Attentively Embracing Noise for Robust Latent Representation in BERT

Gwenaelle Cunha Sergio[1], **Dennis Singh Moirangthem**[1], **Minho Lee**[2]
[1] School of Electronics and Electrical Engineering
[2] Department of Artificial Intelligence
Kyungpook National University, Daegu, 41566, South Korea
{gwena.cs,mdennissingh,mholee}@gmail.com

Abstract

Modern digital personal assistants interact with users through voice. Therefore, they heavily rely on automatic speech recognition (ASR) in order to convert speech to text and perform further tasks. We introduce **EBERT**, which stands for **E**mbrace**BERT**, with the goal of extracting more robust latent representations for the task of noisy ASR text classification. Conventionally, BERT is fine-tuned for downstream classification tasks using only the `[CLS]` starter token, with the remaining tokens being discarded. We propose using all encoded transformer tokens and further encode them using a novel attentive embracement layer and multi-head attention layer. This approach uses the otherwise discarded tokens as a source of additional information and the multi-head attention in conjunction with the attentive embracement layer to select important features from clean data during training. This allows for the extraction of a robust latent vector resulting in improved classification performance during testing when presented with noisy inputs. We show the impact of our model on both the Chatbot and Snips corpora for intent classification with ASR error. Results, in terms of F1-score and mean between 10 runs, show that our model significantly outperforms the baseline model.[1]

1 Introduction

In recent year, machine learning methods have significantly contributed to the development of automatic speech recognition (ASR) (Deng and Li, 2013; Padmanabhan and Johnson Premkumar, 2015; Benkerzaz et al., 2019; Nassif et al., 2019). The fast advances in the ASR field can be explained by the urgent need of efficiently bridging the gap between humans and the ever evolving technologies surrounding us. As technology becomes smaller and more portable, it becomes clear that the only feasible way of interaction is through a digital personal assistant with voice interface (Tulshan and Dhage, 2018). However, high error rates still prevail and pose a tremendous hurdle in the widespread adoption of speech technology by users worldwide (Zavareh et al., 2013; Errattahi et al., 2018).

Researchers in the computational linguistics community analyze noisy text (Subramaniam, 2010) obtained through informal settings, such as social networks, or through processing techniques, such as optical character recognition (Granet et al., 2018; Tomokiyo et al., 2018), ASR (Vinciarelli, 2005; Agarwal et al., 2007; Shrestha et al., 2019), or machine translation models (Sergio et al., 2018). We focus on noisy ASR text classification. Researchers in this area have simulated noise introduced through ASR systems and they have investigated the performances of Naive Bayes and Support Vector Machine (SVM) classifiers in noisy text classification (Vinciarelli, 2005; Agarwal et al., 2007). Shrestha et al. (2019) have also investigated the impact in performance of using text embedded with speech data with neural networks and logistic regression. The aforementioned researches, however, focus mainly on investigating the impact that noise has in the performance of shallow machine learning models, such as SVM or vanilla neural network, rather than focusing on extracting more robust input representation for improved performance. More recently, the transformer (Vaswani et al., 2017) based model BERT (Devlin et al.,

[1]Code, data, and results available at https://github.com/gcunhase/EmbraceBERT.

Proceedings of the 28th International Conference on Computational Linguistics, pages 3479–3491
Barcelona, Spain (Online), December 8-13, 2020

2018) has achieved state of the art in various natural language processing tasks. This inspired Hrinchuk et al. (2020) to suggest the use of a transformer encoder-decoder architecture for the task of ASR correction and Liao et al. (2020) to improve readability also using transformer-based sequence-to-sequence architectures for the same task.

The pre-trained model BERT is fine-tuned for downstream tasks by extracting an encoded transformer token for each corresponding input token. In BERT's classic text classification approach, only the sentence starter `[CLS]` token is used for classification. However, we hypothesize that the discarded tokens can be used as a source of additional information, and should therefore also be used. We also search for inspiration outside of the noisy text analytics domain into the multimodal data domain, with concepts from EmbraceNet (Choi and Lee, 2019), which is proposed to improve robustness in the absence of data. The authors achieve that with an embracement layer that randomly selects important features from each modality vector and meshes them into one robust vector called embraced vector. We propose using the otherwise discarded tokens as a source of additional information and further encode them with a novel attentive embracement and multi-head attention layer. The combination of these layers allows the model to select important features from clean data during training, so that when presented with noisy inputs during testing, it is able to extract a robust latent vector for improved classification performance. We call our proposed model EmbraceBERT (EBERT).

In summary, our contributions are three-fold:

- Novel model, called EBERT, for extraction of robust latent representation from noisy text. This model further encodes all transformer encoded tokens, used as a source of additional information, with a combination of a novel attentive embracement and multi-head attention layer. This approach allows for important features to be selected from clean data during training so as to improve performance during testing, when presented with noisy inputs.

- We show improved performance in the ASR noisy text classification task when compared to the baseline model. The dataset used for evaluation is the Chatbot Natural Language Understanding (NLU) Evaluation Corpus for intent classification (clean data) and two variations with ASR error. The model is evaluated on three settings: (1) trained and tested with clean data, (2) trained with clean and tested with noisy data, and (3) trained and tested with noisy data.

- We provide extensive ablation study to show the importance of each module in our method and we also release our PyTorch code.

2 Related Works

2.1 BERT for Text Classification

BERT (Devlin et al., 2018), short for Bidirectional Encoder Representations from transformers, is a powerful language representation model that achieves state of the art on various natural language processing tasks, including text classification (Devlin et al., 2018). Fig. 1(a) shows that this model is built on transformers (Vaswani et al., 2017), the first fully attentional model to be used in sequence-modeling tasks, surpassing the existing best models with the introduction of a self-attention mechanism for input representations and positional encoding for sequence ordering.

The self-attention mechanism consists of a scaled dot-product attention being used to map a query Q and available information, in the form of key-value pairs K-V, to an output. This mapping is shown in Eq. (1):

$$Att(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

where d_k is the dimension of K. Note that in self-attention, the Q, K, V information is originated from the same input.

