figure 1 illustrates an example of an annotated tweet. That tweet reports a TESTED POSITIVE case. The goal of the task (i.e., which has been framed as a slot filling problem) is to identify that the text spans "Tom Hanks" and "Rita Wilson" are slots of type "who", and "Australia" is a slot of type "where".

The rest of the paper is organized as follows. Section 2 introduces the related work on slot filling tasks. Section 3 describes the dataset we used to build our system along with the tweet preprocessing technique. Section 4 presents our proposed Multilabel-CT-BERT system as well as other

methods that have been tested in the context of this competition. Section 6 shows our experimental setup and results for the proposed systems and the baseline model. Section 7 summarizes our findings and concludes our work on this W-NUT shared task.

2 Related Work

The slot filling task is the problem where the goal is to identify fine-grained information (related to specific events of interest) from an input sequence. Specifically, given a text sequence, the aim is to find relevant text spans for certain types of slots (Benson et al., 2011), where in our case are the different slot types (e.g., "who", "where") for different event types (e.g., `TESTED POSITIVE`, `TESTED NEGATIVE`).

For the slot filling task, several approaches have been proposed in the literature. In particular, convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have been exploited in the works of Peng et al. (2015); Kurata et al. (2016); Vu (2016). More recently, Chen et al. (2019) developed a BERT-based joint intent classification and slot filling model. Their system relies on the pre-trained BERT model to encode the input sequences, and jointly trains the intent classification and slot filling tasks by maximizing the conditional probability between the two tasks. In another work, Coope et al. (2020) introduced a slot filling model called Span-ConveRT that is based on the ConveRT language model (Henderson et al., 2019) to extract text spans from dialogs. In this work, we focus on developing a system that can extract COVID-19 related information from Twitter messages. To do so, we make use of a language model that is optimized for COVID-19 related tweets, and we fine-tune the system by jointly training the different slots.

3 Dataset

We use the dataset provided by the shared task for our proposed system. The name of the dataset is COVID-19 Twitter Event Corpus (Zong et al., 2020).

3.1 COVID-19 Twitter Event Corpus

This corpus contains five COVID-19 related event types: `TESTED POSITIVE`, `TESTED NEGATIVE`, `CAN-NOT-TEST`, `DEATH` and `CURE & PREVENTION`. These event types are used to study the information that people are sharing on social media during the COVID-19 era. Table 1 shows the statistics of the COVID-19 Twitter Event Corpus. The number of overall annotated tweets is 7,500. However, due to the Twitter policy, the redistribution of Twitter content is restricted. Since the distributed corpus only includes the tweet IDs, we collected the tweet text from the Twitter server. The total number of the collected annotated tweets is 7,149, and the reason is that some of the tweets are deleted by the users or the Twitter server.

Event Type	# of Annotated Tweets	# of Collected Tweets	# of Slots
TESTED POSITIVE	2,500	2,400	9
TESTED NEGATIVE	1,200	1,146	8
CAN NOT TEST	1,200	1,127	5
DEATH	1,300	1,231	6
CURE & PREVENTION	1,300	1,245	3
TOTAL	7,500	7,149	31

Table 1: Statistics of COVID-19 Twitter Event Corpus and the number of tweets for the collected corpus.

3.2 Pre-processing

The COVID-19 Twitter Event Corpus provides candidate chunks along with annotations for each tweet text. To prepare instances for our system, we first replace the URLs in the tweets by a special `[URL]` token, and then enclose the current candidate chunk within special entity start and end tags: `<E>` and `</E>`, respectively (as in the work of Zong et al. (2020)). Table 2 shows an example of a processed tweet with an enclosed candidate chunk. In the example, the current candidate chunk is the word "Australia". To process the tweet, we enclose it inside the `<E>` and `</E>` tags. The processed result is "Tom Hanks and Rita Wilson tested positive for coronavirus, in `<E>` Australia `</E>`.".

4 Proposed System

In this section, we first introduce the underlying methods (BERT and CT-BERT) of our system. Then, we describe the proposed Multilabel-CT-BERT system in detail.

4.1 BERT

BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) is a language representation model based on Transformers (Vaswani et al., 2017). The BERT model is pre-trained on large unlabeled corpora (by using a bidirectional strategy) either in the masked language model task or in the next sentence prediction

tweet text	"Tom Hanks and Rita Wilson tested positive for coronavirus, in Australia."
candidate chunk	"Australia"
processed tweet text	"Tom Hanks and Rita Wilson tested positive for coronavirus, in <E> Australia </E>."

Table 2: An example of a pre-processed tweet. Given the text of a tweet and the current candidate chunk, we enclose the candidate chunk inside <E> and </E> tags within the tweet text. In this example, the candidate chunk Australia is enclosed in the aforementioned tags and placed back to the original tweet.

task. By applying appropriate inputs and outputs to the pre-trained BERT model, it can be easily fine-tuned end-to-end on a specific NLP (Natural Language Processing) task, such as sentence classification, sentence tagging, and question answering.

BERT is basically a stack of multiple transformer encoders. The encoder consists of a self-attention layer that helps the encoder to pay attention to other tokens of the input sequence while encoding a specific token, and a feed forward neural network that processes the output encoding from the attention layer. The BERT input representation is the sum of three embeddings: (i) token, (ii) segment and (iii) position embeddings. For the token embeddings, a special `[CLS]` token is added at the beginning of the input sequence, and a special `[SEP]` token is added at the end of each input sequence. The segment embeddings can inform the BERT model about the sequence that the current token belongs to. The position embeddings indicate the positions of the input tokens inside the sequence. The outputs of the BERT model are high-level representations of the input tokens. The hidden states of the `[CLS]` token can be used to perform various NLP tasks (e.g., text classification). There are two model architectures for BERT: BERT-BASE and BERT-LARGE. The BERT-LARGE model has more model parameters (e.g., hidden dimensions) than the BERT-BASE model.

4.2 CT-BERT

COVID-Twitter-BERT (CT-BERT) (Müller et al., 2020) is a transformer-based language model that is pre-trained on a large number of COVID-19 related tweets, while the BERT model is pre-trained on textual data from Wikipedia and book corpora. Therefore, the main drawback of the BERT model in the context of this competition is that the pre-trained BERT model does not contain relevant information about COVID-19, while it mainly includes formal language. In this competition, the dataset consists of user-generated noisy text from social media like Twitter, which contains informal language. To overcome the aforementioned shortcomings of the BERT model, the CT-BERT is proposed to better represent the COVID-19 related text.

The CT-BERT is based on the BERT-LARGE model and uses the same pre-training techniques as BERT, but the pre-training step of the CT-BERT starts with the trained weights from the BERT-LARGE model. The pre-trained CT-BERT can then be used for a wide variety of NLP tasks, such as classification and question answering.

4.3 Multilabel-CT-BERT

The idea of our multilabel system is motivated by the following observation: we observed that some slot types are semantically related to each other, so we can use a multilabel classifier to jointly train all the slot types. In addition, the use of the CT-BERT pre-trained model mitigates the noisy nature of the tweets.

Figure 2 illustrates the architecture of our proposed Multilabel-CT-BERT system. The small yellow rectangles represent the input tokens. The blue rectangle is the pre-trained CT-BERT model. The small green rectangles are the hidden representations of the input tokens. The grey rectangle is the multilabel classifier which consists of a feed-forward neural network with a *sigmoid* activation function. The red rectangle is the predicted score for the enclosed candidate chunk.

The system is based on the pre-trained CT-BERT model. It takes a processed tweet with an enclosed candidate chunk as an input to the pretrained CT-BERT model. The model produces hidden states for each input token. Then the hidden representation of the special entity tag <E> is used as input to the multilabel classifier on top of the CT-BERT to predict the labels for the current chunk. Our system aims to minimize the binary cross-entropy with logits loss (BCEWithLogitsLoss), which is implemented in PyTorch (Paszke et al., 2017).

5 Other systems

In this section, we describe several other systems including the baseline model, the Multilabel-BERT

system, the NLI (Natural Language Inference) system and the Pairwise system.

5.1 Baseline

Multitask-BERT-BASE (Zong et al., 2020) is a fine-tuned model that relies on BERT-BASE and has been proposed by the task organizers. For this model, each slot filling task is transformed into a binary classification problem: given a tweet T and a candidate slot S, the model predicts whether the slot S answers the pre-defined question (Zong et al., 2020). To leverage the semantically related slot types such as the "gender" slot with the "who" slot, this model jointly trains the slot types by sharing the same parameters for these slot type classifiers. The input to this model is the text of the tweet with the candidate chunk enclosed inside the special start <E> and end </E> tags. The final BERT hidden representation of the <E> tag is passed to a fully connected layer with a *softmax* activation function in order to validate or not the candidate chuck (i.e., binary prediction). Moreover, the BERT-LARGE model was exploited and is referred to as **Multitask-BERT-LARGE**.

5.2 Multilabel-BERT System

Multilabel-BERT-BASE is the variant of the Multilabel-CT-BERT that instead of using the CT-BERT pre-trained model, this model relies on BERT-BASE. Equivalently, the **Multilabel-BERT-LARGE** model is based on BERT-LARGE.

5.3 NLI System

NLI, which stands for Natural Language Interference, is the task of determining whether a "hypothesis" sentence is correct or not given a "premise" sentence. For the NLI system that we used in this work, the "premise" sentence is the slot filling question that used to annotate the COVID-19 Twitter Event Corpus, and the "hypothesis" sentence is the pre-processed tweet with an enclosed candidate chunk. A multiclass classifier, which consists of a feed-forward neural network and a *softmax* layer, is added on top of the BERT-BASE model. The number of classes is determined by the slot types in an event type, since one event type can have several slot types. The input to this system is a pair of one "premise" sentence and one "hypothesis" sentence. The output of this system is the predicted class, which is determined by the highest prediction score from all the classes. Two models are developed for this system: **NLI-CLS** and **NLI-E**.