BERT attains such impressive results with the development of two pre-training tasks: masking and prediction of a small percentage of the input tokens, and prediction of the likelihood that sentence A is

followed by sentence B. The combination of these two strategies greatly increases context awareness, enabling it to be a high performing language model. Regarding downstream tasks, BERT can be fine-tuned by using the encoded transformer tokens (green tokens in Fig. 1(a)), which are extracted for every input token and include a special starter symbol called `[CLS]` token (Devlin et al., 2018). These encoded tokens contain their distributional contextual representation in the sentence and the choice of which ones are used depend on the task at hand. For the text classification task, the `[CLS]` token is used as a feature vector representing the entire sentence, which can then be classified with a simple feedforward classifier, as shown in Eq. (2):

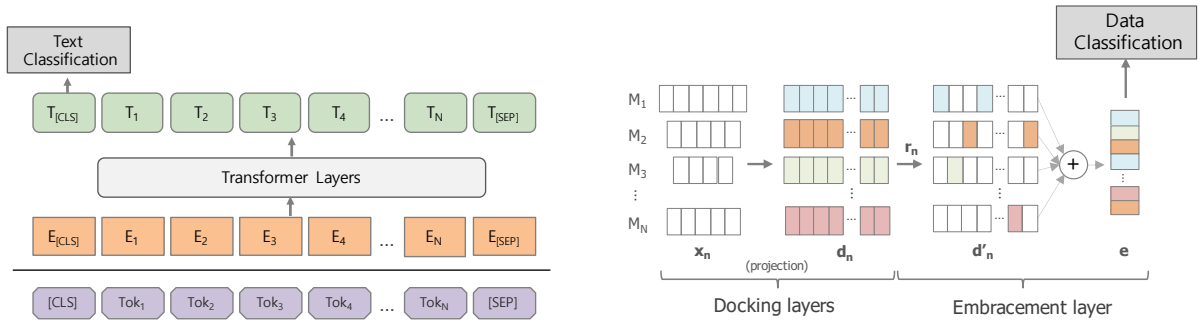
$$p(c|\mathbf{T}_{[\mathbf{CLS}]}) = \text{softmax}(W\mathbf{T}_{[\mathbf{CLS}]}) \tag{2}$$

where p is the probability of class c given the `[CLS]` token $\mathbf{T}_{[\mathbf{CLS}]}$ and W is the trainable weight matrix. In this classic approach, the remaining tokens are discarded. Although we concede that the `[CLS]` token might be a good single representation of the input for clean data, it lacks the same representation efficiency in noisy data. We hypothesize that the remaining tokens can be used to alleviate this issue by serving as a source of additional information to improve the model's robustness and performance.

2.2 EmbraceNet for Incomplete Data

EmbraceNet (Choi and Lee, 2019), shown in Fig. 1(b), is a model proposed to improve robustness in classification tasks involving incomplete multimodal data. The authors assume that each modality $M_{n\in\{1,2,\ldots,N\}}$ is represented with features of different sizes x_n, so their first measure is to project those features into vectors of same length d_n through a docking layer, sometimes also called adapter or projection layer. Following that layer is an embracement layer, which randomly selects important features d'_n from each feature vector, and adds them in order to obtain an embraced vector e that can be used for data classification.

The authors show good performance using the embraced vector in classification tasks with partially missing data in bi-modal MNIST, sensor, and activity recognition datasets. However, there a few limitations of EmbraceNet that we address in this work. First, since the probability p depends on the number of modalities available, the user must indicate which data are missing and adjust p accordingly. Secondly, the selection probability p is the same for every modality, whereas we believe some might be more important than others. Lastly, the docking layer is redundant for our purpose since the considered tokens have the same size.



(a) BERT (b) EmbraceNet

Figure 1: Architecture of models relevant to this work: (a) BERT for text classification and (b) EmbraceNet for incomplete multimodal data classification.

3 Proposed Model

In this section, we propose the novel EBERT model for the task of noisy ASR text classification. As the name suggests, our model is based on BERT, and as such, its first layers are transformer layers that encode the input embeddings E into tokens T. Conventionally, when fine-tuning BERT for downstream classification tasks, only the `[CLS]` starter token is used, with the remaining $T_{i\forall i\in\{1,\ldots,N\}}$ tokens being

discarded. Although we concede that the [CLS] token might be a good single representation of the input for clean data, it lacks the same representation efficiency in noisy data.

We tackle this issue by proposing the use of all encoded tokens as a source of additional information to improve the model's robustness and performance. Now, instead of a single vector that can directly be used for classification, we are faced with multiple ones. Our solution, shown in Fig. 2, is to introduce a multi-head attention layer with Q, K, V obtained from T to extract additional important information d, followed by a novel attentive embracement layer to obtain the embraced vector e. Note that we consider each token T, which can be missing or incorrect, as a different modality in order to use the embracement layer. The embraced vector is then given to a projection layer together with the [CLS] token, resulting in a single robust representation vector T'_C that can be used for classification. The projection layer is as in Eq. (3):

$$\mathbf{T'_C} = \text{proj}(T_{[CLS]}, e) = W_{proj} \begin{bmatrix} T_{[CLS]} \\ e \end{bmatrix} \tag{3}$$

where W_{proj} is the trainable projection layer weight matrix. This token can then be classified with a simple feedforward classifier, as shown in Eq. (4):

$$p(c|\mathbf{T'_C}) = \text{softmax}(W_C \mathbf{T'_C}) \tag{4}$$

where p is the probability of class c given the final classification token $\mathbf{T'_C}$ and W_C is the trainable classification layer weight matrix.