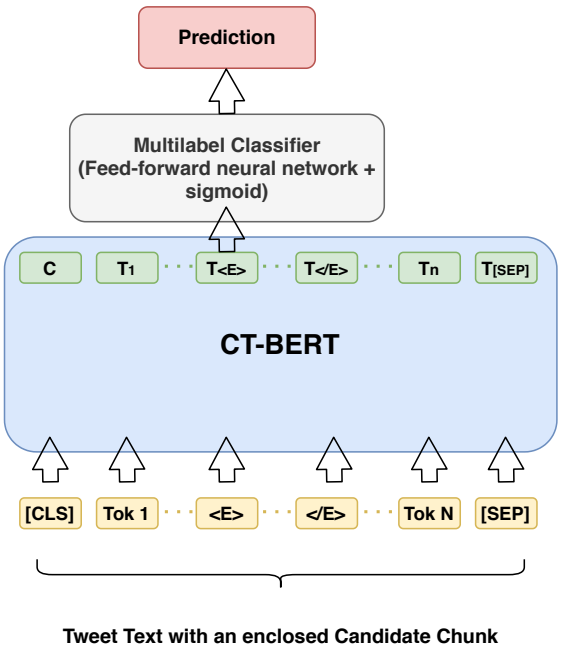

Figure 2: Multilabel-CT-BERT system architecture. This system consists of a pre-trained CT-BERT model in blue rectangle, and a multilabel classifier on top of the CT-BERT model in the grey rectangle. The small yellow and green rectangles represent the input sequence and the hidden representations, respectively. C is the hidden state of the [CLS] token. T_x is the corresponding hidden state of the input token X. The red rectangle is the prediction of the current enclosed chunk. The input to this system is a tweet text with an enclosed candidate chunk, and the hidden state of the entity tag <E> is used by the multilabel classifier for making predictions.

The input to the multiclass classifier in the NLI-CLS is the hidden state of the [CLS] token while in the NLI-E is the hidden state of the <E> token. This system aims to minimize the cross entropy loss between the different classes. Figure 3 shows the NLI system architecture.

5.4 Pairwise System

Figure 4 shows the architecture of the pairwise system. The pairwise system contains two parts: the BERT-BASE model and on top of that a binary classifier. The binary classifier has a feed-forward neural network with a *tanh* activation layer. This system focuses on each slot type for a single event type and treats each slot filling task as a binary prediction problem. One sentence for this system is defined as a tweet text with an enclosed candidate chunk like the processed tweet text shown in Table 2. For each candidate chunk, we generate such a sentence. In total, the number of all the sen-

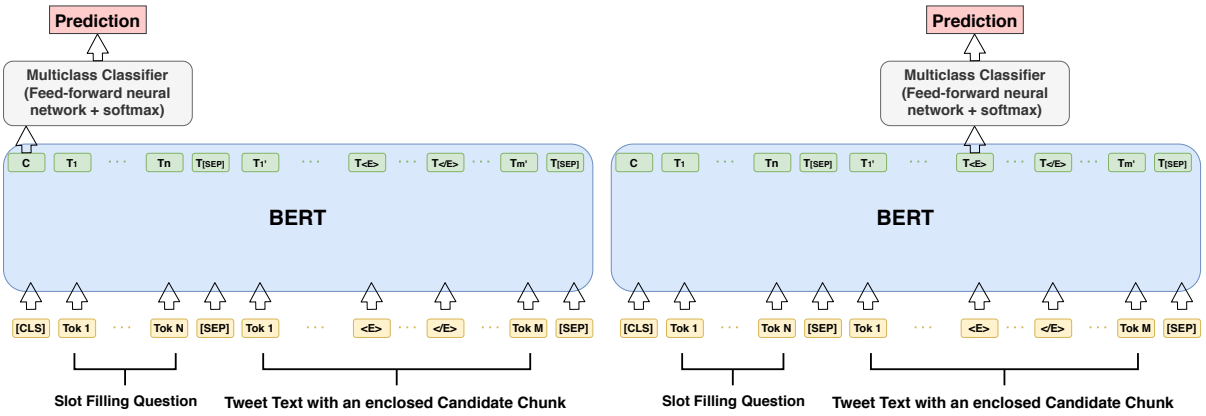

Figure 3: NLI system architecture. The left part is the NLI-CLS model and the right part is the NLI-E model. These two models have a similar structure. Both have two components, a pre-trained BERT model in the blue rectangle and a multiclass classifier in the grey rectangle. The small yellow rectangles represent the system input. The green rectangles are the hidden states for the input tokens. C is the hidden state of the `[CLS]` token. T_x is the corresponding hidden state of the input token X. The input to these models is a slot filling question and a tweet text. These two models differ in the way they use the hidden state as the input to the multiclass classifier.