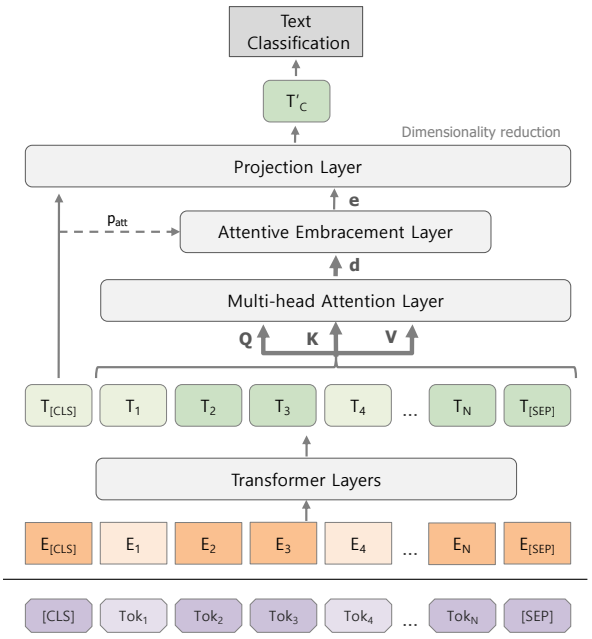


Figure 2: Proposed EBERT model: hierarchical structure of transformer layers and multi-head attention followed by a novel attentive embracement layer to obtain the embraced vector e from tokens T. The embraced vector is then given to a projection layer together with the [CLS] token, resulting in a single robust representation vector T'_C that can be used for classification.

In the conventional embracement layer (Fig. 3(a)), the embraced vector $e = \left[e^{(1)}, e^{(2)}, \ldots, e^{(l)}\right]$ is obtained by first drawing a vector $r^{(i)}$ from a multinomial distribution, as in Eq. (5):

$$r^{(i)} \sim \text{Multinomial}(1, \mathbf{p}) \tag{5}$$

where i is the i-th component in vectors e, r, d, and d' of same length, and p is the probability of each token being chosen to compose $e^{(i)}$, as in Eq. (6):

$$p = \left[p_1, p_2, \ldots, p_n\right] = \left[\frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N}\right] \tag{6}$$

with $\sum_n p_n = 1$. Now, assume that each vector has length l and the sampling vector $r_n = \left[r_n^{(1)}, r_n^{(2)}, \ldots, r_n^{(l)}\right]$. In order to obtain d'_n, the dot product as calculated as in Eq. (7):

$$d'_n = r_n \cdot d_n = \left[d_n'^{(1)}, d_n'^{(2)}, \ldots, d_n'^{(l)}\right] \tag{7}$$

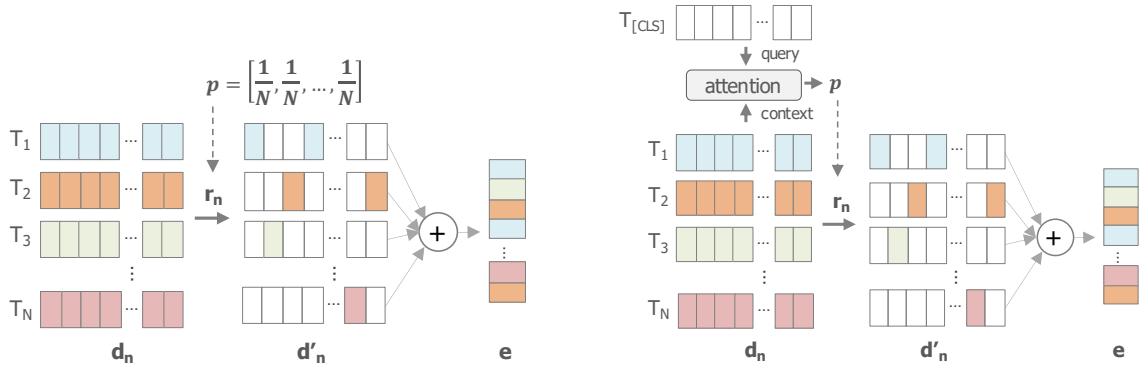
Finally, the embraced vector is obtained as follows, in Eq. (8):

$$e^{(i)} = \sum_n d_n'^{(i)} \tag{8}$$

In the proposed attentive embracement layer (Fig. 3(b)), the embraced vector $e = \left[e^{(1)}, e^{(2)}, \ldots, e^{(l)}\right]$ is obtained in a similar manner, except the probability p of each token being chosen to compose $e^{(i)}$. In the conventional method, the probability of each token being selected is the same, and by all means random. This approach has two limitations. First, in order to know the probability of each token, the user must indicate which data are missing and adjust p accordingly. Second, not all features in the data have similar importance, and this needs to be addressed for a more robust account of the data. We tackle both issues by introducing an attention layer to obtain the importance of each token $T_{i\in\{1,\ldots,N\}}$, compared to the $T_{[CLS]}$ token, to be used in the selection process, as in Eq. (9):

$$p = p_{att} = Att\left(T_{[CLS]}, [T_1, \ldots, T_N]\right) \tag{9}$$

where $T_{[CLS]}$ is the query and $T_{i\forall i\in\{1,\ldots,N\}}$ is the context component in this attention module.



(a) Conventional embracement layer

(b) Attentive embracement layer

Figure 3: We consider two types of embracement layer: (a) *conventional*, where the probability of a token being selected is is the same for every token, and (b) *attentive*, where an attention layer is used to determine the importance of each token, compared to the `[CLS]` token, to be used in the selection process.

4 Dataset

The dataset used to evaluate the model's performance is the Chatbot NLU Evaluation Corpus for intent classification, introduced by Braun et al. (Braun et al., 2017) to test NLU services. It is a publicly available[2] benchmark and is composed of sentences obtained from a German Telegram chatbot used to answer questions about public transport connections. The dataset has two intents, namely *Departure Time* and *Find Connection* with 100 train and 106 test samples[3]. Even though English is the main language of the benchmark, this dataset contains a few German station and street names.

The original dataset contains clean data, so in order to include ASR noise, we apply a text-to-speech (TTS) followed by a speech-to-text (STT) module to that data. This process is shown in Fig. 4, and it's

[2] `https://github.com/sebischair/NLU-Evaluation-Corpora`

[3] More details on the dataset distribution between classes is available in Appendix A.1.

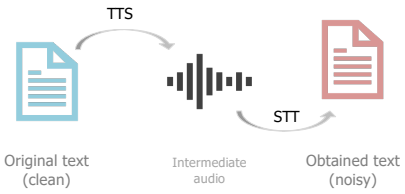


Figure 4: Diagram of 2-step process to obtain text with ASR error from clean data.

efficient due to TTS and STT modules available being imperfect, resulting in sentences with reasonable levels of ASR noise. The TTS module is called *macsay*[4], named after the terminal command *say* used to convert text to speech in the Mac OS platform. We obtain two datasets with ASR noise by applying two distinct STT modules after the TTS module: *witai*[5], freely available and maintained by Wit.ai, and *sphinx*[6], open-source Python functionality in the CMU Sphinx speech recognition engine. The mentioned TTS and STT modules are chosen according to code availability and whether it's freely available or has high daily usage limitations. The noise level in the sentences is measured quantitatively with the Word Error Rate (WER) metric, a common metric used to evaluate ASR systems where lower scores mean lower noise levels[7].