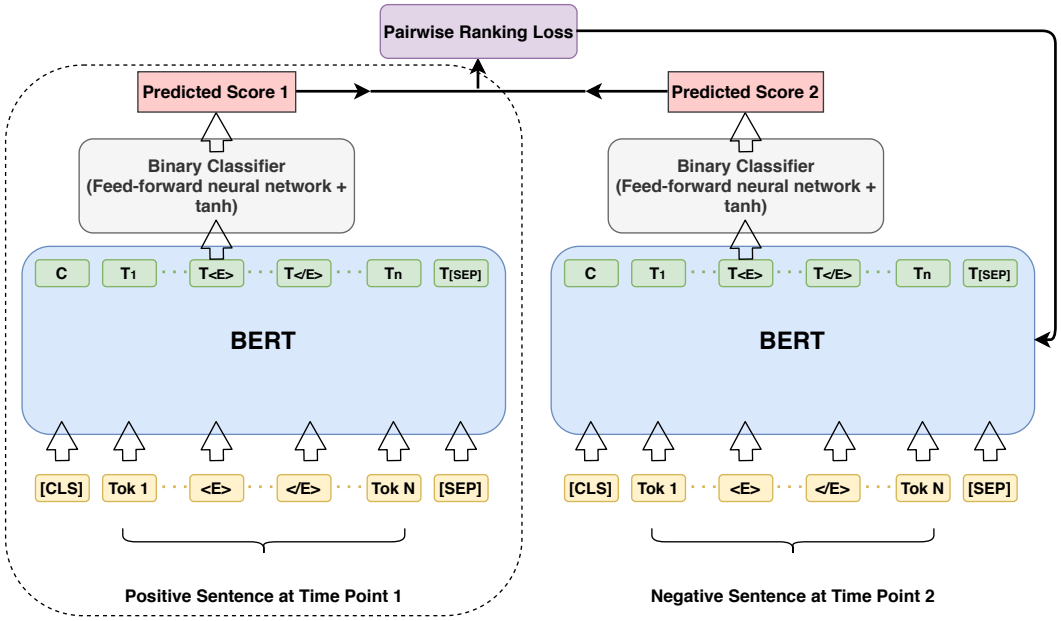

Figure 4: Pairwise system architecture. The left and right parts represent the same model structure but at different time points. The blue part is the pre-trained BERT model, the yellow part is the input tokens, the green part is the hidden states, the grey part is the binary classifier. C is the hidden state of the `[CLS]` token. T_x is the corresponding hidden state of the input token X. The system first takes as input a "positive" sentence at time point 1 and produces a predicted score for the "positive" sentence. Then the system processes a "negative" sentence at time point 2 to generate a predicted score for the "negative" sentence. The two predicted scores are used to calculate the pairwise ranking loss to update the model parameters.

tences is the number of all the candidate chunks in the provided corpus. The input to this system is a pair of one "positive" sentence and one "negative" sentence. The "positive" sentence is the sentence that the inner enclosed candidate chunk answers the current slot filling question. The "negative" sentence is determined by the "positive" sentence: for each "positive" sentence, all the remaining sentences from the total sentences are considered as "negative" sentences. The output is the predicted label for the enclosed chunk, which is determined by thresholding the predicted score from the binary classifier. This system aims to minimize the pairwise ranking loss using PyTorch (Paszke et al.,

2017), which is calculated as:

$$\text{loss}(x, y) = \max(0, -y \times (x_1 - x_2) + \text{margin}) \quad (4)$$

where x_1 and x_2 are input pairs (i.e., a "positive" and a "negative" sentence), $y = 1$ or -1. If $y = 1$ then the first input x_1 ranks higher than the second input x_2, and vice-versa for $y = -1$.

6 Experiments and Results

We compare our Multilabel-CT-BERT system with other systems (i.e., NLI, pairwise and baseline) to demonstrate the effectiveness of the proposed architecture.

6.1 Experimental Setup

Since each slot filling task in Multitask-BERT-BASE (Zong et al., 2020) is modeled as a binary classification problem, the non-binary "gender" slot is split into "gender male" and "gender female" slots. The Multitask-BERT-BASE uses a 60/15/20 split ratio for the train/dev/test sets and is optimized using the Adam algorithm (Kingma and Ba, 2015) with 2×10^{-5} as a learning rate. In order to compare with the Multitask-BERT-BASE, the same parameters are used for the rest of the models.

For the pairwise system, the number of training instances (pairs of sentences) could be too large so that the pairwise model becomes difficult to train. Thus we used a downsampling strategy to reduce the number of the sentence pairs. For every positive sentence, we randomly select a specific number of negative sentences. The ratio of the number of negative sentences with respect to the number of positive sentences is denoted as r. Two models are built for this system: **Pairwise-r50** (r = 50) and **Pairwise-r100** (r = 100).