5 Experimental Results and Discussion

5.1 Training Specifications

The proposed model is fine-tuned in an end-to-end manner on the pre-trained weights from $BERT_{\text{BASE}}$: uncased, 12 transformer blocks, hidden size of 768, and 12 self-attention heads. The model is fine-tuned on a Titan X GPU for 100 epochs with Adam Optimizer, learning rate of $2 * 10^{-5}$, maximum sequence length of 128, and batch size of 8. The baseline model is $BERT_{\text{BASE}}$, with 109.5M number of parameters, and the proposed model has number of parameters varying from 109.5M to 117.2M (see Table 2 for more detailed information). Each model is run 10 times, with the results being shown as mean and standard deviation.

5.2 Results

For the performance evaluation, we calculate the F1-scores for all models fine-tuned on the Chatbot corpus for intent classification for three settings: (1) trained and tested with clean data, (2) trained with clean and tested with noisy data, and (3) trained and tested with noisy data (witai or sphinx STT noise). Results are shown in Table 1 in order of containing low to higher noise.

For a fair comparison, we consider two baseline models: BERT using only the `[CLS]` token for classification and BERT using all tokens, with structures shown in Fig. 5. Results show improved performance with BERT using all tokens in settings containing noisy text, with the biggest improvement in setting 2.

We consider a few combinations of the proposed model, which are grouped into the conventional embracement layer approach or the novel attentive approach, with an expanded analysis in the following section. Results in Table 1 show that our model outperforms the baseline models in all settings. In settings 1 and 2, the best model is the one with conventional embracement and multi-head attention layers, where bigger improvement can be noticed in datasets with higher noise level. From lower to higher noise, clean data shows improvement of 0.19 points, 'witai' with WER of 3.11 shows improvement of 2.83 points and 'sphinx' with WER of 6.58 shows improvement of 3.30 points. In setting 3, the model with attentive embracement and multi-head attention layers achieves the best performance, with 'witai' showing improvement of 1.70 and 'sphinx' 1.42 points. We hypothesize that the conventional embracement is able to obtain better results in setting 2 because it retains more general characteristics from data

[4] `https://ss64.com/osx/say.html`
[5] `https://wit.ai`
[6] `https://cmusphinx.github.io/wiki/`
[7] More details on the WER metric and examples from the dataset with ASR noise can be found in the Appendix.

by giving equal weights to all tokens, regardless of its importance. Whereas in the attentive embracement approach, if a token deemed important is absent during testing, this might hinder performance. On the other hand, attentive embracement shows the strength of token differentiation in setting 3 and its ability to extract a robust representation of noisy inputs for improved text classification.

Model	**Train/Test clean**$^{(1)}$	**Train clean/Test noisy**$^{(2)}$		**Train/Test noisy**$^{(3)}$	
		witai	sphinx	witai	sphinx
	(0.00)	(3.11)	(6.58)	(3.11)	(6.58)
Only CLS token					
BERT	98.49 ± 0.86	88.11 ± 2.90	76.42 ± 7.22	93.21 ± 2.14	87.45 ± 1.94
All tokens (ours)					
BERT	98.02 ± 1.43	89.72 ± 3.49	78.11 ± 4.21	93.58 ± 2.10	87.64 ± 3.17
Conventional embracement					
EBERT(no QKV)	98.58 ± 1.05	90.28 ± 3.58	76.98 ± 7.26	94.43 ± 1.15	87.83 ± 1.66
EBERT	$\mathbf{98.68 \pm 0.63}$	$\mathbf{90.94 \pm 3.30}$	$\mathbf{79.72 \pm 4.78}$	93.87 ± 1.80	88.49 ± 2.14
Attentive embracement					
EBERT(no QKV)	98.49 ± 0.46	90.19 ± 2.36	76.89 ± 6.28	93.96 ± 2.12	87.36 ± 1.75
EBERT	98.49 ± 0.86	89.34 ± 2.67	77.55 ± 5.01	$\mathbf{94.91 \pm 1.21}$	$\mathbf{88.87 \pm 1.62}$
Improvement	+0.19	+2.83	+3.30	+1.70	+1.42

Table 1: F1-scores (%), in terms of mean and standard deviation between 10 runs, are calculated on the Chatbot corpus for three settings: (1) trained and tested with clean data, (2) trained with clean and tested with noisy data, and (3) trained and tested with noisy data (witai or sphinx STT noise). Each dataset is accompanied by a WER score under it to indicate the noise level, where higher values means higher noise. Note that the BERT baseline model only uses the `[CLS]` token for text classification, whereas the remaining models use all tokens including `[CLS]`. We also group the models into conventional and attentive embracement for more clarity.

5.3 Ablation Study

In this section, we provide an extensive ablation study on the models that use all tokens for text classification. Fig. 5 shows the different architectures using BERT in order for it to use all tokens in the classification task. The simplest architecture, Fig. 5(a), consists of adding a dimensionality reduction layer, either attention or projection, to obtain a single token for classification. However, the limitation of that approach is that it considers the `[CLS]` token as having the same importance as the remaining tokens. Since this token is used by itself in the conventional BERT classification task, it is fair to assume that the starter token holds very meaningful features and should be considered in higher regard. Thus, the second approach in Fig. 5(b) consists of projecting tokens $T_{i \forall i \in \{1,...,N\}}$ into an intermediate vector and using an attention layer with that intermediate vector and the `[CLS]` token to obtain the final T'_C token for classification. The last approach, Fig. 5(c), consists of adding an attention layer with the `[CLS]` token as query and the remaining as context to extract an importance feature vector, which is then projected with the `[CLS]` token into the final T'_C token. Results in Table 2 show that the approach in Fig. 5(c) outperforms other combinations in all settings. We hypothesize that by extracting important features from the remaining tokens, when compared to the starter token, the model is able to retain meaningful information that would have been otherwise discarded.