For the Multitask-BERT-BASE, the Multilabel-BERT-BASE, the NLI-CLS, the NLI-E, the Pairwise-r50, and the Pairwise-r100, the batch size and epochs are set to 32 and 8, respectively. For the Multitask-BERT-LARGE, Multilabel-BERT-LARGE, and Multilabel-CT-BERT, the batch size and epochs are set to 64 and 15, respectively. For all the models, except for the NLI-CLS and the NLI-E, the best threshold for each slot filling task is determined by performing a grid search on the corresponding dev set. For grid search, the candidate thresholds for all the models, except for the NLI-CLS, the NLI-E, the Pairwise-r50, and the Pairwise-r100, are $\{0.1, 0.2, ..., 0.9\}$. The candidate thresholds for the Pairwise-r50 and the Pairwise-r100 are $\{-0.9, -0.8, ..., -0.1, 0, 0.1, ..., 0.9\}$.

Model	micro avg F_1	macro avg F_1
Multitask-BERT-BASE	0.5826	0.5498
Multitask-BERT-LARGE	0.5827	0.5539
NLI-E	0.5567	0.4984
NLI-CLS	0.5694	0.4913
Pairwise-r50	0.5458	0.4978
Pairwise-r100	0.5580	0.5064
Multilabel-BERT-BASE	0.6005	0.5717
Multilabel-BERT-LARGE	0.6206	0.5928
Multilabel-CT-BERT	**0.6585**	**0.6132**

Table 3: The aggregated results in terms of micro avg F_1 and macro avg F_1 scores. Micro avg F_1 combines the predictions (TP, FP, FN) from all the slot types and macro avg F_1 is the mean of all the F_1 scores for all the slot types.

6.2 Results

Table 3 shows the results[1] of the Multitask-BERT-BASE, Multitask-BERT-LARGE, NLI-E, NLI-CLS, Pairwise-r50, Pairwise-r100, Multilabel-BERT-BASE, Multilabel-BERT-LARGE, and Multilabel-CT-BERT in terms of micro avg F_1 (combining the predictions from all the slot types) and macro avg F_1 (the mean of all the F_1 scores of all the slot types) scores. We observe that our proposed Multilabel-CT-BERT is the best system among the compared systems on this shared task. The NLI and pairwise models are not performing better with respect to the baseline models. The reasons for this performance difference are that the NLI system uses formal language while the tweets contains informal language and the pairwise models use only part of the training data. Since these two systems are not performing well on this shared task, the detailed results for these two models are included only in the Appendix (see A.1 and A.2). As we can observe from the results, the multilabel BERT-based models also perform better than the baseline models. This is because that these systems not only train the slots jointly, but also share the same parameters for them by using a feed-forward neural

[1]For the W-NUT shared task 3, the organizers initially released an old dataset and then replaced it with a newer version of the dataset. Since we built the NLI and pairwise systems before the release of the new dataset, the reported results on these two systems are based on the old dataset. However, the results of the rest of the systems (Multilabel-CT-BERT, Multilabel-BERT system and the baseline system) are based on the new dataset.