Fig. 6 shows the different architectures used in the ablation study of EBERT. Here, we investigate the importance of the conventional and attentive embracement layers, multi-head attention layer, and different types of dimensionality reduction methods. We also evaluate the architecture that concatenates the intermediate feature vector obtained the from attention layer and the one obtained from the embracement layer. The architecture in Fig. 6(a) extracts intermediate information from a multi-head attention layer and a conventional embracement layer, followed by a dimensionality reduction layer which can either be projection or attention. In Fig. 6(b), we concatenate the intermediate feature vector obtained with the best performing architecture in Fig. 5(c) and the intermediate feature vector obtained from a multi-head attention followed by a conventional embracement layer. Figs. 6(c) and (d) are similar to (a) and (b), respectively, except that the embracement layer follows an attentive approach.

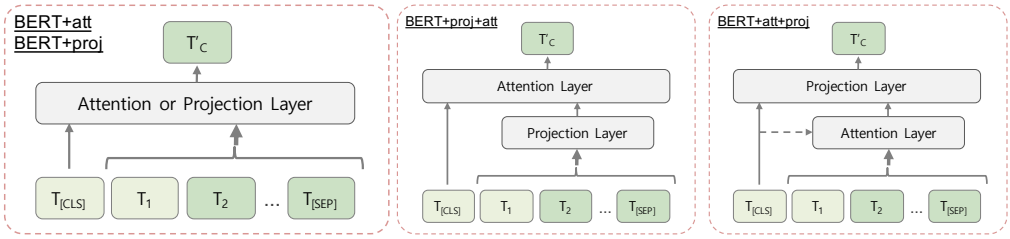


Figure 5: Different architectures in the ablation study of BERT using all tokens: (a) attention (BERT+att) or projection (BERT+proj) layer for dimensionality reduction, (b) projection layer with tokens $T_{i \forall i \in \{1,...,N\}}$ followed by an attention layer for dimensionality reduction (BERT+proj+att), and (c) attention layer for extraction of an importance feature vector to be projected with the `[CLS]` token into the final T'_C token (BERT+att+proj).

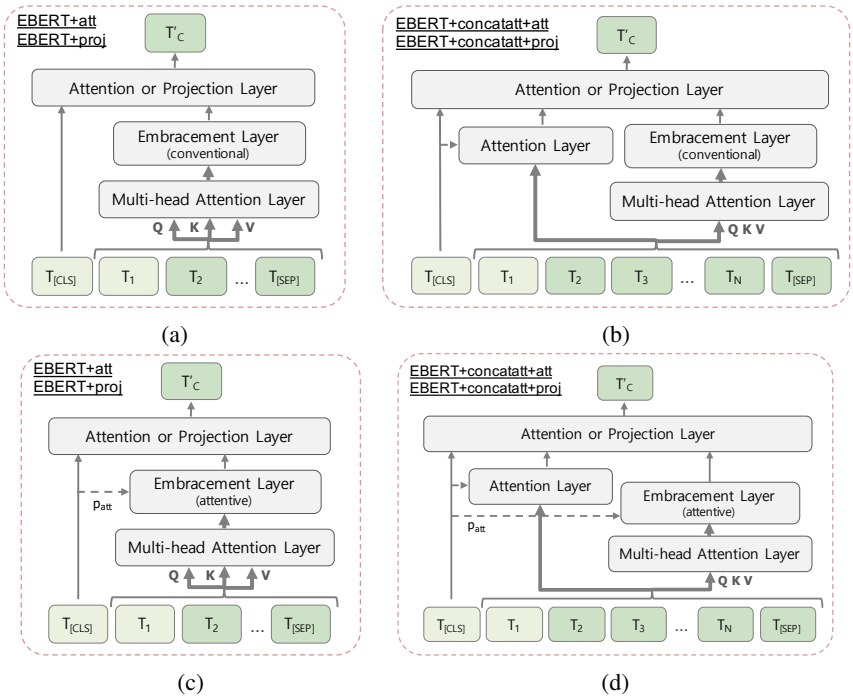


Figure 6: Different architectures used in the ablation study of EBERT: (a) multi-head attention and conventional embracement layer followed by an attention (EBERT+att) or projection (EBERT+proj) layer for dimensionality reduction, (b) concatenation of (a) and the best performing architecture in Fig. 5(c) (EBERTconcatt+att or EBERTconcatt+proj), (c) and (d) are the same as (a) and (d), respectively, but with an attentive embracement layer. Note that the models used in the ablation study for EBERT without the multi-head attention layer (EBERT(no QKV)) are essentially the same except for the absence of the mentioned layer.

Results in Table 2 show that EBERT, using architecture in Fig. 6(d) with an attentive embracement layer and projection layer for dimensionality reduction, outperforms all of the models in the third setting. In setting 1, the best performance can be seen from the same model with conventional embracement layer. Lastly, in setting 2, EBERT with conventional embracement and projection layer, Fig. 6(a), achieves the best scores. This study shows the importance of the embracement and multi-head attention layer combination to achieve high performance in settings with clean and noisy data. This is especially true in settings with noisy data in both training and testing. Furthermore, Table 2 also shows that this increase in performance is achieved with just a slight increase in computational complexity, measured in the number of parameters.