Positive		Multitask-BERT-BASE			Multitask-BERT-LARGE			Multilabel-BERT-BASE			Multilabel-BERT-LARGE			Multilabel-CT-BERT		
slot	#	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
who	471	0.75	0.72	0.73	0.71	0.78	0.75	0.76	0.78	0.77	0.78	0.74	0.76	0.83	0.74	0.78
c.contact	39	0.29	0.27	0.28	0.46	0.5	0.37	0.56	0.24	0.33	0.59	0.3	0.4	0.61	0.43	0.5
relation	12	0.4	0.31	0.35	0.4	0.67	0.5	0.35	0.67	0.46	0.39	0.69	0.5	0.47	0.69	0.56
employer	77	0.58	0.36	0.45	0.37	0.69	0.49	0.53	0.47	0.5	0.5	0.54	0.52	0.6	0.41	0.49
recent .v	34	0.44	0.33	0.38	0.47	0.33	0.38	0.67	0.41	0.51	0.48	0.41	0.44	0.43	0.46	0.44
age	15	0.71	0.67	0.69	0.76	0.72	0.74	0.76	0.72	0.74	0.81	0.72	0.76	0.86	0.67	0.75
where	150	0.58	0.59	0.58	0.57	0.68	0.62	0.55	0.66	0.6	0.58	0.63	0.6	0.65	0.66	0.65
gender_m	126	0.64	0.71	0.67	0.68	0.66	0.67	0.64	0.72	0.68	0.67	0.65	0.66	0.64	0.7	0.67
gender_f	47	0.76	0.54	0.63	0.67	0.63	0.65	0.64	0.65	0.65	0.68	0.54	0.6	0.64	0.7	0.67
when	13	0.29	0.43	0.35	0.48	0.43	0.45	0.37	0.39	0.39	0.4	0.61	0.48	0.67	0.5	0.57
micro F_1				0.6193			0.6508			0.6452			0.6497			**0.6834**
macro F_1				0.511			0.562			0.563			0.5726			**0.608**
Negative		Multitask-BERT-BASE			Multitask-BERT-LARGE			Multilabel-BERT-BASE			Multilabel-BERT-LARGE			Multilabel-CT-BERT		
slot	#	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
who	140	0.6	0.58	0.59	0.59	0.58	0.59	0.67	0.58	0.63	0.64	0.6	0.62	0.72	0.64	0.68
relation	25	0.6	0.63	0.61	0.57	0.67	0.62	0.73	0.67	0.7	0.63	0.71	0.67	0.78	0.58	0.67
where	22	0.44	0.52	0.48	0.48	0.48	0.48	0.34	0.45	0.39	0.44	0.73	0.55	0.58	0.64	0.61
gender_m	48	0.65	0.59	0.62	0.54	0.75	0.62	0.6	0.65	0.62	0.65	0.65	0.65	0.67	0.61	0.64
gender_f	20	0.61	0.52	0.56	0.45	0.48	0.47	0.63	0.57	0.6	0.59	0.62	0.6	0.67	0.67	0.67
micro F_1				0.586			0.5785			0.6095			0.6213			**0.6614**
macro F_1				0.572			0.556			0.588			0.618			**0.654**
CAN NOT TEST		Multitask-BERT-BASE			Multitask-BERT-LARGE			Multilabel-BERT-BASE			Multilabel-BERT-LARGE			Multilabel-CT-BERT		
slot	#	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
who	108	0.56	0.51	0.53	0.55	0.56	0.55	0.52	0.52	0.52	0.65	0.56	0.6	0.71	0.57	0.63
relation	44	0.71	0.45	0.56	0.54	0.47	0.51	0.58	0.57	0.57	0.8	0.45	0.58	0.61	0.57	0.59
where	23	0.68	0.39	0.5	0.61	0.45	0.52	0.59	0.42	0.49	0.62	0.68	0.65	0.7	0.61	0.65
symptoms	54	0.6	0.46	0.53	0.56	0.55	0.56	0.57	0.52	0.54	0.64	0.57	0.6	0.58	0.59	0.58
micro F_1				0.5297			0.5409			0.5304			0.6069			**0.6157**
macro F_1				0.53			0.535			0.53			0.6075			**0.6125**
DEATH		Multitask-BERT-BASE			Multitask-BERT-LARGE			Multilabel-BERT-BASE			Multilabel-BERT-LARGE			Multilabel-CT-BERT		
slot	#	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
who	141	0.67	0.68	0.67	0.66	0.6	0.63	0.7	0.73	0.71	0.66	0.74	0.7	0.56	0.84	0.67
relation	26	0.7	0.27	0.39	0.64	0.35	0.45	0.68	0.58	0.62	0.55	0.62	0.58	0.63	0.38	0.48
when	23	0.66	0.63	0.64	0.68	0.5	0.58	0.64	0.46	0.53	0.68	0.74	0.71	0.62	0.78	0.69
where	54	0.74	0.53	0.62	0.78	0.53	0.63	0.55	0.79	0.65	0.61	0.72	0.66	0.46	0.79	0.59
age	33	0.68	0.87	0.76	0.72	0.93	0.81	0.58	0.77	0.66	0.7	0.77	0.73	0.69	0.83	0.76
micro F_1				0.6516			0.6318			0.6613			**0.6866**			0.6527
macro F_1				0.616			0.62			0.634			**0.676**			0.638
CURE&PREV		Multitask-BERT-BASE			Multitask-BERT-LARGE			Multilabel-BERT-BASE			Multilabel-BERT-LARGE			Multilabel-CT-BERT		
slot	#	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
opinion	62	0.5	0.56	0.53	0.33	0.61	0.43	0.39	0.71	0.5	0.41	0.34	0.37	0.68	0.58	0.63
what	148	0.56	0.61	0.58	0.7	0.57	0.63	0.65	0.62	0.64	0.61	0.69	0.65	0.59	0.76	0.66
who	61	0.49	0.41	0.45	0.45	0.41	0.43	0.46	0.53	0.49	0.51	0.41	0.45	0.52	0.53	0.52
micro F_1				0.532			0.5113			0.5559			0.5387			**0.6125**
macro F_1				0.52			0.4967			0.5433			0.49			**0.6033**
micro avg F_1				0.5826			0.5827			0.6005			0.6206			**0.6585**
macro avg F_1				0.5498			0.5539			0.5717			0.5928			**0.6132**

Table 4: The results of Multitask-BERT-BASE, Multitask-BERT-LARGE, Multilabel-BERT-BASE, Multilabel-BERT-LARGE, and Multilabel-CT-BERT. # indicates the number of golden slots in the test set. Positive is the `TESTED POSITIVE` event. Negative is the `TESTED NEGATIVE` event. c.contact is the close contact slot type. recent.v is the recent travel slot type. gender_m is the gender male slot type while the gender_f is the gender female slot type.

network on top of the underlying model (BERT or COVID-Twitter-BERT). Multilabel-CT-BERT outperforms the multilabel BERT-based models because the CT-BERT-based model is optimized on COVID-19 related tweets.