Model	Number of parameters	Train/Test clean(1)	Train clean/Test noisy(2)		Train/Test noisy(3)	
			witai	sphinx	witai	sphinx
		(0.00)	(3.11)	(6.58)	(3.11)	(6.58)
		Only CLS token				
BERT	109.5M	98.49 ± 0.86	88.11 ± 2.90	76.42 ± 7.22	93.21 ± 2.14	87.45 ± 1.94
		All tokens (ours)				
BERT+att	111.3M	97.83 ± 0.85	83.58 ± 7.21	67.26 ± 11.20	93.02 ± 1.35	86.42 ± 1.90
BERT+att+proj	111.3M	98.02 ± 1.43	89.72 ± 3.49	78.11 ± 4.21	93.58 ± 2.10	87.64 ± 3.17
BERT+proj	109.5M	97.92 ± 1.18	87.08 ± 3.73	73.40 ± 5.42	91.79 ± 3.63	87.17 ± 1.53
BERT+proj+att	111.3M	97.55 ± 1.59	85.09 ± 6.69	69.34 ± 11.73	92.55 ± 1.71	86.70 ± 2.13
Conventional embracement						
EBERT(no QKV)+att	111.3M	97.83 ± 1.27	86.51 ± 5.27	71.89 ± 9.81	92.92 ± 0.97	86.23 ± 1.13
EBERT(no QKV)+proj	109.5M	98.58 ± 1.05	89.81 ± 3.40	73.77 ± 8.87	94.43 ± 1.15	87.83 ± 1.66
EBERT(no QKV)+concatatt+att	113.0M	97.92 ± 1.10	86.98 ± 5.70	69.43 ± 8.48	92.17 ± 1.52	85.66 ± 3.09
EBERT(no QKV)+concatatt+proj	111.3M	98.02 ± 0.78	90.28 ± 3.58	76.98 ± 7.26	93.68 ± 1.84	87.26 ± 1.35
EBERT+att	113.6M	97.55 ± 1.41	85.66 ± 4.32	69.72 ± 7.85	92.26 ± 2.52	85.75 ± 1.36
EBERT+proj	111.8M	98.40 ± 0.95	$\mathbf{90.94 \pm 3.30}$	$\mathbf{79.72 \pm 4.78}$	93.11 ± 1.98	87.55 ± 2.79
EBERT+concatatt+att	115.4M	97.55 ± 1.13	88.49 ± 5.02	75.38 ± 7.26	92.74 ± 1.20	87.08 ± 2.07
EBERT+concatatt+proj	113.6M	$\mathbf{98.68 \pm 0.63}$	90.09 ± 1.70	76.89 ± 5.24	93.87 ± 1.80	88.49 ± 2.14
Attentive embracement						
EBERT(no QKV)+att	113.0M	97.83 ± 1.69	86.89 ± 4.37	72.08 ± 8.72	92.45 ± 2.42	86.51 ± 3.01
EBERT(no QKV)+proj	111.3M	98.40 ± 1.27	90.19 ± 2.36	76.89 ± 6.28	93.96 ± 2.12	87.36 ± 1.75
EBERT(no QKV)+concatatt+att	114.8M	97.92 ± 0.71	86.70 ± 5.52	70.94 ± 7.65	92.17 ± 1.40	86.51 ± 2.46
EBERT(no QKV)+concatatt+proj	113.0M	98.49 ± 0.46	89.06 ± 3.02	76.70 ± 4.39	93.40 ± 1.46	86.79 ± 2.67
EBERT+att	115.4M	98.21 ± 0.78	84.72 ± 3.86	70.28 ± 5.37	92.17 ± 1.89	87.17 ± 2.32
EBERT+proj	113.6M	98.11 ± 1.12	89.34 ± 2.67	77.55 ± 5.01	93.68 ± 1.04	87.64 ± 1.30
EBERT+concatatt+att	117.2M	97.36 ± 1.02	86.60 ± 4.79	73.68 ± 7.42	92.55 ± 1.71	86.60 ± 1.73
EBERT+concatatt+proj	115.4M	98.49 ± 0.86	89.06 ± 4.20	74.06 ± 10.53	$\mathbf{94.91 \pm 1.21}$	$\mathbf{88.87 \pm 1.62}$

Table 2: Ablation study with F1-scores (%) and WER metric on the Chatbot corpus for three settings: (1) trained and tested with clean data, (2) trained with clean and tested with noisy data, and (3) trained and tested with noisy data (witai or sphinx STT noise). Results are shown in terms of mean and standard deviation between 10 runs. Note that the BERT baseline model only uses the `[CLS]` token for text classification, whereas the remaining models use all tokens including `[CLS]`. We also group the models into conventional and attentive embracement for more clarity.

5.3.1 Study on a larger dataset

As part of our ablation study, we also provide detailed evaluation results on the Snips NLU Corpus (Coucke et al., 2018)[8], a large dataset containing 15K crowd-sourced queries with a total of 7 intents, namely *Add To Playlist*, *Book Restaurant*, *Get Weather*, *Play Music*, *Rate Book*, *Search Creative Work*, and *Search Screening Event*, totalling in 13,784 train and 700 test samples[9]. The noisy version of this dataset includes ASR error added to the data in the two-step process previously shown in Fig. 4.

We train the same models listed in Table 2 on the Snips dataset and with training specifications discussed in Section 5.1. The only modification being that we use a larger batch size of 32 for faster training. The ablation study in Table 3 shows that our proposed approach, of using all tokens for text classification, results in better performance over all three settings when trained on the larger dataset. In setting 1, it can be seen that all but one EBERT variant outperforms the baseline model. We also evaluate the models' performance on noisy data, with the results showing improved performance of +0.47 for 'witai' and +2.13 for 'sphinx' on unseen noisy data (setting 2), and improved classification result when trained with noisy data (setting 3) albeit with a smaller improvement margin.

Conclusion

We proposed a novel BERT-based model, called EBERT, that improves robustness in the task of noisy ASR text classification. Conventionally, when fine-tuning BERT for downstream classification tasks, only the `[CLS]` starter token is used, with the remaining tokens being discarded. Our model used those otherwise discarded tokens as a source of additional information, together with a multi-head attention and attentive embracement layer, to more efficiently extract robust latent representations in noisy data.

[8] `https://github.com/snipsco/nlu-benchmark`

[9] The dataset distribution is available in Appendix A.2.