Table 4 shows the detailed results of Multitask-BERT-BASE, Multitask-BERT-LARGE, Multilabel-BERT-BASE, Multilabel-BERT-LARGE, and Multilabel-CT-BERT. The results are based on the test set. For each model, the precision (P), recall (R) and F_1 (F_1) scores are reported for each slot type of each event type. In addition, the micro F_1 and macro F_1 are reported for each event type, and the micro avg F_1 and macro avg F_1 are reported for all the event types.

From these detailed results of Table 4 (i.e., micro avg F_1 and macro avg F_1), we can observe that the multilabel-based systems (Multilabel-BERT-BASE, Multilabel-BERT-LARGE and Multilabel-CT-BERT) outperform the multitask-based systems (Multitask-BERT-BASE and Multitask-BERT-LARGE). We also notice that systems using BERT-

LARGE perform better than those using BERT-BASE. Except for the slightly worse performance in terms of micro F_1 and macro F_1 on the DEATH event type compared to other multilabel systems, the Multilabel-CT-BERT substantially improves the system performance on the rest of the event types. Compared to the baseline model, the Multilabel-CT-BERT achieves a micro avg F_1 of 65.85% (7.59 percentage points absolute improvement) and macro avg F_1 of 61.32% (6.34 percentage points absolute improvement).

To conclude, the Multilabel-CT-BERT system is the best performing system among the compared systems on the COVID-19 Twitter Event Corpus.

7 Conclusion

In this paper, a COVID-Twitter-BERT based Multilabel-CT-BERT system is proposed in the context of the W-NUT shared task to deal with the slot filling problem of the recently introduced COVID-19 Twitter Event Corpus. Our experimental results illustrate that the proposed Multilabel-CT-BERT system outperforms the baseline and other proposed models in terms of micro avg F_1 and macro avg F_1 scores.

References

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.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*.

Sam Coope, Tyler Farghly, Daniela Gerz, Ivan Vulić, and Matthew Henderson. 2020. Span-convert: Few-shot span extraction for dialog with pretrained conversational representations. *arXiv preprint arXiv:2005.08866*.

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.

Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Ivan Vulić, et al. 2019. Convert: Efficient and accurate conversational representations from transformers. *arXiv preprint arXiv:1911.03688*.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, San Diego, USA.

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.

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*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *Proceedings of the NIPS 2017 Autodiff Workshop*.

Baolin Peng, Kaisheng Yao, Li Jing, and Kam-Fai Wong. 2015. Recurrent neural networks with external memory for spoken language understanding. In *Natural Language Processing and Chinese Computing*, pages 25–35. Springer.

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.

Ngoc Thang Vu. 2016. Sequential convolutional neural networks for slot filling in spoken language understanding. In *Proceedings of Interspeech 2016*, pages 3250–3254.

Shi Zong, Ashutosh Baheti, Wei Xu, and Alan Ritter. 2020. Extracting covid-19 events from twitter. *arXiv preprint arXiv:2006.02567*.

A Appendices

A.1 NLI System Results

Table 5 shows the detailed results of the NLI models: NLI-CLS and NLI-E.

A.2 Pairwise System Results

Table 6 shows the detailed results of the pairwise models: Pairwise-r50 and Pairwise-r100.

Positive		NLI-E			NLI-CLS		
slot	#	P	R	F_1	P	R	F_1
who	471	0.74	0.73	0.73	0.75	0.75	0.75
c.contact	39	0.39	0.44	0.41	0.3	0.33	0.31
relation	12	0.44	0.33	0.38	0.41	0.58	0.48
employer	77	0.37	0.39	0.38	0.5	0.22	0.31
recent .v	34	0.53	0.26	0.35	0.38	0.18	0.24
age	15	0.58	0.73	0.65	0.57	0.87	0.68
where	150	0.49	0.59	0.53	0.46	0.69	0.55
gender_m	126	0.7	0.51	0.59	0.7	0.58	0.63
gender_f	47	0.7	0.66	0.68	0.78	0.62	0.69
when	13	0.37	0.77	0.5	0.2	0.62	0.3
micro F_1			0.6194			**0.6245**	
macro F_1			**0.52**			0.494	
Negative		NLI-E			NLI-CLS		
slot	#	P	R	F_1	P	R	F_1
who	140	0.55	0.66	0.6	0.61	0.62	0.62
relation	25	0.29	0.2	0.24	0.78	0.28	0.41
where	22	0.28	0.32	0.3	0	0	0
gender_m	48	0.59	0.56	0.57	0.57	0.54	0.55
gender_f	20	0.6	0.45	0.51	0.67	0.5	0.57
micro F_1			0.5323			**0.5567**	
macro F_1			**0.44**			0.43	
CAN NOT TEST		NLI-E			NLI-CLS		
slot	#	P	R	F_1	P	R	F_1
who	108	0.72	0.44	0.55	0.56	0.54	0.55
relation	44	0.7	0.36	0.48	0.53	0.43	0.48
where	23	0.48	0.65	0.56	0.4	0.61	0.48
symptoms	54	0.75	0.39	0.51	0.58	0.35	0.44
micro F_1			**0.5305**			0.5023	
macro F_1			**0.525**			0.4875	
DEATH		NLI-E			NLI-CLS		
slot	#	P	R	F_1	P	R	F_1
who	141	0.72	0.44	0.55	0.64	0.76	0.69
relation	26	0.7	0.36	0.48	0.63	0.46	0.53
when	23	0.48	0.65	0.56	0.51	0.51	0.51
where	54	0.75	0.39	0.51	0.81	0.25	0.38
age	33	0.55	0.88	0.67	0.66	0.76	0.7
micro F_1			0.5803			**0.6197**	
macro F_1			0.554			**0.562**	
CURE&PREV slot	#	NLI-E			NLI-CLS		
		P	R	F_1	P	R	F_1
opinion	62	0.49	0.42	0.45	0.53	0.55	0.54
what	148	0.63	0.62	0.63	0.6	0.56	0.58
who	61	0.39	0.21	0.28	0.46	0.26	0.33
micro F_1			0.5209			**0.5439**	
macro F_1			0.453			**0.4833**	
micro avg F_1			0.5567			**0.5694**	
macro avg F_1			**0.4984**			0.4913	

Table 5: The detailed results of NLI-E and NLI-CLS.

Positive		Pairwise-r50			Pairwise-r100		
slot	#	P	R	F_1	P	R	F_1
who	471	0.74	0.75	0.74	0.78	0.7	0.74
c.contact	39	0.23	0.38	0.29	0.24	0.31	0.27
relation	12	0.28	0.58	0.38	0.5	0.08	0.14
employer	77	0.44	0.4	0.42	0.44	0.44	0.44
recent .v	34	0.53	0.26	0.35	0.41	0.65	0.5
age	15	0.42	0.53	0.47	0.5	0.87	0.63
where	150	0.54	0.58	0.56	0.47	0.67	0.55
gender_m	126	0.61	0.61	0.61	0.69	0.61	0.65
gender_f	47	0.51	0.59	0.55	0.65	0.55	0.6
when	13	0.44	0.62	0.52	0.22	0.77	0.34
micro F_1			0.6085			**0.6177**	
macro F_1			**0.489**			0.486	
Negative		Pairwise-r50			Pairwise-r100		
slot	#	P	R	F_1	P	R	F_1
who	140	0.53	0.69	0.6	0.6	0.56	0.58
relation	25	0.39	0.48	0.43	0.37	0.64	0.47
where	22	0.32	0.68	0.43	0.32	0.55	0.41
gender_m	48	0.38	0.27	0.32	0.54	0.46	0.5
gender_f	20	0.61	0.55	0.58	0.36	0.75	0.48
micro F_1			0.5202			**0.5245**	
macro F_1			0.472			**0.4879**	
CAN NOT TEST		Pairwise-r50			Pairwise-r100		
slot	#	P	R	F_1	P	R	F_1
who	108	0.5	0.55	0.52	0.52	0.57	0.54
relation	44	0.52	0.55	0.53	0.5	0.52	0.51
where	23	0.3	0.61	0.41	0.28	0.61	0.38
symptoms	54	0.34	0.65	0.45	0.45	0.41	0.43
micro F_1			0.487			**0.4899**	
macro F_1			**0.4775**			0.465	
DEATH		Pairwise-r50			Pairwise-r100		
slot	#	P	R	F_1	P	R	F_1
who	141	0.63	0.77	0.69	0.69	0.63	0.66
relation	26	0.32	0.73	0.45	0.33	0.69	0.45
when	23	0.47	0.92	0.62	0.48	0.78	0.6
where	54	0.46	0.6	0.52	0.57	0.55	0.56
age	33	0.58	0.85	0.69	0.61	0.82	0.7
micro F_1			**0.6227**			0.6221	
macro F_1			0.594			**0.594**	
CURE&PREV slot	#	Pairwise-r50			Pairwise-r100		
		P	R	F_1	P	R	F_1
opinion	62	0.29	0.76	0.42	0.45	0.5	0.47
what	148	0.52	0.68	0.59	0.6	0.63	0.62
who	61	0.32	0.39	0.36	0.34	0.43	0.38
micro F_1			0.4907			**0.5357**	
macro F_1			0.4566			**0.49**	
micro avg F_1			0.5458			**0.558**	
macro avg F_1			0.4978			**0.5046**	

Table 6: The detailed results of Pairwise-r50 and Pairwise-r100.