Model	Number of parameters	Train/Test clean[1]	Train clean/Test noisy[2]		Train/Test noisy[3]	
			witai	sphinx	witai	sphinx
		(0.00)	(2.66)	(7.24)	(2.66)	(7.24)
		Only CLS token				
BERT	109.5M	98.67 ± 0.16	93.86 ± 0.66	48.93 ± 1.71	96.54 ± 0.20	91.63 ± 0.65
		All tokens (ours)				
BERT+att	111.3M	98.80 ± 0.13	94.04 ± 0.62	49.16 ± 1.69	96.41 ± 0.30	91.64 ± 0.69
BERT+att+proj	111.3M	98.64 ± 0.13	94.21 ± 0.61	49.64 ± 1.99	96.49 ± 0.23	91.20 ± 0.51
BERT+proj	109.5M	$\mathbf{98.87 \pm 0.17}$	93.99 ± 0.40	49.46 ± 1.36	96.46 ± 0.43	91.40 ± 0.68
BERT+proj+att	111.3M	98.74 ± 0.12	94.24 ± 0.42	49.37 ± 1.76	96.64 ± 0.19	91.26 ± 0.57
Conventional embracement						
EBERT(no QKV)+att	111.3M	98.73 ± 0.12	93.69 ± 0.57	48.74 ± 2.63	96.50 ± 0.31	90.83 ± 0.73
EBERT(no QKV)+proj	109.5M	98.67 ± 0.14	93.93 ± 0.42	50.73 ± 1.78	96.47 ± 0.23	91.31 ± 0.51
EBERT(no QKV)+concatatt+att	113.0M	98.67 ± 0.13	94.10 ± 0.48	47.80 ± 1.57	96.39 ± 0.23	91.61 ± 0.62
EBERT(no QKV)+concatatt+proj	111.3M	98.70 ± 0.27	94.10 ± 0.29	50.61 ± 1.82	96.49 ± 0.17	91.17 ± 0.36
EBERT+att	113.6M	98.74 ± 0.14	94.11 ± 0.42	49.09 ± 2.17	96.39 ± 0.31	91.50 ± 0.53
EBERT+proj	111.8M	98.74 ± 0.17	93.84 ± 0.61	$\mathbf{51.06 \pm 1.55}$	96.46 ± 0.19	$\mathbf{91.70 \pm 0.37}$
EBERT+concatatt+att	115.4M	98.67 ± 0.13	93.81 ± 0.58	49.54 ± 1.54	96.44 ± 0.27	91.26 ± 0.61
EBERT+concatatt+proj	113.6M	98.66 ± 0.13	94.01 ± 0.53	50.83 ± 1.48	96.59 ± 0.32	91.66 ± 0.44
Attentive embracement						
EBERT(no QKV)+att	113.0M	98.79 ± 0.17	93.97 ± 0.35	48.04 ± 1.76	96.53 ± 0.26	91.36 ± 0.98
EBERT(no QKV)+proj	111.3M	98.70 ± 0.19	94.13 ± 0.50	50.34 ± 1.62	96.36 ± 0.30	91.37 ± 0.40
EBERT(no QKV)+concatatt+att	114.8M	98.70 ± 0.19	93.89 ± 0.50	48.09 ± 1.11	96.40 ± 0.35	91.46 ± 0.94
EBERT(no QKV)+concatatt+proj	113.0M	98.64 ± 0.16	93.93 ± 0.58	50.19 ± 1.16	$\mathbf{96.76 \pm 0.22}$	91.50 ± 0.52
EBERT+att	115.4M	98.69 ± 0.17	94.16 ± 0.38	49.34 ± 1.75	96.34 ± 0.23	91.00 ± 0.59
EBERT+proj	113.6M	98.70 ± 0.13	93.89 ± 0.44	49.76 ± 1.55	96.63 ± 0.26	91.20 ± 0.52
EBERT+concatatt+att	117.2M	98.76 ± 0.18	94.07 ± 0.33	48.44 ± 1.95	96.23 ± 0.24	91.46 ± 0.51
EBERT+concatatt+proj	115.4M	98.73 ± 0.21	$\mathbf{94.33 \pm 0.49}$	50.21 ± 0.91	96.59 ± 0.19	91.36 ± 0.45

Table 3: Ablation study with F1-scores (%) and WER metric on the Snips NLU corpus for three settings. This table was obtained similarly to Table 2 but on a larger dataset, so for more details on this study, please check the caption in Table 2.

We evaluated our model in three settings: trained and tested with clean data, trained with clean and tested with noisy data, and trained and tested with noisy data. Results on the Chatbot corpus for intent classification, in terms of F1-score and mean between 10 runs, showed that our model outperforms the baseline models in all settings. In settings 1 and 2, the best model was the one with conventional embracement and multi-head attention layers, with bigger improvement in datasets with higher noise level. In setting 3, the model with attentive embracement and multi-head attention layers achieved the best performance. We hypothesize that conventional embracement was able to obtain better results in setting 2 because it retains more general characteristics from data by giving equal weights to all tokens, regardless of its importance. On the other hand, attentive embracement showed the importance of token differentiation in setting 3 and its ability to extract a robust representation of noisy inputs. This increase in performance is achieved with just a slight increase in computational complexity, measured in the number of parameters. We additionally provided an extensive ablation study showing the importance of the embracement and multi-head attention layer combination to achieve high performance in settings with both clean and noisy data, and we also reinforce the importance of our approach in a larger dataset. In the future, we plan on evaluating our approach with other BERT variations and we plan on considering the effect of clean and noisy data together in the training stage.

Acknowledgements

This work was partly supported by Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (2016-0-00564, Development of Intelligent Interaction Technology Based on Context Awareness and Human Intention Understanding) (50%) and Electronics and Telecommunications Research Institute(ETRI) grant funded by the Korea government(MOTIE) [20ZS1100, Core Technology Research for Self-Improving Integrated Artificial Intelligence System] (50%).

References

Sumeet Agarwal, Shantanu Godbole, Diwakar Punjani, and Shourya Roy. 2007. How much noise is too much: A study in automatic text classification. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 3–12. IEEE.

Saliha Benkerzaz, Youssef Elmir, and Abdeslam Dennai. 2019. A study on automatic speech recognition. *Journal of Information Technology Review*, 10(3):77–85, 08.

Daniel Braun, Adrian Hernandez-Mendez, Florian Matthes, and Manfred Langen. 2017. Evaluating natural language understanding services for conversational question answering systems. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 174–185.

Jun-Ho Choi and Jong-Seok Lee. 2019. Embracenet: A robust deep learning architecture for multimodal classification. *Information Fusion*, 51:259–270.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Li Deng and Xiao Li. 2013. Machine learning paradigms for speech recognition: An overview. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(5):1060–1089.

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

Rahhal Errattahi, Asmaa El Hannani, and Hassan Ouahmane. 2018. Automatic speech recognition errors detection and correction: A review. *Procedia Computer Science*, 128:32–37.

Adeline Granet, Emmanuel Morin, Harold Mouchère, Solen Quiniou, and Christian Viard-Gaudin. 2018. Transfer learning for a letter-ngrams to word decoder in the context of historical handwriting recognition with scarce resources. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1474–1484, Santa Fe, New Mexico, USA, August. Association for Computational Linguistics.

Oleksii Hrinchuk, Mariya Popova, and Boris Ginsburg. 2020. Correction of automatic speech recognition with transformer sequence-to-sequence model. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7074–7078. IEEE.

Junwei Liao, Sefik Emre Eskimez, Liyang Lu, Yu Shi, Ming Gong, Linjun Shou, Hong Qu, and Michael Zeng. 2020. Improving readability for automatic speech recognition transcription. *arXiv preprint arXiv:2004.04438*.

Ali Bou Nassif, Ismail Shahin, Imtinan Attili, Mohammad Azzeh, and Khaled Shaalan. 2019. Speech recognition using deep neural networks: A systematic review. *IEEE Access*, 7:19143–19165.

Jayashree Padmanabhan and Melvin Jose Johnson Premkumar. 2015. Machine learning in automatic speech recognition: A survey. *IETE Technical Review*, 32(4):240–251.

Gwenaelle Cunha Sergio, Dennis Singh Moirangthem, and Minho Lee. 2018. Temporal hierarchies in sequence to sequence for sentence correction. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE.

Niraj Shrestha, Elias Moons, and Marie-Francine Moens. 2019. Using related text sources to improve classification of transcribed speech data. In *International Conference on Advanced Machine Learning Technologies and Applications*, pages 507–517. Springer.

L Venkata Subramaniam. 2010. Noisy text analytics. In *NAACL (Tutorial Abstracts)*, pages 5–6.

Mutsuko Tomokiyo, Christian Boitet, and Mathieu Mangeot. 2018. Towards an automatic classification of illustrative examples in a large japanese-french dictionary obtained by ocr. In *First Workshop on Linguistic Resources for Natural Language Processing*, pages 112–121.

Amrita S Tulshan and Sudhir Namdeorao Dhage. 2018. Survey on virtual assistant: Google assistant, siri, cortana, alexa. In *International Symposium on Signal Processing and Intelligent Recognition Systems*, pages 190–201. 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.

Alessandro Vinciarelli. 2005. Noisy text categorization. *IEEE transactions on pattern analysis and machine intelligence*, 27(12):1882–1895.

Farshid Zavareh, Ingrid Zukerman, Su Nam Kim, and Thomas Kleinbauer. 2013. Error detection in automatic speech recognition. In *Proceedings of the Australasian Language Technology Association Workshop 2013 (ALTA 2013)*, pages 101–105, Brisbane, Australia, December.

A Datasets

A.1 Chatbot NLU Evaluation Corpus

Table 4 shows the dataset distribution between classes and train/test on the Chatbot NLU Evaluation Corpus (Braun et al., 2017) and Table 5 shows some examples of clean and their respective noisy sentences with different TTS-STT combinations, therefore varying rates of noise.

Class (Intent)	Train	Test	Total
Departure Time	43	35	98
Find Connection	57	71	128
Total	100	106	206

Table 4: Details on the Chatbot NLU Evaluation Corpus' dataset distribution between classes and train/test division.

STT	WER	Original sentence	With ASR error
witai	3.11	"how can i get from moosach to	"how can i get from quiddestraße."
sphinx	6.58	quiddestraße?"	"can i get from los at lives three."
witai	3.11	"how to get from alte heide to	"how to get from altona to maryland."
sphinx	6.58	marienplatz"	"call now from outside to memory and flaps."
witai	3.11	"next bus from central station"	"next bus from central station."
sphinx	6.58		"there are strong central station."

Table 5: Examples of sentence from Chatbot NLU Corpus with different TTS(macsay)-STT(witai, sphinx) combinations and their respective WER score, which denotes the level of ASR noise in the text. The datasets are shown in order of lower to higher noise.

A.2 Snips NLU Corpus

Table 6 shows the dataset distribution between classes and train/test on the Snips NLU Corpus (Coucke et al., 2018), and Table 7 shows some examples of clean and their respective noisy sentences with different TTS-STT combinations.

Class (Intent)	Train	Test	Total
Add To Playlist	1,942	100	2,042
Book Restaurant	1,973	100	2,073
Get Weather	2,000	100	2,100
Play Music	2,000	100	2,100
Rate Book	1,956	100	2,056
Search Creative Work	1,954	100	2,054
Search Screening Event	1,959	100	2,059
Total	13,784	700	14,484

Table 6: Details about the Snips NLU Corpus' distribution between classes and train/test division.

STT	WER	Original sentence	With ASR error
witai	2.66	"what are the movie times"	"what are the movie times."
sphinx	7.24		"we're tarzan movie times."
witai	2.66	"will Custer National Forest be chillier	"will there national forest be chillier at seven pm."
sphinx	7.24	at seven Pm"	"will cause a national far is still you're upset him."
witai	2.66	"Book a reservation for an oyster bar"	"reservation for an oyster bar."
sphinx	7.24		"for reservations for an oyster bar."

Table 7: Examples of sentence from Snips NLU Corpus with different TTS(macsay)-STT(witai, sphinx) combinations and their respective WER score, which denotes the level of ASR noise in the text. The datasets are shown in order of lower to higher noise.

B WER metric

The noise level in the sentences is measured quantitatively with the WER metric, a common metric used to evaluate ASR systems where lower scores mean lower noise levels. This metric calculates the minimum number of edits required to change a candidate sentence into a reference sentence. Mathematically speaking, WER considers words substitution S, deletion D and insertion I compared to the total number of words N in the reference, as shown in Eq. (10):

$$WER = \frac{S + D + I}{S + D + C} = \frac{S + D + I}{N} \tag{10}$$

where C the number of the correct words. The lower WER, the better, since it means less editions are needed for the sentence to be converted to the gold standard. The lowest possible value in this metric is 0, with no ceiling for its maximum value.